E D I C E

KOMUNIKACE & SÍTĚ

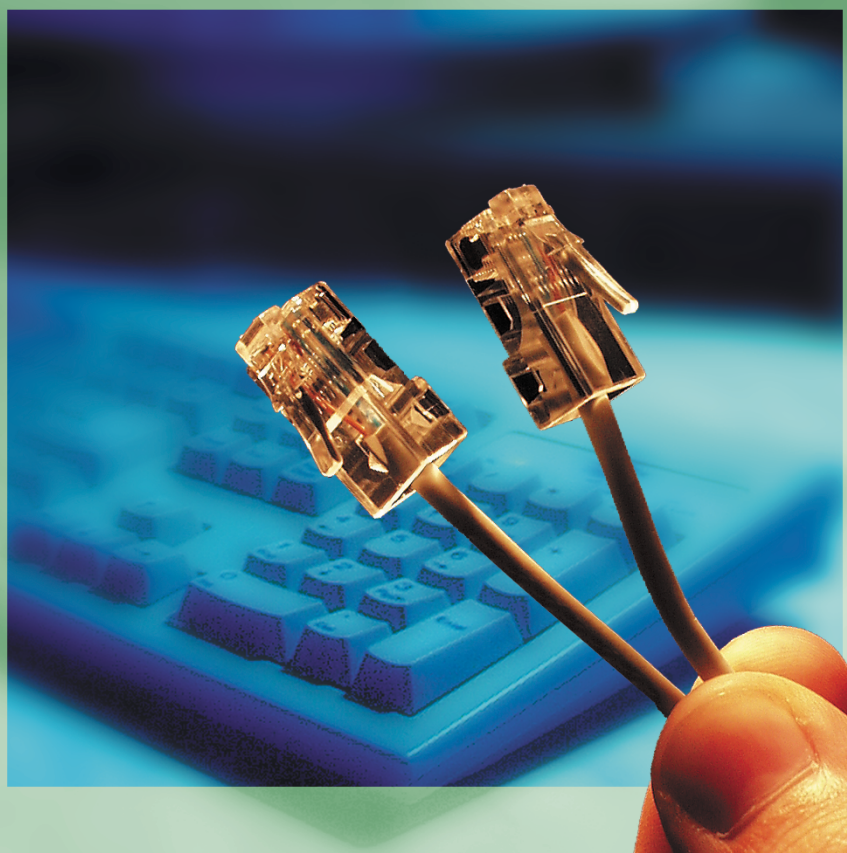
PROFI

druhé aktualizované vydání

Libor Dostálek, Alena Kabelová

Velký průvodce protokoly TCP/IP a systémem DNS

- Vysvětlení nejpoužívanějších protokolů a konfigurace současných sítí
- Internet i vnitřní podnikové sítě
- Delegace domén, přidělování IP-adres, firewally, GSM
- Aktualizováno pro Windows 2000



CD-ROM

Přiložené CD obsahuje knihu v HTML, dokumentaci k protokolům, výukové texty k MIME, CGI a Perlu a množství užitečného softwaru, např. programy Antisniff, AutoPing, GeoBoy, Jaggernaut a mnoho dalších.

computer press

VŠECHNY CESTY K INFORMACÍM

Libor Dostálek
Alena Kabelová

Velký průvodce protokoly TCP/IP a systémem DNS

Computer Press
Praha
2000

Libor Dostálek
Alena Kabelová

Velký průvodce protokoly TCP/IP a systémem DNS

- **Vysvětlení nejpoužívanějších protokolů a konfigurace současných sítí**
- **Internet i vnitřní podnikové sítě**
- **Delegace domén, přidělování IP-adres, firewally, GSM**
- **Aktualizováno pro Windows 2000**

Velký průvodce protokoly TCP/IP a systémem DNS,
2. aktualizované vydání

Libor Dostálek, Alena Kabelová

Vydavatelství a nakladatelství Computer Press®,
Hornocholupická 22, 143 00 Praha 4, http://www.cpress.cz

ISBN 80-7226-323-4

Prodejní kód: K0410

Odborná korektura: Computer Help
Jazyková korektura: Josef Novák
Vnitřní úprava: Petr Klíma
Sazba: Petr Klíma
Rejstřík: Libor Dostálek
Obálka: Pavel Drinka
Komentář na zadní straně obálky: Jakub Mácha

Technická spolupráce: Jiří Matoušek, Pavlína Bauerová
Odpovědný redaktor: Ivo Magera
Vedoucí technické redakce: Martin Hanslian
Vedoucí knižní redakce: Ivo Magera
Vedoucí produkce: Kateřina Vobecká

Tisk: PBTISK

Veškeré dotazy týkající se distribuce směřujte na:

Computer Press Brno, náměstí 28. dubna 48, 635 00 Brno-Bystrc,
tel. (05) 46 12 21 11, e-mail: distribuce@cpress.cz

Computer Press Bratislava, Hattalova 12/A, 831 03 Bratislava, Slovenská republika,
tel.: +421 (7) 44 45 20 48, e-mail: distribucia@cpress.sk

Nejnovější informace o našich publikacích naleznete na adrese:
http://www.cpress.cz/knihy/bulletin.html.
Máte-li zájem o pravidelné zasílání bulletinu do Vaší e-mailové schránky, zašlete nám jakoukoli i prázdnou zprávu na adresu bulletin@cpress.cz.

Obsah

1

Síťové protokoly

Podobně jako diplomaté při svých jednáních používají diplomatický protokol, tak i počítače v počítačových sítích používají pro vzájemnou komunikaci síťové protokoly. Síťových protokolů existuje celá řada. V Internetu se používají síťové protokoly TCP/IP.

Síťový protokol je norma napsaná na papíře (resp. textovým editorem na počítači). V Internetu se používají normy nazývané Request For Comments – zkratkou RFC, které se číslují průběžně od jedničky. V současné době jich jsou necelé tři tisíce. Mnohé však postupem času zastaraly, takže z první tisícovky jich je aktuálních jen několik.

Mezinárodní normalizační úřad (ISO) normalizoval soustavu protokolů označovaných jako ISO OSI. Další slovutnou organizací vydávající normy v oblasti komunikací je ITU se sídlem v Ženevě (dříve CCITT – nejstarší celosvětová organizace vůbec, založena 1865). Dále se setkáme s normami vydanými sdružením elektrotechnických inženýrů IEEE. Běžný uživatel se však může dostat pouze k normám RFC, protože ostatní organizace neposkytují své normy zdarma. *

Nejprve si musíme objasnit, proč je problematika komunikace mezi počítači vždy rozdělena do více protokolů. Odpověď je velice jednoduchá, celá problematika je velice komplikovaná a pokrývá několik profesí. Většina publikací o síťových protokolech uvádí přirovnání ke komunikaci dvou cizinců (či filozofů, šamanů apod.), kteří umí každý jen svůj jazyk. Aby si vyměnili své myšlenky, musí si každý obstarat překladatelku do společného jazyka – např. do češtiny. Viz obr. 1.1.

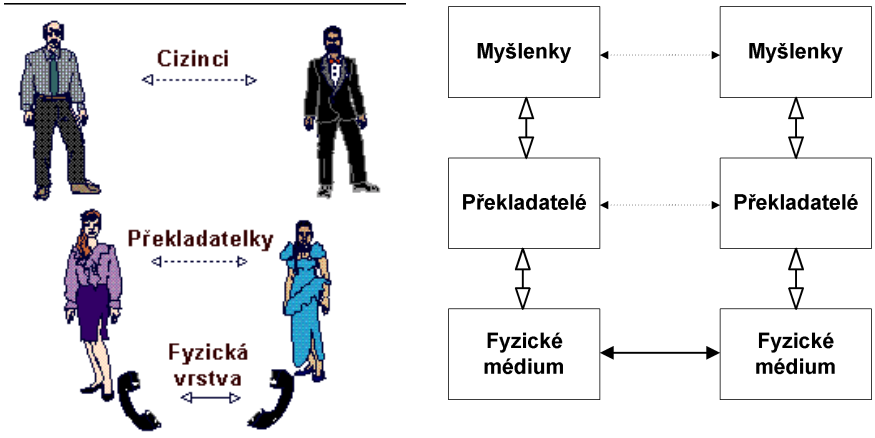
Vzájemně si oba cizinci předávají své myšlenky, tj. komunikují mezi sebou. Jenže mezi sebou komunikují jen pomyslně (virtuálně). Ve skutečnosti oba své informace předávají překladatelkám, a ty pak pomocí svých hlasivek rozvlní vzduch, aby přenesly informace. Nebo jsou obě strany vzdáleny a překladatelé komunikují pomocí telefonu, pak se informace fyzicky přenášejí po telefonních linkách.

Rozeznáváme virtuální komunikaci ve vodorovném směru (filozofickou, společným jazykem mezi překladatelkami a elektrickými signály po telefonním vedení) a skutečnou komunikaci ve svislém směru, tj. cizinec – překladatel a překladatel – telefon. Rozlišujeme tedy celkem tři vrstvy komunikace:

- Komunikace mezi cizinci
- Komunikace mezi překladatelkami
- Fyzický přenos informací po médiu (např. telefonní vedení, zvukové vlny atp.)

* Na přiloženém CD jsou normy RFC, RIPE, PKCS a WAP, které byly aktuální ke dni předání rukopisu.

Obr. 1.1 Třívrstvá komunikační architektura

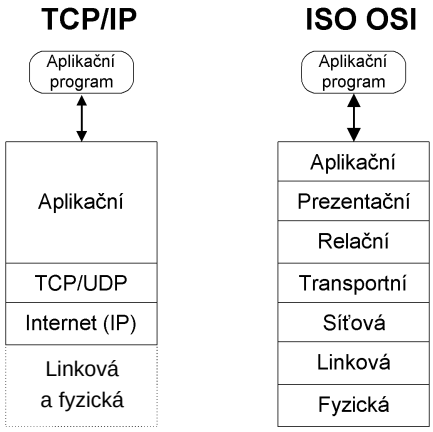


Komunikace cizinec – cizinec a překladatel – překladatel je pouze pomyslná (virtuální). Ve skutečnosti (reálně) komunikuje cizinec s překladatelem.

V počítačových sítích používáme ještě více vrstev. Počet vrstev závisí na tom, jakou soustavu síťových protokolů použijeme. Místo o soustavě síťových protokolů někdy též mluvíme o tzv. síťovém modelu. Nejčastěji se budeme setkávat s modelem, který používá Internet, tento model se též nazývá rodinou protokolů TCP/IP. Kromě protokolů TCP/IP se setkáme ještě s modelem ISO OSI, který standardizoval mezinárodní standardizační úřad (ISO).

Rodina protokolů TCP/IP využívá čtyři vrstvy a protokoly ISO OSI používají vrstev dokonce sedm, jak je znázorněno na obr. 1.2.

Obr. 1.2 Porovnání síťových modelů TCP/IP a ISO OSI



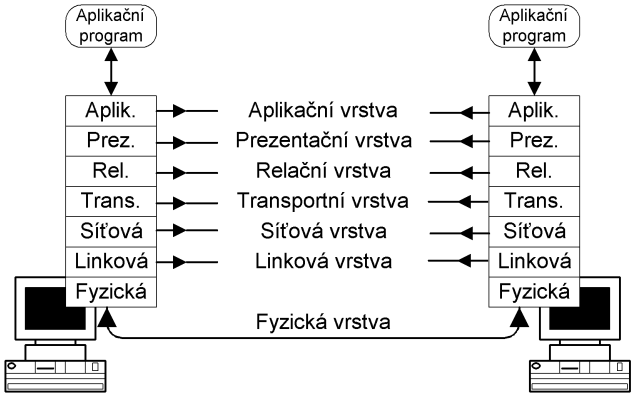
Soustavy síťových protokolů TCP/IP a ISO OSI se od sebe liší – jsou vzájemně neporovnatelné. Z obrázku 1.2 je však patrné, že na síťové a transportní vrstvě jsou si velmi blízké.

Rodina síťových protokolů TCP/IP neřeší (až na výjimky, jako je protokol SLIP) linkovou a fyzickou vrstvu, proto se i v Internetu setkáváme s linkovými a fyzickými protokoly z modelu ISO OSI.

1.1 ISO OSI

Komunikace mezi dvěma počítači je schématicky znázorněna na obr. 1.3.

Obr. 1.3 Sedmivrstvá architektura ISO OSI



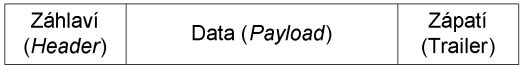
1.1.1 Fyzická vrstva

Fyzická vrstva popisuje elektrické či optické signály používané při komunikaci mezi počítači. Na fyzické vrstvě je vytvořen tzv. fyzický okruh. Na fyzický okruh mezi dva počítače bývají často vkládána další zařízení, např. modemy, které modulují signál na telefonní vedení atp.

1.1.2 Linková vrstva

Linková vrstva zajišťuje v případě sériových linek výměnu dat mezi sousedními počítači a v případě lokálních sítí výměnu dat v rámci lokální sítě.

Obr. 1.4 Linkový rámec



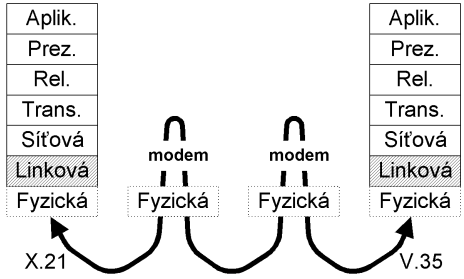
Základní jednotkou pro přenos dat je na linkové vrstvě datový rámec (viz obr. 1.4). Datový rámec se skládá ze záhlaví (*Header*), přenášených dat (*Payload*) a zápatí (*Trailer*).

Datový rámec nese v záhlaví linkovou adresu příjemce, linkovou adresu odesílatele a další řídící informace. V zápatí nese mj. obvykle kontrolní součet z přenášených dat. Pomocí něho lze zjistit, zdali nedošlo při přenosu k porušení dat. V přenášených datech je pak zpravidla nesen paket síťové vrstvy.

Z obr. 1.5 je vidět, že na fyzické vrstvě mohou být pro každý konec spojení použity jiné protokoly. V našem případě jeden konec používá protokol X.21 a druhý konec používá protokol V.35. Tento fakt neplatí jen pro sériové linky, ale i pro lokální sítě. U lokálních sítí se ale spíše setkáváme s komplikovanějším případem, kdy mezi oba konce spojení je vložen např. přepínač (*Switch*), který konvertuje lin-

kové rámce jednoho linkového protokolu na rámce jiného linkového protokolu (např. Ethernet na FDDI), což má pochopitelně za následek i použití jiných protokolů na fyzické vrstvě.

Obr. 1.5 Komunikace na linkové vrstvě



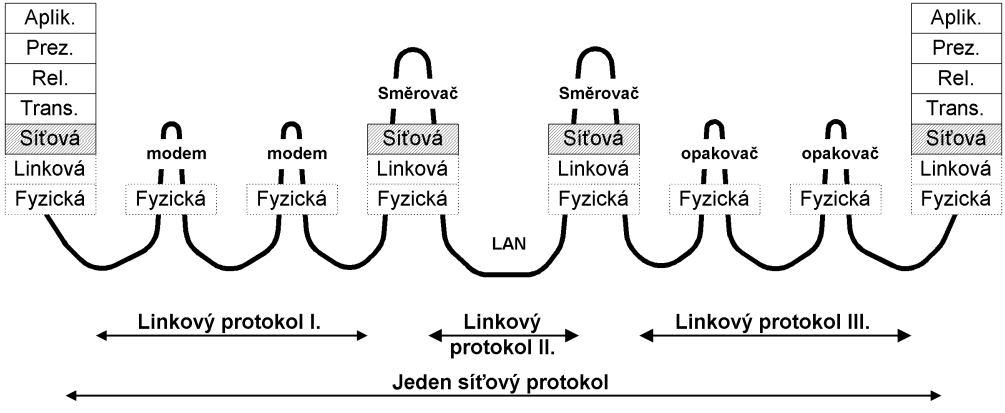
1.1.3 Síťová vrstva

Síťová vrstva zabezpečuje přenos dat mezi vzdálenými počítači WAN. Základní jednotkou přenosu je síťový paket, který se balí do datového rámce. Síťový paket se také skládá ze záhlaví a datového pole. Se zápatím se u síťových protokolů setkáváme jen zřídka.

Obr. 1.6 Síťový paket a jeho vkládání (encapsulation) do linkového rámce

Z obr. 1.6 je patrné, že síťové záhlaví společně s daty síťového paketu tvoří data linkového rámce.

V rozsáhlých sítích (WAN) mezi počítači leží zpravidla jeden nebo více směrovačů. Mezi sousedními směrovači je na linkové vrstvě vždy přímé spojení. Směrovač vybalí síťový paket z datového rámce (jednoho linkového protokolu) a před odesláním do jiné linky jej opět zabalí do jiného datového rámce (obecně jiného linkového protokolu).



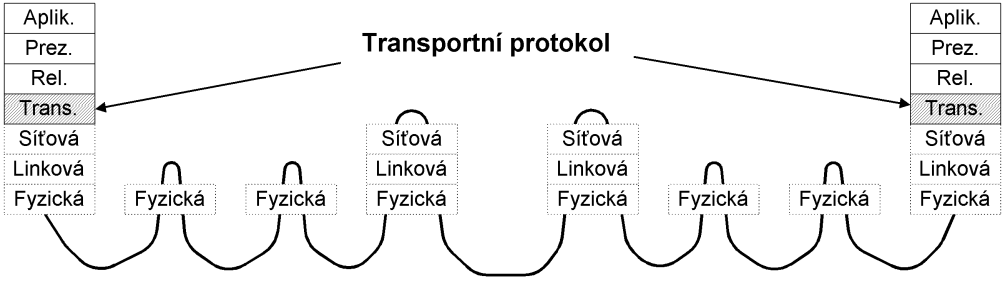
Obr. 1.7 Komunikace na síťové vrstvě

Síťovou vrstvu příliš nezajímá jaké jednotlivé linkové protokoly byly na cestě mezi oběma konci spojení použity.

Na síťové vrstvě je jednoznačně v celé WAN adresováno síťové rozhraní. Síťovým rozhraním může být např. karta pro Ethernet.

1.1.4 Transportní vrstva

Síťová vrstva zabezpečí spojení mezi vzdálenými počítači, takže transportní vrstvě se jeví jakoby žádné modemy, opakovače, mosty či směrovače na cestě nebyly. Transportní vrstva se zcela spoléhá na služby nižších vrstev. Také předpokládá, že spojení mezi počítači je zajištěno, proto se bez zbytečných starostí může věnovat spojení mezi aplikacemi na vzdálených počítačích.

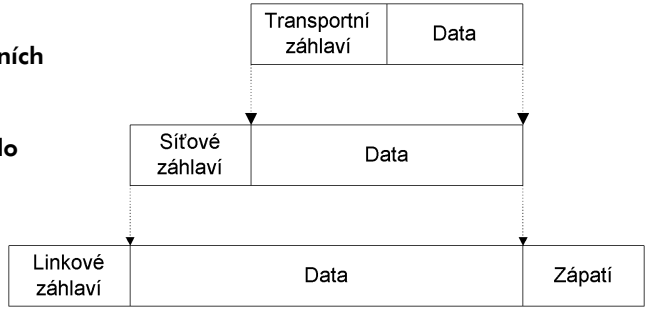


Obr. 1.8 Spojení na transportní vrstvě

Mezi dvěma počítači může být několik transportních spojení současně, jedno např. pro virtuální terminál a druhé pro elektronickou poštu. Z hlediska síťové vrstvy jsou pakety adresovány adresou počítače (resp. jeho síťového rozhraní). Z hlediska transportní vrstvy jsou adresovány jednotlivé aplikace. Aplikace jsou jednoznačně adresovány v rámci jednoho počítače.

Jednotkou přenosu je transportní paket, který se opět skládá ze záhlaví a datové části. Transportní paket se přenáší v datové části síťového paketu.

Obr. 1.9
Vkládání transportních paketů do síťových paketů, které jsou následně vloženy do linkových rámců



1.1.5 Relační vrstva

Relační vrstva zabezpečuje výměnu dat mezi aplikacemi, tj. provádí tzv. checkpoint, synchronizaci transakcí (*commit*), korektní uzavírání souborů atd. Dobře představitelnou relací je např. sdílení síťového disku. Disk může být sdílen po určitou dobu, avšak pracuje se s ním jen zřídka. Vždy, když je např. třeba pracovat se souborem na síťovém disku, tak se naváže na dobu od otevření souboru až po jeho uzavření spojení na transportní vrstvě. Avšak relace na relační vrstvě existuje po celou dobu sdílení disku.

Základní jednotkou je relační paket, který se opět vkládá do transportního paketu. V literatuře se můžeme často sekat s obrázkem, jak se relační paket skládá z relačního záhlaví a relačních dat a celý relační paket se vkládá do transportního paketu. Od transportní vrstvy výše tomu tak být nemusí. Informace relační vrstvy mohou být přenášeny uvnitř dat. Ještě markantnější je tato situace u prezentační vrstvy, která data např. zašifruje, takže změní celý obsah paketu.

1.1.6 Prezentační vrstva

Prezentační vrstva je zodpovědná za reprezentaci a zabezpečení dat. Reprezentace dat může být na různých počítačích různá. Např. se jedná o problém zdali je nejvyšší bit v bajtu zcela vlevo nebo vpravo atp. Zabezpečením se rozumí šifrování, zabezpečení integrity dat, digitální podepisování atd.

1.1.7 Aplikační vrstva

Aplikační vrstva předepisuje v jakém formátu a jak mají být data přebírána/předávána od aplikačních programů. Např. protokol Virtuální terminál popisuje jak mají být data formátována, ale i dialog mezi oběma konci spojení.

Obr. 1.10 Některé protokoly z rodiny protokolů ISO OSI

Aplikační	X.400, FTAM, CMIP
Prezentační	X.226, X.216, ASN.1
Relační	X.225, X.215
Transportní	TP 0-4, TP nespoj.
Síťová	X.25, X.75, ISDN
Linková	HDLC, LAPB, ISDN
Fyzická	V.24, V.35, X.21, ISDN

1.2 TCP/IP

Rodina protokolů TCP/IP se nezabývá (až na výjimky) fyzickou a linkovou vrstvou. V praxi se i v Internetu používají pro fyzickou a linkovou vrstvu často protokoly vyhovující normám ISO OSI, které standardizoval ITU.

Jaký je vztah mezi protokoly ISO OSI a TCP/IP? Každá skupina má vlastní definici svých vrstev i protokolů jednotlivých vrstev. Proto jsou protokoly ISO OSI a TCP/IP obecně nesouměřitelné. V praxi však je třeba využívat komunikační zařízení vyhovující ISO OSI pro přenos IP-paketů nebo např. naopak realizovat služby podle ISO OSI přes Internet.

1.2.1 Internet Protokol

Internet Protokol (dále jen IP-protokol) prakticky odpovídá síťové vrstvě. IP-protokol přenáší tzv. IP-datagramy mezi vzdálenými počítači. Každý IP-datagram ve svém záhlaví nese adresu příjemce, což je úplná směrovací informace pro dopravu IP-datagramu k adresátovi. Takže síť může přenášet každý IP-datagram samostatně. IP-datagramy tak mohou k adresátovi dorazit v jiném pořadí než byly odeslány.

Každé síťové rozhraní v rozsáhlé síti Internet má svou celosvětově jednoznačnou IP-adresu (jedno síťové rozhraní může mít více IP-adres, avšak jednu IP-adresu nesmí používat více síťových rozhraní). Internet je tvořen jednotlivými sítěmi, které jsou propojeny pomocí směrovačů. Směrovač se anglicky nazývá *router*, ve starších publikacích se však označuje jako *gateway*.

IP-protokolu jsou věnovány kapitoly 5, 6 a 7.

1.2.2 Protokoly TCP a UDP

Protokoly TCP a UDP odpovídají transportní vrstvě. Protokol TCP dopravuje data pomocí TCP segmentů, které jsou adresovány jednotlivým aplikacím. Protokol UDP dopravuje data pomocí tzv. UDP datagramů.

Protokoly TCP a UDP zajišťují spojení mezi aplikacemi běžícími na vzdálených počítačích. Protokoly TCP a UDP mohou zajišťovat i komunikaci mezi procesy běžícími na témže počítači, to je však z našeho pohledu nepříliš zajímavé.

Rozdíl mezi protokoly TCP a UDP spočívá v tom, že protokol TCP je tzv. spojovanou službou, tj. příjemce potvrzuje přijímaná data. V případě ztráty dat (ztráty TCP segmentu) si příjemce vyžádá zopakování přenosu. Protokol UDP přenáší data pomocí datagramů (obdoba telegramu), tj. odesílatel odešle datagram a už se nezajímá o to, zdali byl doručen.

Adresou je tzv. port. Pro pochopení rozdílu mezi IP-adresou a portem se používá srovnání s poštovní adresou. IP-adresa odpovídá adrese domu a port jménu a příjmení osoby, které má být dopis doručen.

Protokolu TCP se věnuje kapitola 9 a protokolu UDP se věnuje kapitola 10.

1.2.3 Aplikační protokoly

Aplikační protokoly odpovídají několika vrstvám ISO OSI. Relační, prezentační a aplikační vrstva ISO OSI je zredukována do jedné aplikační vrstvy TCP/IP.

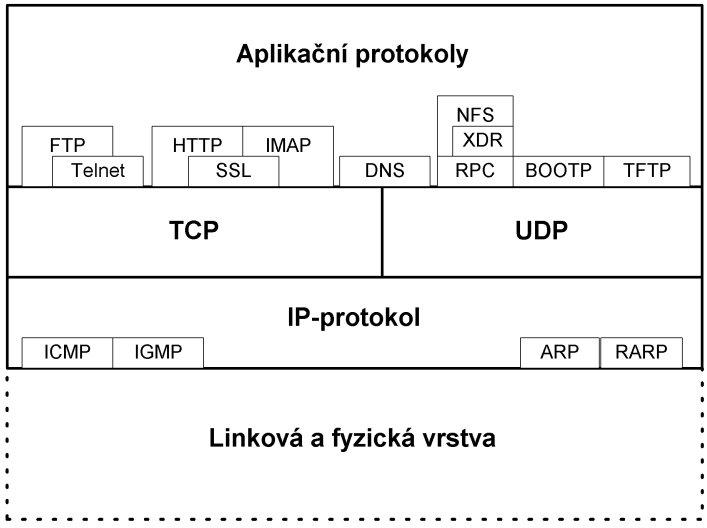
Absence prezentační vrstvy se řeší zavedením specializovaných „prezentačních-aplikačních“ protokolů, jako jsou protokoly SSL a S/MIME specializující se na zabezpečení dat. Nebo protokoly Virtuální terminál a ASN.1 určené pro prezentaci dat. Protokol Virtuální terminál (nezaměňovat se stejnojmenným protokolem v ISO OSI) specifikuje prezentaci dat v síti pro protokol Telnet, avšak využívají jej i další protokoly (FTP, SMTP a částečně i HTTP). Obdobně protokol ASN.1 (včetně kódování BER, resp. DER) byl nejprve využíván protokolem SNMP, avšak je využíván např. i protokolem S/MIME.

Aplikačních protokolů je velké množství. Z praktického hlediska je lze rozdělit na:

- Uživatelské protokoly, které využívají uživatelské aplikace (např. pro vyhledávání informací v Internetu). Příkladem takových protokolů jsou protokoly: HTTP, SMTP, Telnet, FTP, IMAP, POP3 atd.
- Služební protokoly, tj. protokoly se kterými se běžní uživatelé Internetu nesetkají. Tyto protokoly slouží pro správnou funkci Internetu. Jedná se např. o směrovací protokoly, které používají směrovače mezi sebou, aby si správně nastavily směrovací tabulky. Dalším příkladem je protokol SNMP, který slouží ke správě sítí.

V této publikaci se blíže věnujeme pouze jedinému aplikačnímu protokolu – protokolu DNS. Protokol DNS je zvláštní v tom, že jej nelze snadno zařadit ani do jedné z uvedených kategorií. Běžný uživatel jej pravděpodobně zařadí mezi služební protokoly. Tento názor mu však vydrží pouze po dobu, kdy uživatelův počítač pracuje správně. Jakmile uživatel začne mít potíže s protokolem DNS, pak obratem zjistí, že jej velice nutně ke své práci potřebuje, že jej používá v téměř každém svém příkazu.

Obr. 1.11
Některé protokoly z rodiny protokolů TCP/IP



1.3 Způsoby přenosu informací

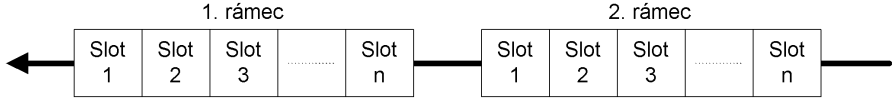
Síťových protokolů je velké množství, dokonce na jedné vrstvě máme často k dispozici několik protokolů. Zejména u protokolů nižších vrstev rozlišujeme jaký typ přenosu protokol zabezpečuje a zdali zabezpečuje službu spojovanou nebo nespojovanou, zdali protokol používá virtuální okruhy atd.

Rozeznáváme přenos synchronní, paketový a asynchronní.

1.3.1 Synchronní přenos

Synchronní přenos je vyžadován např. pro zvuk a video, tj. v případě, kdy je třeba stejnoměrně po dobu přenosu zajistit požadovanou šíři pásma. Stane-li se, že odesílatel nevyužije zajištěné pásmo, pak pásmo zůstává nevyužito.

Obr. 1.12
Rozdělení rámců na sloty u synchronního přenosu

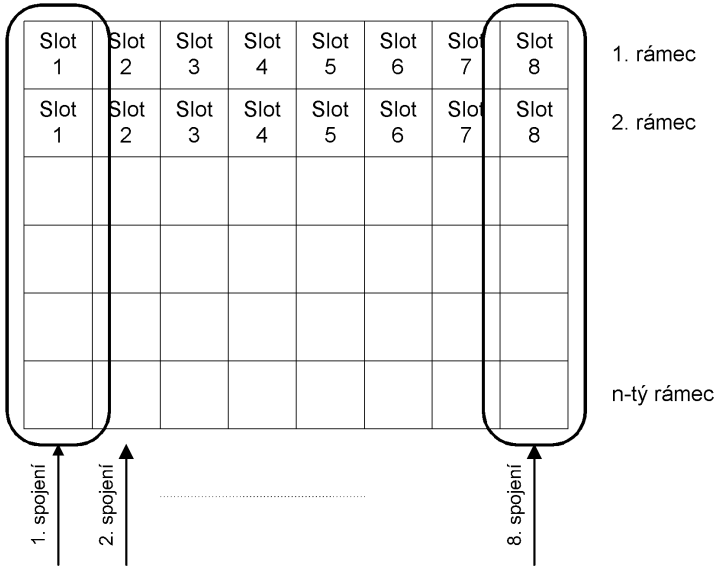


Synchronní přenos používá rámce konstantní délky, které jsou přenášeny sítí konstantní rychlostí.

Garance šíře přenosového pásma se u synchronního přenosu provádí rozdělením přenášených rámců na sloty. Pro dané spojení se pak v každém přenášeném rámci vyhradí jeden (či více) slotů, viz obr. 1.12. Představíme-li si, že v každém rámci je např. slot číslo 1 vyhrazen pro naše spojení, pak jelikož rámce stejnoměrně plynou sítí za sebou, tak naše aplikace má garantovánu šíři pásma, která je dána tím, kolik slotů číslo jedna přenese síť za vteřinu.

Podstatu pochopíme, když si několik rámců nakreslíme pod sebe do tzv. super-rámce, viz obr. 1.13. Sloty pod sebou patří témuž spojení.

Obr. 1.13
Super-rámec



Se synchronním přenosem se setkáváme např. u připojení podnikové telefonní ústředny k ústředně Telecomu. Ta bývá připojena např. linkou E1, která obsahuje 32 slotů, každý o šířce pásma 64 kb/s. Slot lze využít pro telefonní hovor. Současně je tak teoreticky garantováno 32 hovorů (některé sloty se však používají jako služební).

Internet nepoužívá synchronní přenos, tj. negarantuje šíři přenášeného pásma. Kvalitní přenos zvuku či videa se v Internetu zpravidla dociluje předimenzováním přenosových linek.

1.3.2 Paketový přenos

Paketový přenos je výhodný zejména pro přenos dat. Pakety nesou data obecně různé délky.

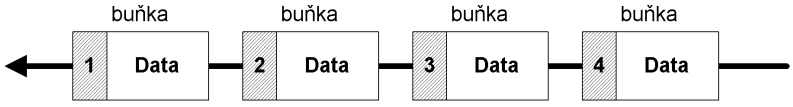
Obr. 1.14
Paketový přenos dat

Paket nese data vždy jedné aplikace (jednoho spojení). Jelikož jsou pakety různé délky, nelze garantovat šíři pásma. Výhodou je efektivní využití pásma, protože v případě, že aplikace nepotřebuje přenášet data, pak pásmo mohou využít jiné aplikace.

1.3.3 Asynchronní přenos

Asynchronní přenos používá protokol ATM. Tento typ přenosu kombinuje paketový přenos se synchronním přenosem.

Obr. 1.15
Asynchronní přenos dat

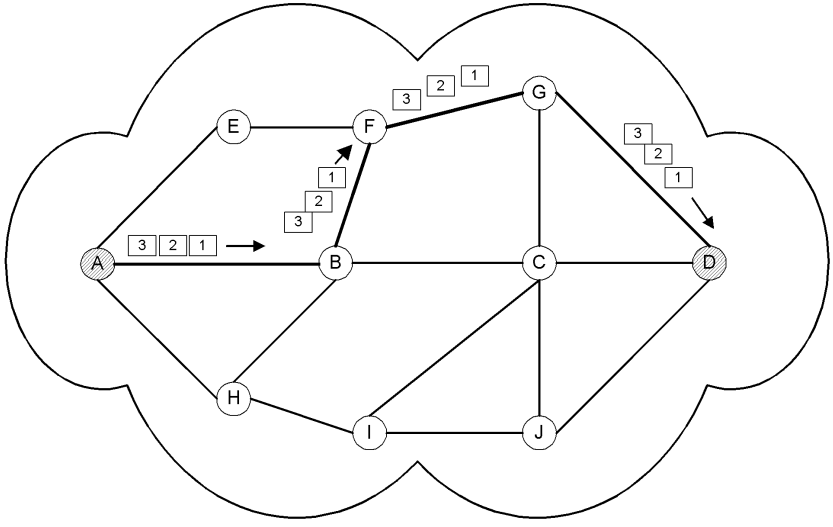


Podobně jako u paketového přenosu jsou u asynchronního přenosu data přenášena v malých paktech, které se však nazývají buňky. Obdobně jako u paketového přenosu se v jedné buňce přenáší data jedné aplikace (jednoho spojení). Avšak buňky mají stejnou délku, takže garantuje-li se, že každá x-tá buňka bude k dispozici konkrétní aplikaci (konkrétnímu spojení), pak se tím garantuje i šířka pásma. Navíc, pokud aplikace buňku neodešle – nevadí, může být odeslána buňka jiné aplikace.

1.4 Virtuální okruh

Některé síťové protokoly vytváří v síti virtuální okruh (*Virtual Circuit*). Virtuální okruh je vedený sítí a všechny pakety spojení pak prochází tímto okruhem. V případě, že se okruh někde přeruší, tak se spojení přeruší, vytvoří se nový okruh a poté se opět mohou přenášet data.

Obr. 1.16
Virtuální okruh



Na obr. 1.16 je vytvořen virtuální okruh mezi uzly A a D přes uzly B, F a G. Všechny pakety musí procházet tímto okruhem.

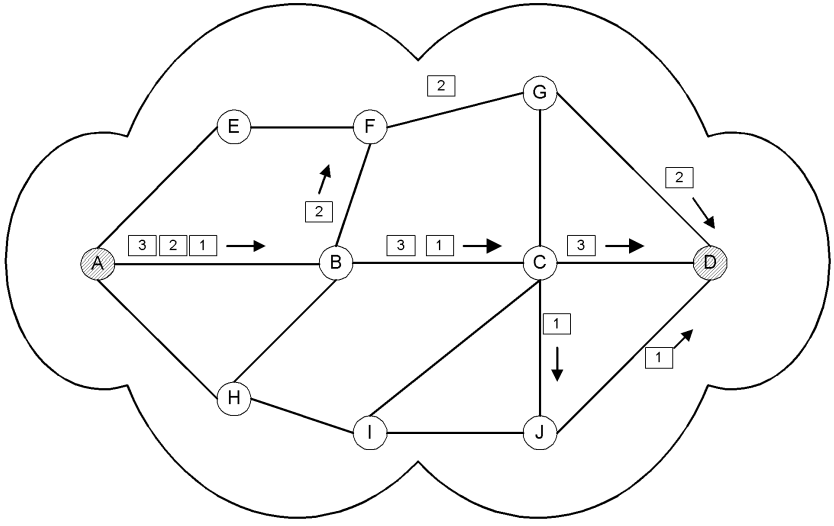
Na virtuálním okruhu je možné buď přenášet datagramy, kdy okruh negarantuje doručení datagramu příjemci (tj. v případě zahlcení sítě může datagram i zahodit), takovýmto protokolem je např. protokol Frame Relay. Nebo naopak virtuální okruh může navázat spojení a doručení dat garantovat, tj. přená-

šená data čísluje a příjemce potvrzuje příjem dat. Pokud by se nějaká data ztratila, pak je dožádáno jejich opakování. Tento mechanismus používá např. protokol X.25.

Výhodou virtuálního okruhu je, že je nejprve sestaven (pomocí signalizace) a teprve do sestaveného okruhu se vkládají data. Každý paket obecně nemusí ve svém záhlaví nést globálně jednoznačnou adresu příjemce, ale pouze identifikaci okruhu.

V Internetu se mechanismus virtuálních okruhů nepoužívá, protože zničení uzlu ve virtuálním okruhu znamená přerušení spojení, což nevyhovovalo tvůrcům rodiny protokolů TCP/IP, která byla prvotně vytvořena pro ministerstvo obrany USA. Z tohoto důvodu IP-protokol nepoužívá virtuální okruhy. Každý IP-datagram nese IP-adresu příjemce (tj. úplnou směrovací informaci) a je proto dopravován samostatně. Zničení uzlu sítě může zničit pouze IP-datagram právě procházející zničeným uzlem v okamžiku zničení uzlu. Další IP-datagramy jsou směrovány přes jiné uzly.

Obr. 1.17 IP-protokol nepoužívá virtuální okruhy



Z obr. 1.17 je vidět, že IP-datagramy 1, 2 a 3 odešly z uzlu A společně na uzel B, ale z tohoto uzlu jsou již datagramy 1 a 3 směrovány jinou cestou než datagram 2. Do cílového uzlu D pak každý z našich datagramů dorazí jinou cestou. Obecně IP-datagramy mohou dorazit i v jiném pořadí než byly odeslány, tj. např. v pořadí 2, 1 a 3.

Nad nespojovaným IP-protokolem se v Internetu používá protokol TCP (protokol vyšší vrstvy), který spojení naváže a který garantuje doručení dat. Tj. pokud zjistí, že se některá data ztratila, vyžádá si jejich opakování. Pokud byla data ztracena díky zničení některého uzlu sítě a v síti existuje ještě jiná cesta, pak opakování dat již automaticky proběhne po této záložní cestě.

Abychom byli objektivní, tak na obranu protokolu X.25 musíme uvést, že to je protokol popisující pouze rozhraní mezi uživatelem a sítí X.25. Uvnitř sítě X.25 (kam uživatel nevidí), mohou být pak data přenášena obdobně jako v Internetu (tj. nikoliv protokolem X.25 samotným).

1.4.1 Pevné a komutované virtuální okruhy

Virtuální okruhy rozeznáváme:

- Pevné (*Permanent Virtual Circuit – PVC*), tj. virtuální okruhy pevně sestavené administrátorem sítě.
- Komutované (*Switched Virtual Circuit – SVC*), tj. virtuální okruhy dynamicky vznikající podle okamžité potřeby. SVC se vytváří pomocí tzv. signalizačních protokolů, což jsou protokoly pomocí kterých spolu mohou komunikovat uživatel a samotná síť. Síť signalizuje uživateli různé mimořádné stavy, pomocí kterých lze spravovat i monitorovat síť, ale právě i vytvářet okruhy. Na SVC se tak komunikace skládá ze dvou kroků: z vytvoření virtuálního okruhu a z jeho vlastního využití ke komunikaci.

PVC je obdobou pevných linek a SVC je obdobou komutovaných (vytáčených) linek v telefonní síti.

Poznámka: Protokoly využívající virtuální okruhy se anglicky nazývají *Connection Oriented Network Services* (CONS) a protokoly dopravující své pakety bez sestavení virtuálních okruhů se anglicky nazývají *Connection-Less Network Services* (CLNS), tj. spojované a nespojované síťové služby. V této publikaci spojovanou službou míníme službu, kdy je navázáno spojení, kterým jsou potvrzována přijatá data, v případě ztráty dat je pak dožadováno zopakování vysílání.

2

MS Network Monitor

MS Network Monitor slouží k monitorování dat v síti. Monitorováním se rozumí zachytávání linkových rámců v síti a jejich ukládání do paměti. Následně je pak možné prohlížet obsah jednotlivých rámců. MS Network Monitor má zobrazování rámců propracováno, tj. umí nejenom oddělit záhlaví jednotlivých paketů od dat, ale zejména rozpitvat jednotlivé položky záhlaví síťových protokolů.

Network Monitor používají nejčastěji správci sítí pro dohledání závady v komunikaci na síti nebo pro monitorování vytížení sítě. Avšak MS Network Monitor je také nepostradatelným pomocníkem pro programátory, kteří vyvíjejí síťové aplikace. Např. napíšete-li aplikaci klient/server, spustíte ji „a nic“ – klient se serverem ani nenaváží spojení. V takovém okamžiku nevíte, zdali je na vině klient nebo server. Pokud budete odchytávat datové rámce, pak např. zjistíte, že klient odeslal datový rámec, ale server na něj vůbec nezareagoval, takže jste zjistili, že závada bude pravděpodobně na serveru. Nebo jste si všimli, že data odeslaná klientem jsou jiná, než jste předpokládali…

My budeme využívat MS Network Monitor pro demonstrace při popisu jednotlivých síťových protokolů.

Obdobných programů jako je MS Network Monitor je celá řada. MS Network Monitor je přibalován „zdarma“ k distribuci některých produktů firmy Microsoft.

V operačních systémech UNIX je k dispozici obdobný program *tcpdump*. V komerční sféře však na operačních systémech UNIX běží servery, takže využití programu *tcpdump* je spíše pro správce serverů a běžní uživatelé nemají šanci jej použít. Navíc program *tcpdump* nemá na rozdíl od MS Network Monitoru tak propracované zobrazování informací, takže není ani pro běžné uživatele určen.

Existují i síťové monitory realizované hardwarově. Jaké jsou jejich výhody? Tyto monitory jsou nepostradatelné zejména pro technický personál. Softwarové monitory zobrazí pouze rámce, které jsou v pořádku. Může se např. stát, že stanice na síti má poškozenou síťovou kartu a generuje poškozené rámce. Takovou stanici lze těžko objevit pomocí softwarových monitorů. Rovněž služební rámce FDDI se zobrazovat nebudou.

Pokud potřebujete sledovat provoz na sériových linkách, kde nejsou připojeny žádné PC (např. spoj k poskytovateli FrameRelay), tak nelze rovněž použít softwarové monitory pracující na PC. Je však třeba poznamenat, že pro monitorování takových linek je možné použít software, jenž je součástí systému směrovačů a provést tzv. *dump* linky.

Na lokálních sítích mohou být zachycovány:

- Pouze rámce jež jsou adresovány stanici, na které běží network monitor (včetně oběžníků).
- Veškerý provoz segmentu lokální sítě. Pro tento případ musí být síťová karta přepnuta do tzv. promiskuitního módu, tj. módu způsobujícího, že síťová karta nebude akceptovat pouze rámce stanici adresované, ale všechny rámce. V některých operačních systémech je takovéto přepnutí síťové karty umožňováno pouze privilegovaným uživatelům. Navíc některé starší síťové karty (např. protokolu Token Ring) promiskuitní mód vůbec nepodporují.

Existuje řada mutací MS Network Monitoru (Pro Windows 2000, Windows NT, Windows 98 atd.). Na první pohled jsou všechny mutace velice podobné. Rozdíl bývá ve skutečnosti, zdali je možné zachycovat všechny linkové rámce na LAN, nebo jen linkové rámce adresované stanici, kde monitor běží. Dalším rozdílem je pak počet síťových protokolů, jejichž pakety umí monitor podrobně rozpitvat. Ve Windows 2000 Advanced Server je Network Monitor součástí základního softwaru a lze jej nainstalovat již při instalaci systému Windows 2000.

Instalaci MS Network Monitoru je třeba provádět velice pozorně a přesně podle průvodce instalací. Uprostřed instalace jste většinou vyzváni k přidání Network Monitor Agenta. Pokud neprovedete vše dle instrukcí, pak pravděpodobně bude MS Network Monitor nefunkční a instalaci bude nutné opakovat.

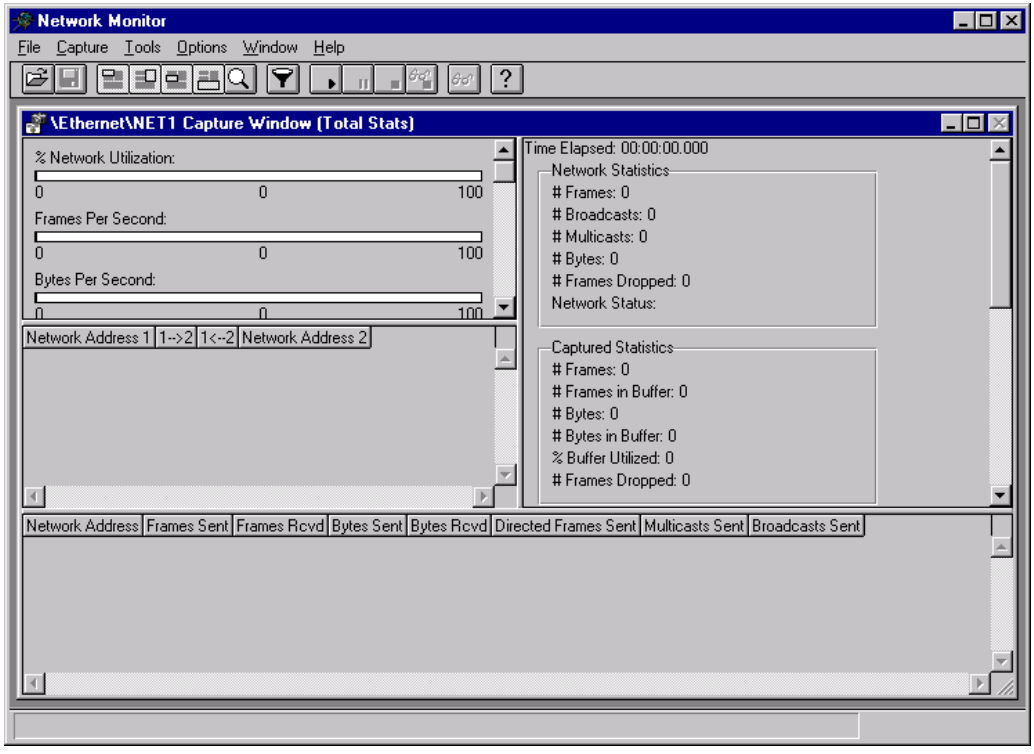
MS Network Monitor využívá pro zachytávání síťových rámců komponentu Windows, která se nazývá Network Monitor Agent. Network Monitor Agent se přidává ve Windows NT pomocí voleb: „Ovládací panely → Síť → Služby → Přidat“.

Největším problémem je otázka bezpečnosti. Argumentem proti nasazení network monitoru bývá skutečnost, že pomocí něj lze velice snadno zjistit hesla uživatelů lokální sítě používajících programy *telnet*, *ftp* i web-prohlížeče (v případě protokolu HTTP).

My naopak považujeme demonstraci odchycení hesel za užitečnou, protože firma po takovéto demonstraci (nikoliv po prvních bezpečnostních incidentech) přejde na jinou autentizaci než pomocí jména uživatele a nezabezpečeného hesla.

2.1 Zachytávání rámců

Po spuštění MS Network Monitoru se objeví okno znázorněné na obr. 2.1. Uvnitř okna je okno pro zachytávání rámců (*Capture Window*). Pokud se toto vnitřní okno neobjeví, pak je Network Monitor nebo Network Monitor Agent špatně nainstalován a je třeba instalaci opakovat.

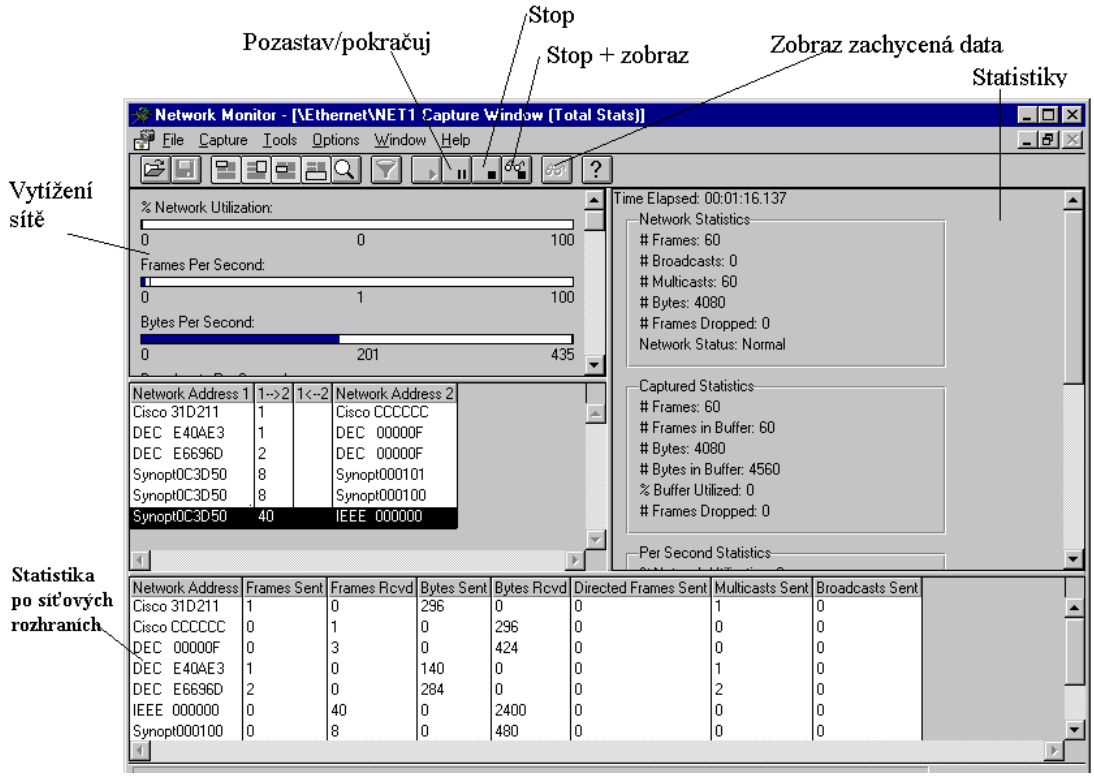


Obr. 2.1 Úvodní obrazovka MS Network Monitoru

Zachytávání rámců lze nastartovat buďto tlačítkem Start umístěným v pruhu nástrojů (viz. obr. 2.2) nebo volbou „Capture → Start“.

Obr. 2.2 Tlačítko start

Po startu zachytávání rámců se objeví okno zobrazené na obr. 2.3.



Obr. 2.3 Zachytávání datových rámců

Okno zachytávání datových rámců se skládá z několika rámů.

Vlevo nahoře jsou stupnice, které graficky vyjadřují:

- Zatížení sítě (*Network Utilization*).
- Počet rámců za vteřinu (*Frames Per Second*).
- Počet přenesených bajtů za vteřinu (*Bytes Per Second*).
- Počet všeobecných oběžníků za vteřinu (*Broadcast Per Second*).
- Počet adresných oběžníků za vteřinu (*Multicast Per Second*).

Vpravo nahoře je rám statistik. Na prvním řádku běží čas od startu zachytávání (resp. od vynulování čítačů volbou „Capture → Clear Statistics“). Jednotlivé oblasti statistik jsou pak uzavřeny do menších rámečků:

- Síťové statistiky (*Network Statistics*) zobrazující celkový počet rámců prošlých segmentem LAN (*Frames*), z toho všeobecných oběžníků (*Broadcast*) atd.
- Statistika zachycených rámců (*Captured Statistics*) udává statistiku jen těch rámců, které byly zachyceny. Případný rozdíl oproti síťové statistice spočívá v tom, že můžeme definovat volbou

„Capture → Clear Filter“ filtr, který umožní nezachycovat do paměti všechny rámce, ale jen ty, které si přejeme.

- Sekundové statistiky (*Per Seconds Statistics*), vyjadřující hodnotu za uplynulou vteřinu.
- Statistiky vztažené k lokální síťové kartě.

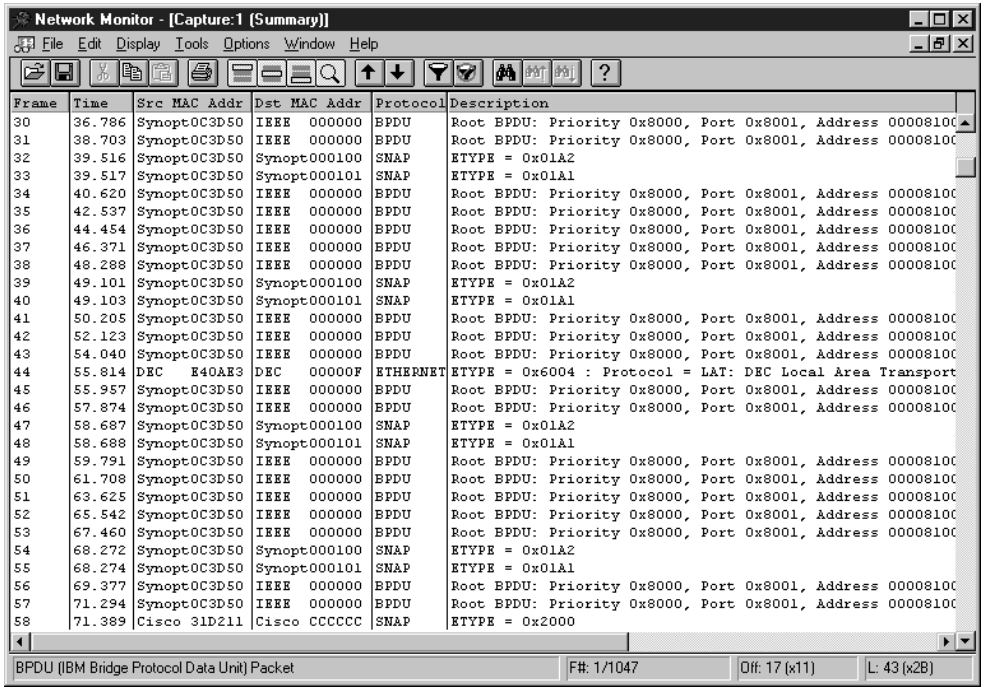
Ve spodní části jsou pak statistiky vztahující se k jednotlivých síťovým rozhraním nacházejícím se na lokální síti.

V pruhu nástrojů mj. lze pomocí tlačítka:

- „Pozastav/pokračuj“ pozastavit zachytávání rámců.
- „Stop“ ukončit zachytávání rámců. Po ukončení zachytávání lze začít s prohlížením zachycených rámců.
- „Zobraz zachycená data“ přejít k prohlížení zachycených rámců.
- „Stop + zobraz“ lze provést ukončení zachytávání a přejít k prohlížení zachycených rámců.

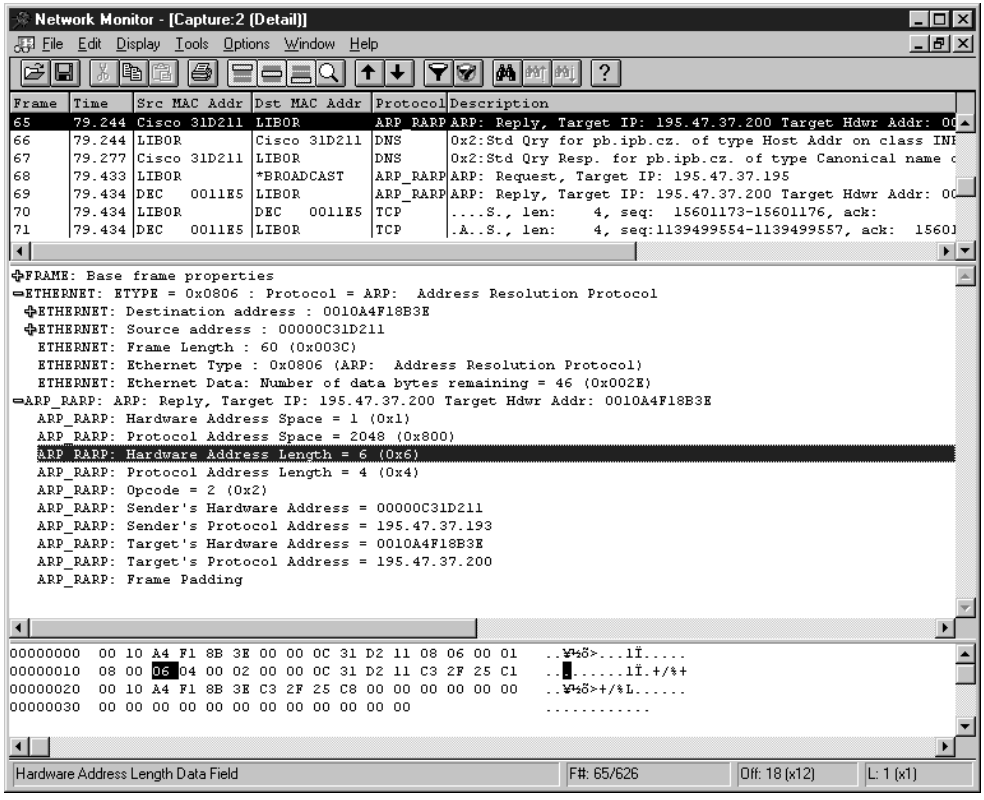
2.2 Prohlížení zachycených rámců

Přejdeme-li do režimu prohlížení zachycených rámců, pak se nám objeví obrazovka se všemi zachycenými rámci – viz obr. 2.4.



Obr. 2.4 Zachycené rámce

Na obrazovce se zachycenými rámci si můžeme nalézt požadovaný rámec. Klepnutím na nalezený rámec se objeví detailní zobrazení rámce – viz obr. 2.5.



Obr. 2.5 Detailní zobrazení rámce

Detailní zobrazení rámce se skládá ze tří rámů. V horním rámu se můžeme pohybovat po zachycených rámcích. V prostředním rámu je pak detailně rozebrán obsah rámce a ve spodním rámu je pak rámec vypsán hexadecimálně a znakově.

Prostřední rám je to, co nás nejvíce zajímá. Na obr. 2.5 je rozebráno linkové záhlaví, následované záhlavím síťového paketu. Případně pak pokračuje transportní záhlaví atd. až po aplikační vrstvu. Na aplikační vrstvě je pro některé protokoly jako je např. DNS rozebrán paket zcela.

Před některými položkami záhlaví je znak plus vyznačující, že je možné po klepnutí na něj získat detailnější rozpitvání položky. Dostaneme se tak k jednotlivým položkám záhlaví všech protokolů popisovaných v této publikaci.

Rozpitvané rámce lze také vypsat na tiskárnu nebo do souboru pomocí volby „File → Print“. Rámec z obr. 2.5 jsme pak vytiskli a dostali jsme následující výpis:

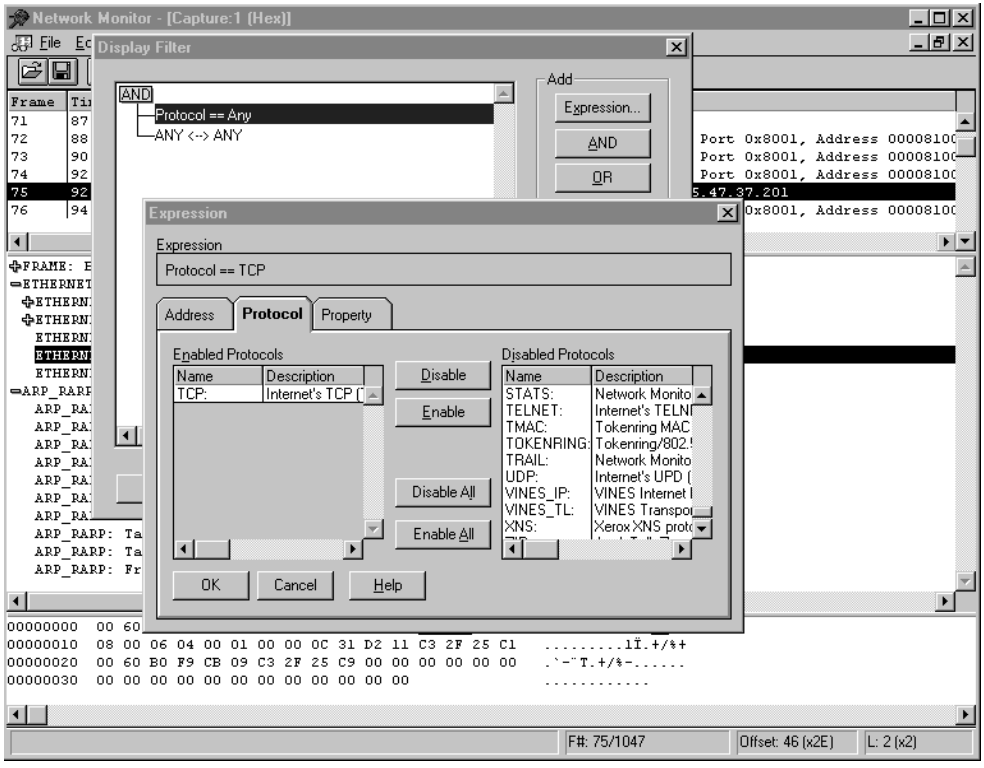
```
…
Target Hdwr Addr: 0010A4F18B

+ FRAME: Base frame properties
  ETHERNET: ETYPE = 0x0806 : Protocol = ARP:  Address Resolution Protocol
    + ETHERNET: Destination address : 0010A4F18B3E
    + ETHERNET: Source address : 00000C31D211
      ETHERNET: Frame Length : 60 (0x003C)
      ETHERNET: Ethernet Type : 0x0806 (ARP:  Address Resolution Protocol)
      ETHERNET: Ethernet Data: Number of data bytes remaining = 46 (0x002E)
  ARP_RARP: ARP: Reply, Target IP: 195.47.37.200 Target Hdwr Addr: 0010A4F18B3E
      ARP_RARP: Hardware Address Space = 1 (0x1)
      ARP_RARP: Protocol Address Space = 2048 (0x800)
      ARP_RARP: Hardware Address Length = 6 (0x6)
      ARP_RARP: Protocol Address Length = 4 (0x4)
      ARP_RARP: Opcode = 2 (0x2)
      ARP_RARP: Sender's Hardware Address = 00000C31D211
      ARP_RARP: Sender's Protocol Address = 195.47.37.193
      ARP_RARP: Target's Hardware Address = 0010A4F18B3E
      ARP_RARP: Target's Protocol Address = 195.47.37.200
      ARP_RARP: Frame Padding

00000:  00 10 A4 F1 8B 3E 00 00 0C 31 D2 11 08 06 00 01   .....>...1......
00010:  08 00 06 04 00 02 00 00 0C 31 D2 11 C3 2F 25 C1   .........1.../%.
00020:  00 10 A4 F1 8B 3E C3 2F 25 C8 00 00 00 00 00 00   .....>./%.......
00030:  00 00 00 00 00 00 00 00 00 00 00 00               ............
```

Jedná se o ethernetový rámec nesoucí paket protokolu ARP.

2.3 Filtry pro zobrazení zachycených rámců

Zásadním problémem při použití Network Monitoru je nalézt požadovaný rámec ve zpravidla velkém množství zachycených rámců. Pro usnadnění tohoto hledání slouží filtry. Filtry jsou dvojí:

- Filtry při zachytávání rámců, které se aktivují před startem zachytávání.
- Filtry pro zobrazení zachycených rámců, které se aktivují při prohlížení rámců. Tyto filtry umožňují zobrazit jen některé ze zachycených rámců.



Obr. 2.6 Nastavování filtru

Filtry pro zobrazování již zachycených rámců se nastavují volbou „Display → Filter“ v režimu prohlížení zachycených rámců. Volbou „Capture → Filter“ se aktivují filtry pro zachytávání rámců, jejichž dialogová okna jsou analogická.

Filtr se skládá z logických podmínek spojených logickými operátory AND, OR a NOT. Podmínka se může týkat:

- Adresy (např. IP-adresy v případě IP-protokolu).
- Protokolu, tj. budou zobrazeny jen rámce nesoucí vyjmenované protokoly (např. IP-protokol, či protokol TELNET). Tato možnost je znázorněna na obr. 2.6.
- Hodnoty nějaké položky konkrétního protokolu, např. že TCP port odesílatele je 1345.

2.4 Domácí cvičení

Nyní jste již připraveni pro první experimenty s MS Network Monitorem. Jako první cvičení doporučujeme odchycení vlastního hesla (v žádném případě nedoporučujeme odchytávání cizího hesla – to ani nejde, pokud nemůžete přepnout síťovou kartu do promiskuitního módu). Doporučujeme procvičit tři následující příklady:

1. Protokol Telnet. Nemusíte mít ani konto na žádném serveru. Zvolte si libovolný server, kde běží server pro Telnet. Spusťte MS Network Monitor a pomocí aplikace TELNET se pokuste navázat spojení, zvolte libovolného uživatele a heslo. Server spojení sice odmítne, ale vy zastavíte MS Network Monitor a pomocí prohledávání rámců vámi zadané jméno a heslo najděte. Je třeba mít jistou trpělivost, protože terminál se nejprve snaží se serverem pomocí příkazů <IAC> aplikačního protokolu Telnet nastavit vlastnosti terminálu. Tento dialog je třeba přeskočit.
2. Protokol FTP. Zde je situace jednodušší, protože zde pravděpodobně nebude dialog příkazy <IAC>.
3. Protokol HTTP se základní autentizací (nesmí se jednat o protokol HTTPS – tam heslo odchytit nelze, protože je zabezpečeno). Zde je jméno a heslo součástí každého dotazu, pokud se nejedná o přístup na anonymní server. Avšak protože hlavičky protokolu HTTP nesmí obsahovat znaky, které nejsou ASCII, tak jméno uživatele a heslo jsou odděleny dvojtečkou a celé je to kódováno Base64. Je třeba provést dekódování buď ručně, nebo nějakým programem. Hlavička, která nese tyto informace je

   ```
   Authorization: Base64(uživatel:heslo)
   ```

   nebo v případě autentizace vůči firewallu (proxy) se jedná o hlavičku:

   ```
   Proxy-Authorization: Base64(uživatel:heslo)
   ```

   Kde Base64() vyjadřuje, že argument je Base64 kódován do sedmibitového tvaru.

3

Fyzická vrstva

Pro drtivou většinu uživatelů jsou protokoly na fyzické vrstvě „ty naprosto odtažité protokoly, které popisují signály na konektorech (uživatelé říkají zástrčkách) na zadní straně počítače, na které je připojena šňůra propojující počítač s počítačovou sítí“. Uživatelé starosti o fyzickou vrstvu přenechávají technikům, o kterých se domnívají, že to jsou lidé, „kteří natahují dráty a případě na nich něco měří nějakým voltmetrem nebo čím“. Situace je dnes úplně jiná, technik je spíše správcem programového vybavení, které ovládá všechny „tajuplné krabičky v zamčených místnostech“. Tato představa se v podstatě netýká pouze fyzické vrstvy, ale i linkové vrstvy. Uživatelé se většinou zabývají až IP-protokolem (resp. síťovými protokoly). V IP-protokolu už totiž „vidí“ nebo „nevidí“ servery či sousedy. Kdežto fyzická a linková vrstva zajišťuje pouze komunikaci s nějakou krabičkou na půl cesty k serveru, o které běžný uživatel ani neví, že tam je.

V zásadě rozlišujeme dva typy počítačových sítí: lokální sítě (LAN) a rozsáhlé sítě (WAN). Z hlediska fyzické vrstvy jsou v podstatě protokoly pro LAN jednou skupinou protokolů a protokoly pro WAN druhou skupinou. Kromě toho dnes populární protokol ATM smazávající rozdíl mezi LAN a WAN používá nejen nové protokoly, ale je zejména schopen využít stávající linky pro WAN včetně jejich protokolů (např. linky E1). Na druhou stranu ATM i emuluje protokoly pro LAN.

WAN

Rozsáhlé sítě pokrývají velkou škálu situací. Od připojení domácího PC k Internetu pomocí sériové asynchronní linky rychlostmi uváděnými v kb/s až po mezikontinentální linky realizované podmořskými kabely či družicovými spoji o rychlostech uváděných v Gb/s.

LAN

Lokální sítě jsou středně rychlé sítě. Základní vlastností LAN je, že na lokální síti spolu zpravidla komunikuje několik stanic na sdíleném médiu. Na LAN je běžné použití oběžníků. V rámci jedné LAN se používá stejný linkový protokol (např. Ethernet). Dnes se však pod pojmem LAN často myslí tzv. rozšířené LAN, které mohou obsahovat mosty a přepínače, které mají síťová rozhraní pro více linkových protokolů a umí konvertovat rámce jednoho linkového protokolu na rámce jiného linkového protokolu. Z hlediska fyzické vrstvy nás však budou zajímat pouze klasické LAN, protože na rozšířené LAN se fyzická vrstva dívá jako na soustavu jednotlivých LAN.

Pro připojení LAN k rozsáhlé síti (WAN) se využívají směrovače. Směrovač je zařízení předávající IP-datagramy z jednoho síťového rozhraní na jiné své síťové rozhraní, přitom každé rozhraní může být na jiné LAN, nebo může být rozhraním do WAN.

Přenosové rychlosti na dnešních LAN se pohybují od 10 Mb/s až po Gb/s.

3.1 Sériové linky

PC má zpravidla na zadní straně konektory pro sériová rozhraní COM1 a COM2. COM1 bývá někdy použit pro myš, takže pro připojení sériové linky k PC zbývá rozhraní COM2. Na sériové rozhraní se zpravidla připojuje modem.

Sériové výstupy PC používají signály specifikované normou ITU V.24 (v USA analogická norma RS232). Jedná se o rozhraní pro sériový asynchronní arytmický přenos dat. V praxi se běžně používá do 64 kb/s, ale modem si doma na něj nejspíše připojíte rychlostí 115 200 b/s a ono to kupodivu bude také pracovat.

3.1.1 Sériový a paralelní přenos dat

Sériový přenos znamená, že pro přenos informace od odesílatele k příjemci je jen jedna dvojice vodičů (resp. u asymetrických rozhraní jeden vodič a společná zem), takže jednotlivé bity každého znaku se přenáší postupně za sebou – tj. sériově.

Na rozdíl od paralelního přenosu, který k přenosu používá osm vodičů (nebo násobek osmi), tj. všechny bity přenášeného znaku se mohou přenést najednou – tj. paralelně. Paralelní přenos se používá zejména uvnitř počítače na jeho sběrnicích, ale i třeba pro komunikaci s paralelní tiskárnou. Existují i modemy využívající paralelní rozhraní.

3.1.2 Symetrický a asymetrický signál

U sériových rozhraní bývají alespoň dva signály: příjem dat a vysílání dat. Pokud každý signál je realizován dvěma vodiči, pak hovoříme o symetrickém nebo diferenciálním signálu. Symetrické signály pro přenos dat používá např. rozhraní V.35 a X.21.

Pokud je každý signál realizován jedním vodičem proti společné zemi, pak hovoříme o asymetrickém signálu. Asymetrické signály používá např. rozhraní V.24, ale také se např. používá pro řídící signály (nikoliv datové) rozhraní V.35.

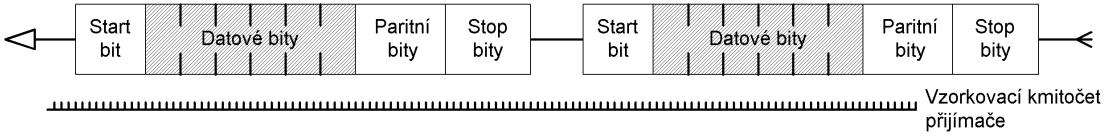
3.1.3 Synchronní nebo asynchronní přenos

Chcete-li se s někým např. telefonem o něčem domluvit, pak musíte mluvit tak rychle, aby on byl schopen vám rozumět. Např. budete-li mluvit desetkrát rychleji, pak vám stěží porozumí. Tj. ten kdo poslouchá se musí synchronizovat s tím kdo mluví.

Z hlediska synchronizace rozeznáváme přenos:

- Synchronní, kdy se informace přenášejí po jednotlivých bitech. Okamžiky přechodu od přenosu jednoho přenášeného bitu k přenosu dalšího bitu jsou vždy stejně vzdáleny.
- Asynchronní, kdy okamžiky přechodu od přenosu jednoho bitu k přenosu dalšího bitu nejsou stejně vzdáleny. Zvláštním případem asynchronního přenosu dat je tzv. arytmický přenos. U arytmického přenosu je přenos znaků asynchronní, ale jednotlivé bity v přenášeném znaku se přenáší synchronně. Pokud se hovoří o asynchronním přenosu, pak se má většinou na mysli asynchronní arytmický přenos.

Při asynchronním arytmickém přenosu je odesílaný znak obalen obálkou tvořenou startovacím bitem, paritními bity a stop bity (viz obr. 3.1).



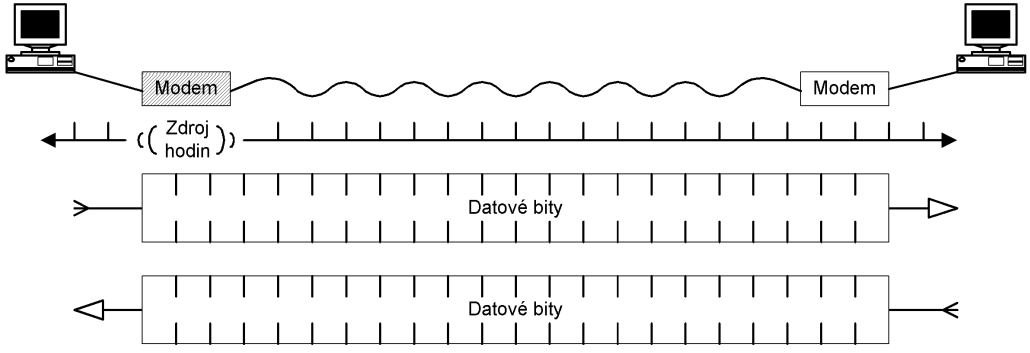
Obr. 3.1 Asynchronní arytmický přenos znaků

Přijímač generuje vzorkovací kmitočet o řád vyšší frekvence než je maximální možná frekvence přenosu jednoho bitu. Přijímač touto frekvencí testuje vzorky přijímaného signálu. Pokud vzorek odpovídá s jistou pravděpodobností startovacímu bitu, předpokládá že narazil na přenášený znak. Pokračuje ve vzorkování, vše až do stop bitů považuje za bity přenášeného znaku. Mezi start bitem a stop bity jsou datové bity přenášeného znaku, navíc tam může být ještě paritní bit zabezpečující jednoduchý kontrolní součet přenášeného znaku.

Asynchronní přenos má výhodu v tom, že přijímač se může se značnou tolerancí přizpůsobit frekvenci vysílače. Avšak obálka se zpravidla skládá z jednoho start bitu, jednoho paritního bitu a jednoho, jeden a půl či dvou stop bitů. Což znamená, že obálka svou režií může způsobit i 50% nárůst požadavků na přenos (přenášený znak má nejčastěji 5 až 8 bitů).

U synchronního přenosu se režie snižuje. V minulosti se používal blokově synchronní přenos dat (BSC), kdy se data přenášela v blocích. Na začátku bloku pak byl jeden nebo více synchronizačních znaků, které byly obdobou start bitu. Přijímač se synchronizoval na těchto synchronizačních znacích. Blok se pak přenáší synchronně.

Dnes je však běžnější zcela jiný princip. Kromě přenášených dat se přenáší ještě synchronizační signál (hodiny). Na obr. 3.2 se na komunikaci podílí čtyři zařízení (dva modemy a dva počítače).



Obr. 3.2 Synchronní přenos

Podobně jako v orchestru může být jen jeden dirigent, tak zdrojem hodin může být jen jedno z těchto čtyř zařízení. Zpravidla to bývá jeden z modemů (*originator*). Ostatní zařízení si přizpůsobí takt svých obvodů tomuto dirigentovi. Jelikož všechna čtyři zařízení jsou synchronizována, tak mohou mezi sebou

přímo komunikovat (bez vzorkování). Pokud by byl zdrojem hodin jeden z počítačů, pak v komunikaci mezi modemy nastavíme jako *originator* modem na straně počítače generujícího hodiny, tento modem však nastavíme do režimu, že používá externí hodiny (z počítače).

3.1.4 Normy V.24, V.35 a X.21

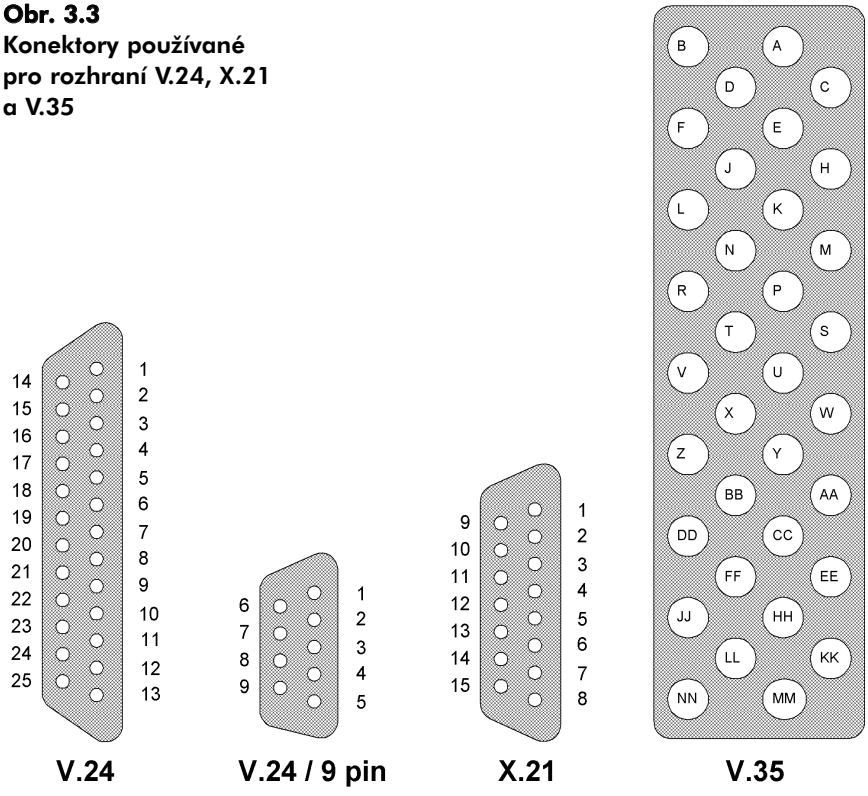
Na fyzické úrovni se pro sériová rozhraní nejčastěji používají normy V.35, X.21 a u PC oblíbená norma V.24. Pochopitelně existují i jiné normy, s těmi se však setkáváme méně často. Nejčasněji se s těmito normami uživatel setká na rozhraní modemu a počítače (resp. směrovače).

Rozhraní V.24 se u PC oblíbené proto, že téměř každé PC je vybaveno porty COM1 a COM2 konstruovanými dle normy V.24.

Rozhraní V.24 je pro přenosové rychlosti nad 64 kb/s zpravidla nedoporučuje, proto pro vyšší rychlosti je nutné použít rozhraní V.35 nebo X.21.

Každá z těchto tří uvedených norem používá jiný typ konektoru, takže propojovací kabely lze těžko zaměnit.

Obr. 3.3
Konektory používané pro rozhraní V.24, X.21 a V.35



U rozhraní V.35 a X.21 jsou pro přenos dat určeny vždy páry vodičů a hodnota signálu je určována mezi vodiči v daném páru (symetricky). Pouze pro signalizaci řízení toku dat se používají signály se společnou zemí (tj. asymetricky). Symetrické signály umožňují použití vyšších frekvencí.

Pomocí rozhraní V.24, V.35 či X.21 je sice možné mezi sebou propojit dva počítače, ale pouze na vzdálenost několika metrů. Na větší vzdálenosti je třeba použít modemy.

V Tab. 3.1 uvádíme význam jednotlivých signálů rozhraní V.24, X.21 a V.35. Pro jednoduchost jsme upravili terminologii na situaci komunikace počítače s modemem. Tj. popisujeme tzv. modemový kabel propojující počítač s modemem.

	Popis signálů	od	Označení signálů		V.24		X.21		V.35		
							15 Pin		Typ	34 Pin	
			EIA	ITU	25 Pin	9 Pin	A	B		A	B
Zem	Ochranná zem		FG	101	1		1			A	
	Signálová zem		SG	102	7	5	8			B	
Data	Vysílání	p	TD	103	2	3	2	9	sym.	P	S
	Příjem	m	RD	104	3	2	4	11	sym.	R	T
Řízení	Výzva k vysílání „Počítač: jsem připraven komunikovat"	p	RTS	105	4	7	3	10	asym.	C	
	Pohotovost k vysílání „Modem: jsem připraven"	m	CTS	106	5	8			asym.	D	
	Modem zapnut	m	DSR	107	6	6			asym.	E	
	Počítač zapnut	p	DTR	108	20	4			asym.	H	
	Nosná	m	DCD	109	8	1	5	12	asym.	F	
	Zvonek	m	RI		22	9			asym.	J	
Hodiny	Generované počítačem	p	TTC		24				sym.	U	W
	Vysílané (z modemu)	m	TC		15				sym.	Y	AA
	Přijímané (z modemu)	m	RC		17		6	13	sym.	V	X
Test	Vzdálená digitální smyčka (V.54/2)	p	RLB		21						
	Lokální analogová smyčka (V.54/3)	p	LLB		18						
	Testovací mód	m	(TM)		25						

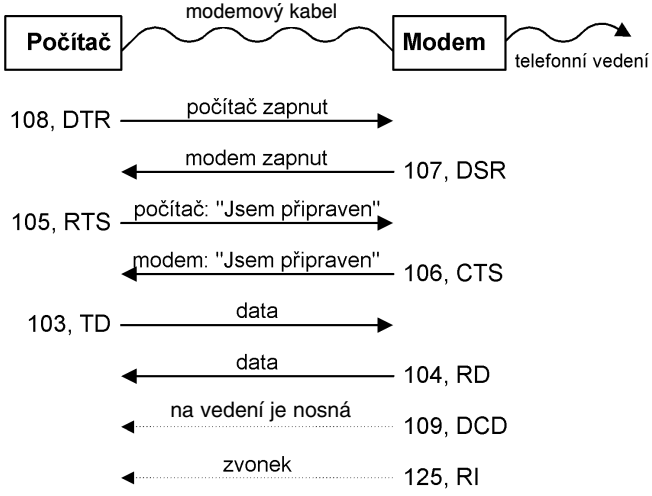
Tab. 3.1 Význam jednotlivých signálů. Sloupec *od* vyjadřuje kdo je zdrojem signálu, buď počítač (p) nebo modem (m). Sloupec *Typ* vyjadřuje, zdali se jedná o symetrický signál (mezi A a B), nebo asymetrický (mezi A a společnou signálovou zemí).

Dialog mezi počítačem a modemem je schématicky vyjádřen na obr. 3.4. Signály DTR a DSR signalizují svému protějšku, že zařízení je zapnuto. V praxi se tyto signály někdy nepoužívají (vývody se nezapojují nebo naopak přímo v konektoru jsou vývody DTR a DSR propojeny). V případě, že signály DTR a DSR nejsou zapojeny, pak je třeba, aby oba konce spojení byly na tuto situaci připraveny (konfigurovány), aby nekonečně nečekaly na signál druhé strany.

Význam signálů RTS a CTS spočívá v řízení toku dat. V případě, že modem má svou vyrovnávací paměť plnou, pak shodí signál CTS a počítač tak signalizuje svému protějšku, aby pozastavil odesílání dat. Po uvolnění vyrovnávací paměti modem opět obnoví signál CTS a počítač jej může opět zásobovat da-

ty. V opačném směru, pokud počítač momentálně není schopen přijímaná data zpracovávat, pak shodí signál RTS.

Je možná i eventualita, že se signály RTS a CTS nepoužijí vůbec (např. nejsou zapojeny příslušné vývody ve šňůře). Pak je třeba konfigurovat oba konce na tuto situaci. V tomto případě je možné k řízení toku dat využít datové signály (vždy v opačném směru). Je-li např. třeba pozastavit příjem, pak se do vysílání vloží znak XOFF. Obnovení se provede znakem XON. Tento protokol XON/XOFF je také třeba na obou koncích spojení shodně konfigurovat a využití má pouze u asynchronních spojů rozhraní V.24, kdy se přenáší znaky.



Obr. 3.4
Schématické znázornění dialogu mezi počítačem a modemem

Signály data (TD i RD) mohou na počátku komunikace přenášet pouze data mezi počítačem a modemem – např. AT-příkazy pro vytáčení. Teprve později, po navázání spojení mezi modemy, mohou být signály TD a RD využívány i pro přenos dat mezi počítači.

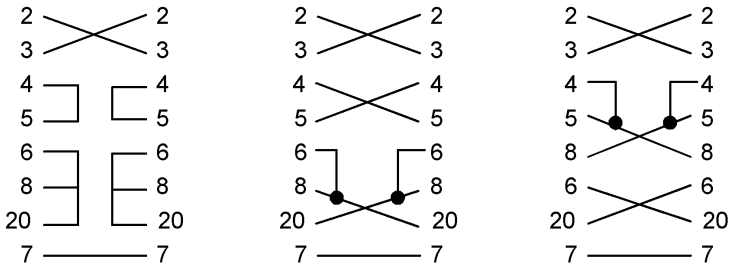
Řízení toku dat pomocí signálů RTS a CTS je efektivní zejména u rozhraní V.24. Avšak rozhraní V.35 a X.21 jsou spíše určena pro vyšší rychlosti. Takovým rozhraním můžeme být připojeni např. k poskytovateli sítě FrameRelay. V takovém případě fyzicky jedním rozhraním (*interface*) prochází několik logických rozhraní (*subinterface*), v případě FrameRelay každé logické rozhraní odpovídá jednomu DLCI, tj. jednomu virtuálnímu okruhu. V případě zahlcení jednoho virtuálního okruhu nelze pozastavit tok dat pomocí signálů RTS či CTS, protože takové pozastavení by znamenalo pozastavení všech logických rozhraní (subinterface) bez ohledu na to, že zahlceno může být jen jedno.

3.1.5 Nulový modem

Pokud byste chtěli pomocí rozhraní V.24, X.21 či V.35 přímo propojit dva počítače stojící vedle sebe, pak propojovací kabel musí být speciálně zapojený. Problém je v tom, že signály, které jedna strana na příslušném pinu vysílá, musí přijímací strana přijímat na pinu určeném pro příjem („vysílání musí být překříženo s příjmem“).

Takováto propojovací šňůra se nazývá nulový modem.

V případě synchronního přenosu musí jedna strana poskytovat synchronizaci. Neumí-li ani jedna ze stran generovat hodiny, pak se nulový modem nemůže skládat pouze ze šňůry, ale musí mezi oběma počítači musí být ještě krabička, která vytváří synchronizační takt (hodiny).



Obr. 3.5
Některé varianty zapojení nulového modemu pro rozhraní V.24 / 25 pin.

3.2 Modemy

Pro připojení na větší vzdálenosti se často používá telefonní síť. Telefon je používán pro zvukovou komunikaci. Chceme-li použít telefonní vedení pro počítačovou komunikaci, pak se musí datové informace na telefonní vedení modulovat a na druhé straně demodulovat. Komunikace je ale obousměrná, takže na obou koncích je potřeba **mo**dulátor/**dem**odulátor, tj. **modem**.

Modem je zařízení, které se bude připojovat k počítači či směrovači modemovým kabelem (tj. v případě PC rozhraním V.24 na COM-port PC). Na druhý vývod modemu se připojuje telefonní linka.

Modem může být do PC i vestavěn nebo realizován na kartě PCMCIA vkládané do notebooku. V takovém případě není třeba modemový kabel.

Linku mezi modemy můžeme postavit vlastními silami (např. mezi budovami firmy), pak se jedná s největší pravděpodobností o pevnou linku.

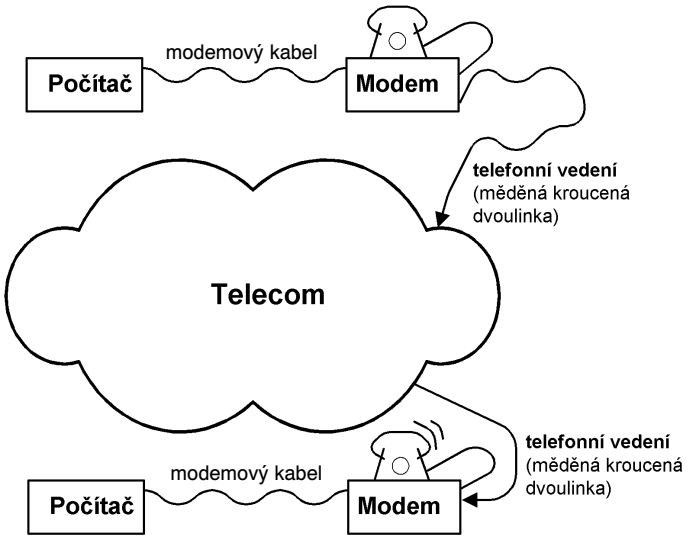
Pokud využijeme služeb telefonního operátora (např. Telecom), pak máme v zásadě dvě možnosti:

- Komutovaná linka
- Pevná linka

3.2.1 Komutovaná linka

S komutovanou linkou se každý z nás již setkal při běžném telefonování. Nejprve se vytočením telefonního čísla vytvoří virtuální okruh, vytvořený okruh je možné použít k telefonnímu hovoru nebo k přenosu dat.

Nejčastěji využíváme již existující telefonní připojení, tj. modem se připojí do uživatelovy účastnické telefonní zásuvky. Telefonní aparát se pak připojí na další vývod modemu. Uživatel vydá modemu počítačem příkaz, aby odpojil telefonní aparát a využil telefonní linku pro počítačovou komunikaci. Po ukončení komunikace se telefonní aparát opět připojí a je možné jej zase používat k telefonním hovorům.



3.2.2 Pevná linka

Druhou variantou je **pevná linka**. Nevyhovuje-li nám stále vytáčet telefonní čísla nebo např. mít počítačovou komunikací neustále obsazován telefon, pak si můžeme telefonní linku pro počítačovou komunikaci pronajmout, tj. technici trvale propojí telefonní okruh. Hovoříme pak o pevné lince.

Výhody pevné linky jsou nesporné: nemusíme stále vytáčet, máme stálé spojení a v neposlední řadě vše je za předem daný paušální poplatek. Firmy budou používat pevné linky, protože je to nejenom pohodlnější, ale lze na nich dosáhnout i vyšších přenosových rychlostí.

Při vyšších rychlostech přenosu se v případě pevných linek někdy nepoužívá pouze jeden telefonní okruh („dvoudrát"), ale dva telefonní okruhy („čtyřdrát"). Jeden okruh je určen pro vysílání a druhý pro příjem, takže okruhy musí být vzájemně překříženy, tj. okruh, který z jednoho modemu vychází jako vysílání musí být do druhého modemu přiveden jako příjem.

3.2.3 Modem je „automatický"

Na komutované lince je problém: kdo vytočí telefonní číslo. Kdysi se i na komutovaných linkách používaly „neautomatické" modemy. Což znamenalo, že uživatel musel nejprve pomocí telefonního aparátu vytočit číslo a pak ručně přepnout modem do režimu přenosu dat (přepínač VOICE/DATA).

Tzv. „Automatické" modemy umí po zapnutí přijímat příkazy od počítače, kterými mj. vytočí číslo. Po navázání spojení se takové modemy samy vzájemně dohodnou na nejvyšší možné přenosové rychlosti a automaticky se přepnou do datového režimu.

Pro komunikaci, kterou počítač ovládá modem se dnes používají tzv. **AT-příkazy** zavedené před lety firmou Hayes. AT-příkazy jsou určeny pro ovládání modemu na asynchronních rozhraních, protože každý AT-příkaz se skládá ze znaků (při synchronním přenosu se přenáší bity, nikoliv znaky). Často se AT-příkazy používají i pro nastavení modemu k synchronnímu přenosu v případě, že je použito rozhraní V.24. Postupuje se tak, že pomocí tlačítek na modemu se modem přepne na tovární nastavení. V továrně bývá nastaven asynchronní přenos. Pak se modem připojí na COM-port PC. Z PC se pomocí

aplikace Terminal (resp. Hyperteminál) vyšlou do modemu AT-příkazy, kterými se modem nastaví. Posledním příkazem pak je přepnutí modemu na synchronní režim. Modem jakoby zmrzne (protože již chce komunikovat synchronně), ale PC komunikuje asynchronně. Nastavený modem se pak přenese např. ke směrovači, kde bude sloužit pro synchronní přenos.

AT-příkazy

Pomocí AT-příkazů lze z počítače ovládat modem. AT-příkazy jsou jednoduché povely. Např. příkaz ATH znamená, že počítač do modemu odešle (resp. odešle na COM-port) řetězec ATH. Modem pak řetězec ATH interpretuje jako příkaz.

Počítač nejprve komunikuje pomocí AT-příkazů s místním modemem, po navázání spojení mezi modemy místní modem signalizuje AT-příkazem CONNECT navázání spojení a modem se přepne do datového režimu. V tomto okamžiku mohou počítače přímo komunikovat, tj. z hlediska počítačů nastane situace, jakoby na lince žádné modemy nebyly (jakoby počítače byly propojeny nulovým modemem). Pokud chce počítač přepnout modem opět do příkazového režimu, aby mohl odesílat AT-příkazy, pak jako data odešle řetězec „+++“.

Pokud pracujete např. na Windows NT, pak si spusťte aplikaci Hyperteminál. Vytvořte připojení libovolného jména na libovolné telefonní číslo. V menu Připojení zvolte připojený zapnutý modem a zmáčkněte tlačítko Konfigurovat. Ve volbě Možnosti, pak zvolte „okno terminálu zobrazit před vytočením čísla“. Volby potvrďte a nakonec zvolte Vytočit. Objeví se okno „Obrazovka terminálu před vytočením“. V tomto okně pak můžete procvičovat AT-příkazy popisované v následujících odstavcích.

PC nejprve pošle do modemu příkaz AT („modeme, jsi schopen pracovat“). Je-li modem připraven, pak odpoví OK (Vyzkoušeli jste si napsat do okna „Obrazovka terminálu před vytočením“ znaky AT?).

Nyní může PC poslat modemu příkaz k vytočení čísla ATD**tč** (např. ATDP1234560), kde t je typ vytáčení (P=pulsně, T=tónově) a č je telefonní číslo protějšího účastníka. Protější modem zvedne a oba modemy se dohodnou na nejvyšší možné přenosové rychlosti. Modem na straně PC vrátí PC příkaz CONNECT, kde jako parametr může udat dohodnutou rychlost mezi modemy. Poté se oba modemy přepnou do datového režimu, tj. oba počítače mohou začít mezi sebou přenášet data jakoby tam žádné modemy nebyly. Celý mechanismus je znázorněn v tabulce 3.1, kde je abstrahováno od chybových stavů i od AT-příkazů pro nastavení modemu, který spojení očekává.

Dnes se modemy, které neumí vytáčet čísla používají pouze pro pevné synchronní linky, kde vytáčení není třeba.

3.2.4 Synchronní přenos

Již jsem se zmínil o tom, že přenos může být **synchronní** nebo **asynchronní**. Synchronní modemy slouží pro synchronní přenos, asynchronní pro asynchronní přenos. Dnešní modemy umí zpravidla oba druhy přenosu.

Poznamenejme, že PC podporuje standardně pouze asynchronní přenos, proto modem, který je nastaven jako synchronní, je před použitím na PC třeba nastavit do asynchronního režimu – jinak se jeví jako pokažený. Jiná je situace u modemů vkládaných do počítače jako karty PCMCIA nebo přídavné karty, ty nevyužívají standardní COM-porty, proto se v takovémto případě může teoreticky použít i synchronní přenos.

	Místní počítač		Místní modem (vytáčející)	Telefonní okruh	Vzdálený modem (očekávající)		Vzdálený počítač
Signalizace	*105, RTS*	→				←	*105, RTS*
		←	*106, CTS*		*106, CTS*	→	
AT-příkazy	**AT**	→		nečinný			
		←	**OK**				
	ATDtč	→					
			vytáčí	vytváření okruhu (dohoda na nejvyšší možné rychlosti)	**zvedá**		
Signalizace		←	*109, DCD*		*109, DCD*	→	
AT-příkazy		←	**CONNECT**				
Přenos dat							

Tab. 3.1

Při konfiguraci synchronního modemu nesmíme zapomenout právě jeden modem nastavit jako zdroj hodin (*originate*). V případě, že zdrojem hodin je počítač, pak jako *originate* nastavíme modem na straně počítače generujícího hodiny, navíc musíme tomuto modemu nakonfigurovat, že využívá externí hodiny.

Dnešní modemy pro rychlosti pod 64 kb/s umí většinou jak asynchronní, tak synchronní režim. Modemy pro rychlost 64 kb/s a výše bývají zpravidla synchronní.

Existují i modemy podporující tzv. autosynchronní režim. Tj. s počítačem komunikují asynchronně, data ukládají do paměti a poté je synchronně modulují do telefonního vedení. S těmito modemy se v Internetu setkáváme jen zřídka, jejich hlavním těžištěm jsou veřejné datové sítě na bázi protokolu X.25, resp. X.32.

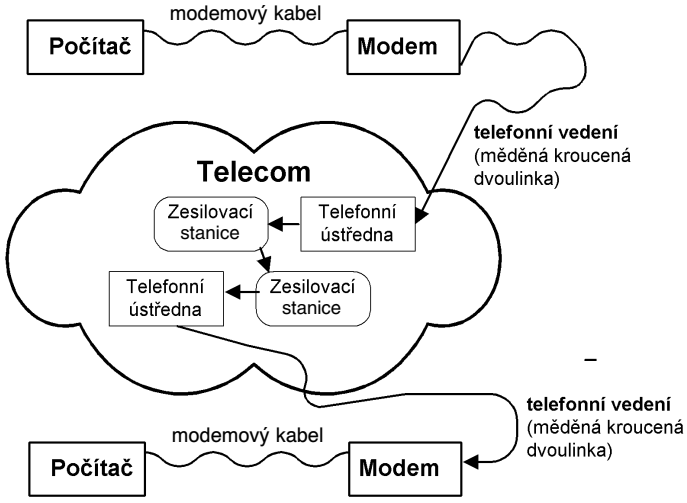
3.2.5 Základní pásmo a přeložené pásmo

Pro přenos zvuku v telefonní kvalitě je nutné přenášet pásmo 0,3 až 3,4 kHz.

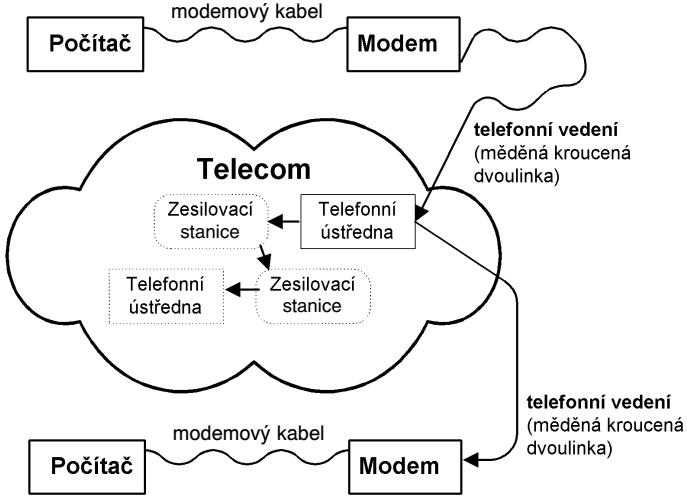
Telefonní vedení vede od domovní telefonní zásuvky zpravidla na svorkovnici místní telefonní ústředny. Místní telefonní ústředna přepojuje telefonní okruh přes další ústředny až na ústřednu v místě volaného účastníka. Jelikož se často jedná o velké vzdálenosti, tak signál musí být po jistých vzdálenostech zesilován zesilovacími stanicemi (viz obr. 3.7). Zesilovací stanice zesilují signál pouze v pásmu 0,3 až 3,4 kHz. Pakliže se telefonní vedení vede přes zesilovací stanice, pak modemy musí signál nesoucí datové informace přeložit do tohoto pásma. Hovoříme tak o přeloženém pásmu (*Voice Band*).

Přeložené pásmo se dnes používá pro přenosové rychlosti do 56 kb/s.

Jiná je situace u vedení, které neprochází zesilovacími stanicemi, např. natáhnete-li si sami vedení mezi dvěma budovami. Nebo oba konce spojení jsou vyvedeny na svorkovnici téže telefonní ústředny, jak je znázorněno na obr. 3.8 (a na cestě nebyla žádná zesilovací stanice).



Obr. 3.7
Telefonní okruh procházející přes ústředny a zesilovací stanice



Obr. 3.8
Modemy pro základní pásmo vyžadují přímý metalický spoj

V takovém případě je možné napojit obě vedení přímo na sebe a přenášet vedením podstatně širší základní pásmo (*Base Band*).

Modemy pracující v základním pásmu je možné docílit podstatně vyšší rychlost přenosu. Je zde běžná rychlost i stovky kb/s na dvoudrátových vedeních dlouhých i několik km. Na čtyřdrátových vedeních (dvě dvojlinky) se dnes setkáváme s rychlostí 2 Mb/s i vyšší.

Modemy pracující v základním pásmu nebývají „automatické“, protože se používají na pevných linkách a navíc používají synchronní přenos.

3.2.6 Přenosová rychlost

Modem posílá/přijímá data na dvě strany: jednak směrem k počítači a jednak směrem do telefonního vedení. Obě rychlosti přenosu přitom nemusí být stejné. Problém vzniká pouze v případě, že se data na nějakou dobu v modemu nahromadí, proto dnešní modemy bývají vybaveny vyrovnávací pamětí.

Hovoří-li se však o přenosové rychlosti modemu, pak se má na mysli přenosová rychlost po telefonním vedení. Přenosová rychlost je dána doporučeními ITU, která modem podporuje. Nejaktuálnější doporučení na okruzích v přeloženém pásmu jsou uvedena v tab. 3.2. (Jiná je situace s přenosovými rychlostmi na pevných linkách s přenosem v základním pásmu nebo na digitálních okruzích.)

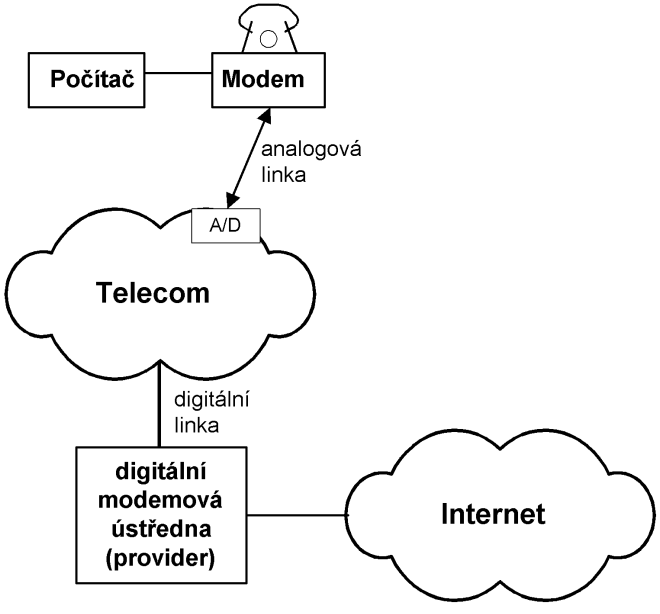
Tab. 3.2

Doporučení ITU	Rychlost v Kkb/s
V.32	9,6
V.32bis	14,4
V.34	28, 8
V.34+	33,6
V.90	56 (od ústředny k modemu) 33,6 (od modemu k ústředně)

Doporučení V.90

Doporučení V.90 není určeno pro každý případ použití modemu. Např. se nehodí pro spojení z domova do kanceláře. Doporučení V.90 je však velice vhodné pro připojení PC k poskytovateli Internetu, pokud je poskytovatel připojen k Telecomu digitální linkou.

Obr. 3.9
Doporučení V.90



Dnešní vybavení telefonních ústředen je plně digitální. Signál jdoucí analogovou linkou od uživatele do Telecomu je na straně Telecomu A/D převodníkem digitalizován a následně zpracováván jako data. Pokud by signál putoval k uživateli klasického analogového telefonu, pak by na straně volaného byla opět nutná konverze A/D převodníkem do analogového tvaru.

Dnešní poskytovatelé Internetu bývají však přímo připojeni velkokapacitním digitálním okruhem k Telecomu. Pokud tomu tak je, pak na straně poskytovatele konverze signálu není nutná. Zajímavý je však opačný směr. Poskytovatel předává digitálně data Telecomu a data jsou převáděna na analogová v Telecomu až na straně uživatele. Co je na tom zajímavého? K informační ztrátě totiž dochází při konverzi signálu z analogového na digitální – nikoliv však v opačném směru. Takže ve směru od Telecomu k uživateli je možné linku budit vyšší rychlostí – až 56 kb/s.

3.2.7 Komprese dat

Kdyby se modemu podařilo data před přenosem zkomprimovat (zhustit) a přijímací modem by je uměl dekomprimovat, pak by se při nezměněné rychlosti modemu přeneslo více dat. Komprese je možná pouze u asynchronního přenosu, který přenáší znaky.

Firma Microcom vyvinula protokol MNP 5 pro kompresi dat. ITU vydalo pro kompresi dat doporučení V.42bis.

Použijeme-li kompresi dat, pak na linkách s přenosovou rychlostí 28,8 kb/s se ve špičkách v praxi přenáší data i rychlostmi nad 100 kb/s.

Proč používáme na rozhraní mezi počítačem a modemem vyšší přenosovou rychlost než na lince mezi modemy? Odpověď je jednoduchá. Je to důležité zejména v případě, že modemy při komunikaci mezi sebou používají kompresi dat. Maximální podporovaná rychlost COM-portu PC je 115 200 b/s, což umožňuje pro rychlost 28,8 kb/s maximální kompresi 4:1. Některé typy dat (např. video) je možné komprimovat v některých případech až v poměru 40:1.

Z těchto důvodů začínají být populární modemy, které se nepřipojují na COM-port PC, ale paralelní port PC, který umožňuje rychlost až 300 000 b/s.

3.2.8 Detekce chyb

Myšlenka spočívá v tom, že data se mezi modemy přenáší v celcích – datových blocích. Vysílající modem z datového bloku vypočte kontrolní součet a přidá jej k přenášeným datům. Přijímající modem z dat (bez kontrolního součtu) opět vypočítá kontrolní součet a oba kontrolní součty porovná. Jsou-li oba kontrolní součty stejné, pak data předá cílovému počítači. Nejsou-li kontrolní součty stejné, pak přenos byl chybný a modem si vyžádá opakování dat.

Původně nejrozšířenější protokoly pro detekci chyb zavedla firma Microcom pod označením MNP 2 až 4. Posléze protokoly pro detekci chyb specifikovala i ITU do doporučení V.42.

Protokol V.42 popisuje také dvoufázový proces navazování spojení. V první „detekční“ fázi modemy na základě výměny předdefinovaných znaků vzájemně zjišťují své možnosti a ve druhé „potvrzovací“ fázi si modemy vzájemně vymění své možnosti ohledně maximální velikosti datového bloku a počtu odvysílaných bloků, po kterém je požadováno potvrzení o správném přijetí bloků.

3.3 Digitální okruhy

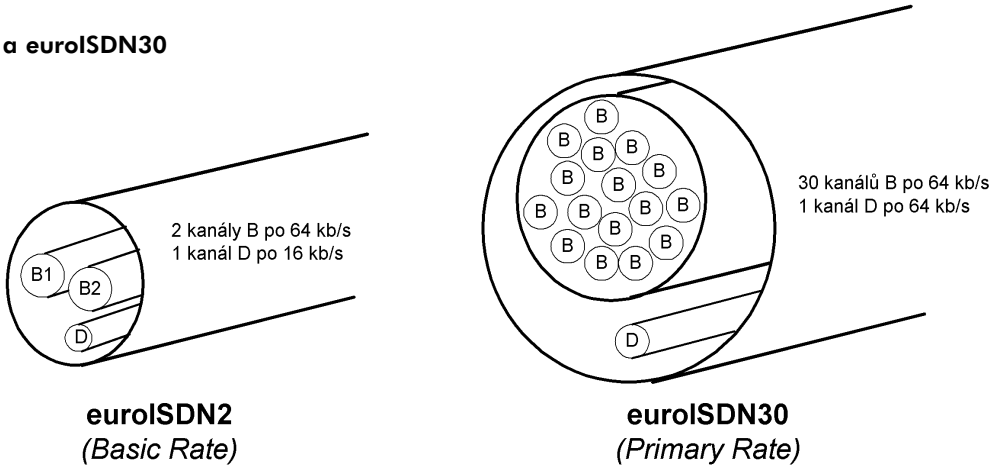
Doposud jsme popisovali analogové okruhy. Život však jde dále a analogové rozvody jsou nahrazovány digitálními. Nejprve se tak dělo uvnitř Telecomu. Dnes však i uživatelé mohou používat digitální okruhy – ISDN.

3.3.1 ISDN

Telecom u nás nabízí připojení euroISDN2 a euroISDN30. To jsou spíše obchodní označení, v literatuře se spíše setkáme s anglickými názvy:

- *Basic Rate* pro euroISDN2, což je typ připojení, kdy ve fyzicky jednom vedení (jedné kroucené dvojlince) jsou dva datové kanály B každý o kapacitě 64 kb/s a jeden signalizační kanál D o kapacitě 16 kb/s.
- *Primary Rate* pro euroISDN30, což je typ připojení, kdy ve fyzicky jednom vedení (např. lince E1) je třicet datových kanálů B, každý o kapacitě 64 kb/s a jeden signalizační kanál D o kapacitě 64 kb/s.

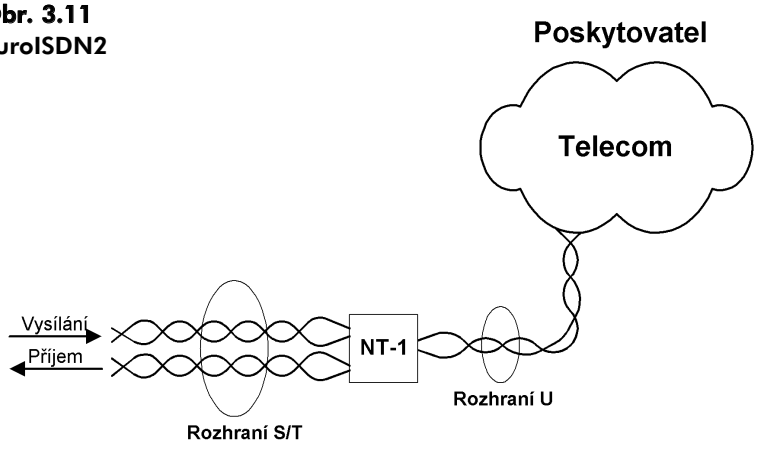
Obr. 3.10
euroISDN2 a euroISDN30



3.3.1.1 euroISDN2

euroISDN2 využívá stávající telefonní rozvody kroucenou dvoulinkou. Tj. většinou lze využít pro rozvod euroISDN2 i stávající metalické rozvody pro analogové telefony. Připojení ISDN popisuje norma V.110. Kroucená dvoulinka přicházející od Telecomu vytváří tzv. rozhraní U (viz obr. 3.11).

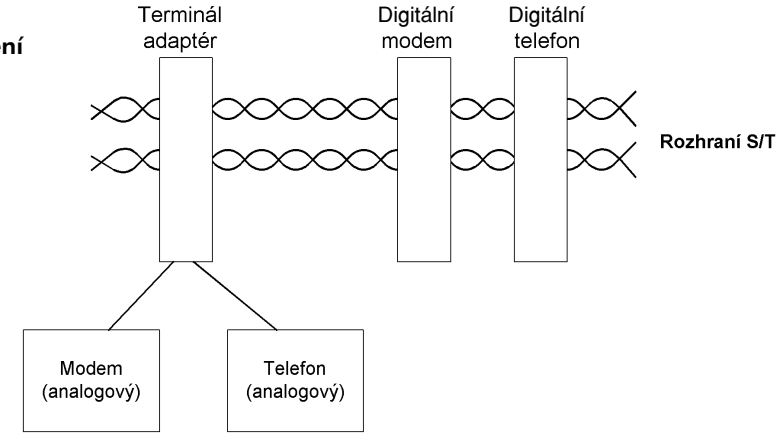
Obr. 3.11
euroISDN2

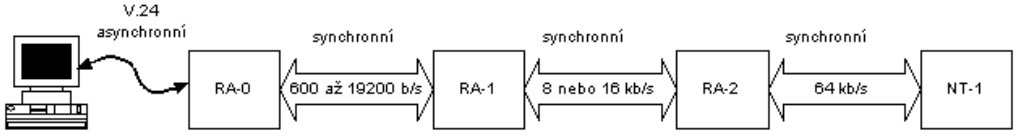


Rozhraní U je rozhraním mezi Telecomem a zařízením (krabičkou) NT-1, kterou rovněž dodává a instaluje Telecom. Ze zařízení NT-1 vychází dvě dvoulinky rozhraní S/T. Rozhraní S/T má charakter sběrnice, na kterou se připojují jednotlivá digitální zařízení. Pokud např. kupujeme směrovač pro připojení na ISDN, pak jej kupujeme s rozhraním S/T – nikoliv s rozhraním U, protože rozhraní U je u nás v režii Telecomu.

Jak je znázorněno na obr. 3.12, jednotlivá zařízení se na rozhraní S/T připojují jako na sběrnici. Jelikož euroISDN2 má k dispozici dva datové kanály B, tak v jednom okamžiku mohou komunikovat současně dvě zařízení (např. digitální telefon a digitální modem nebo dva digitální telefony atd.).

Obr. 3.12 Připojování zařízení na rozhraní S/T





Obr. 3.12a Připojení PC s rozhraním V.24

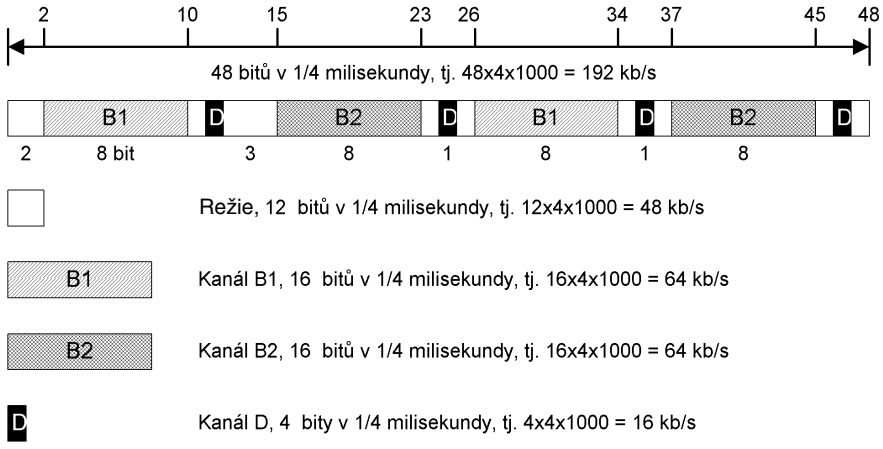
Problémem ISDN je např. jak připojit k ISDN běžné PC s asynchronním rozhraním V.24 např. rychlostí 14,4 kb/s. Na tomto velice nepraktickém příkladu si budeme demonstrovat hloubku problému. ISDN používá synchronní přenos 64 kb/s, tj. předpokládá, že na zařízení NT-1 přichází synchronní signál 64 kb/s.

PC na svém COM portu komunikuje asynchronně a v našem případě nízkou rychlostí 14,4 kb/s. Jak je znázorněno na obr. 3.12a, tak je PC připojeno na zařízení RA-0 (*Rate Adaptation 0)*. Zařízení RA-0 převádí asynchronní signál na synchronní.

Zařízení RA-1 doplní signál o výplň na konstantní rychlost 16 kb/s. Zařízení RA-2 pak doplní signál další výplní na rychlost 64 kb/s. Signál tak může být dále standardně zpracováván telefonními ústřednami. Z uvedeného příkladu je patrné, že nejvýhodnější je vložit do PC kartu, která má přímo synchronní rozhraní 64 kb/s („ISDN kartu“).

Celý mechanismus zařízení RA-0, RA-1 a RA-2 se v principu také používá pro datové služby GSM, protože datový kanál GSM používá rychlost 12 kb/s, avšak tento kanál musí být konvertován do standardního ISDN, které předpokládají ústředny.

Obr. 3.13 Rozdělení euroISDN2 na jednotlivé sloty



Počítač se propojí s mobilním telefonem. Zařízení RA-0 je v tomto případě buď:

- Součástí mobilního telefonu v případě telefonu typu MT-2 (*Mobile Termination type 2).* V tomto případě se mobilní telefon propojuje s počítačem rozhraním V.24 rychlostí až 9600 kb/s. Zařízení RA-0 v mobilním telefonu převání asynchronní signál na synchronní a signál vyplní na rychlost 12,6 kb/s.
- Vloženo do PC (jako karta v případě mobilního telefonu typu MT-1), pak z této karty přímo vystupuje rozhraní S rychlostí 12 kb/s.

Datový kanál mobilního telefonu používá přenosovou rychlost 12,6 kb/s synchronně. Tento kanál je přenesen radiovým kanálem do sítě GSM, kde se konvertuje v zařízeních TRAU na rychlost 64 kb/s analogicky jako to dělají zařízení RA-1 a RA-2.

Kanál D v ISDN slouží k signalizaci, tj. např. k sestavení virtuálního okruhu („vytočení čísla“) či v případě kdy jsou oba kanály B obsazeny telefonními hovory, tak kanálem D může být signalizován další přicházející hovor. Současný hovor můžeme pozdržet a hovořit na nově příchozím hovoru, je možné signalizovat číslo volajícího účastníka atd.

I když máme k dispozici pouze dva datové kanály B, tak na rozhraní S/T můžeme umístit více než dvě digitální zařízení (v jednom okamžiku však mohou komunikovat nejvýše dvě). Např. pět digitálních telefonů. Každý z nich může mít své samostatné číslo. Z hlediska uživatele se to jeví tak, že má pět „telefonních linek“, ale současně může používat pouze dvě.

Na rozhraní S/T může být také připojen tzv. terminál adaptér, který umožňuje připojení klasických analogových modemů, faxů a telefonů.

Ještě se musíme zmínit o termínu „digitální modem“. Mnohým je toto označení trnem v oku, protože modem je zařízení, které moduluje/demoduluje digitální signál na analogový. Digitální modem přitom nic takového nedělá, ten pouze konvertuje jeden typ digitálního rozhraní (např. V.24 v případě PC) na rozhraní S/T (konektor RJ45).

ISDN používá synchronní přenos dat (synchronní přenosem dat ve smyslu kap. 1.3.1, který je synchronním přenosem i ve smyslu kap. 3.1.3). euroISDN2 používá přenos rychlostí 192 kb/s, který je rozdělen do slotů pro jednotlivé kanály, jak je znázorněno na obr 3.13.

3.3.1.2 Protokoly vyšších vrstev a signalizace

Kanály B lze použít k telefonnímu hovoru, pak jeden slot příslušného kanálu B obsahuje jeden vzorek hovoru (zvukového signálu). Nás však zajímá spíše přenos dat.

Při přenosu dat jsou do kanálů B vkládány rámce protokolu LAPB a do kanálu D jsou vkládány rámce protokolu LABD. (Existují další linkové protokoly používané ISDN, jako např. LABF, I.465, V.120 atd.).

Protokoly LAPB a LAPD jsou odvozeny, jak je znázorněno na obr. 3.14, od protokolu HDLC (viz kapitola 4.1.3).

Do rámců protokolu LAPB se vkládají síťové pakety. Pakety síťové vrstvy mohou patřit např. protokolu X.25. Pro nás je důležité, že do rámců protokolu LAPB mohou být vkládány i IP-datagramy.

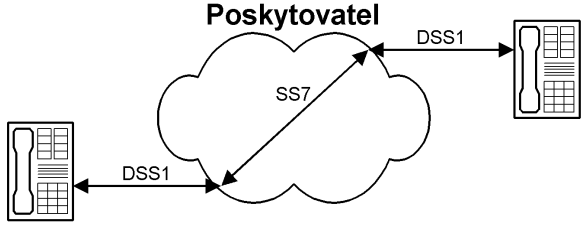
Zatím jsem popisoval stav, kdy je kanál B využit k přenosu dat či hovoru. Otázkou zůstává vytvoření („vytočení“) virtuálního okruhu, což je jednou z funkcí signalizace, která probíhá na kanálu D.

Obr. 3.14 Schématické znázornění rámce linkových protokolů LAPB a LAPD

Rozeznáváme dvě úrovně signalizace. Pomocí signalizace DSS1 vyžaduje uživatel zřízení okruhu a další služby. Na straně poskytovatele se požadavky v signalizaci DSS1 zabalí do signalizace SS7 (slouží i pro signalizaci klasických analogových hovorů) a přenese se na stranu volaného. Na straně volaného se opět pomocí signalizace DSS1 signalizuje příchozí hovor.

Signalizace DSS1 je specifikována doporučeními ITU Q.931 a Q.932. Protokol Q.931 poskytuje základní služby jako vytvoření okruhu a z hlediska síťového modelu ISO OSI je síťovým protokolem. Protokol Q.931 poskytuje další služby jako je např. přidržení hovoru, z hlediska síťového modelu ISO OSI pokrývá protokol Q.932 transportní až aplikační vrstvu.

Obr. 3.15 Signalizace DSS1 a SS7



V signalizaci DSS1 se posílají zprávy. V tabulce 3.3 jsou uvedeny některé základní typy zpráv.

Tab. 3.3 Některé typy zpráv Q.931

Typ zprávy	Analogie u analogové telefonie
Setup	Zvednutí mikrotelefonu
Call Proceeding	Vytáčení
Alerting	Vyzvánění
Connect	Zvednutí
Disconnect/Release	Zavěšení

Obr. 3.16
DSS1 z hlediska modelu ISO OSI

3.3.2 Linky E

V předchozích kapitolách jsme hovořili o lince E1. Linka E1 je nejnižším patrem v hierarchii přenosových cest specifikovaných v doporučeních ITU pod označením G.702 a G.703. Existují i jiné hierarchie přenosových cest pro Severní Ameriku (linky T), pro Japonsko a také pro transatlantické spoje.

Základem je linka o přenosové rychlosti 64 kb/s (v tab. 3.4 označena jako E0). Linka E1 pojme 32 takových základních linek. Linka E2 pojme 4x E1. Používanější je však E3, která pojme 16x E1 (resp. 4x E2) atd.

Tab. 3.4

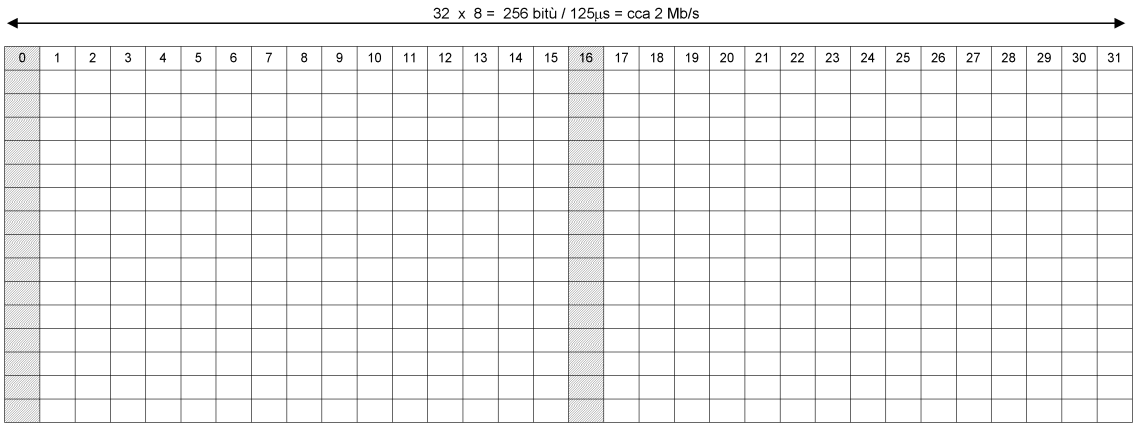
Linka	Přenosová rychlost kb/s
(E0)	64
E1	2 048
E2	8 448
E3	34 368
E4	139 264

I když linka E1 pojme 32 linek E0, tak využít jich lze pouze 30. Sloty 0 a 16 se používají pro režii, která zabezpečuje jednotlivé rámce. Sloty 0 a 16 obsahují také kontrolní součet super-rámce (viz obr. 3.17).

Pronajmout si lze jeden slot (64 kb/s) nebo více slotů. Maximálně tedy v případě linky E1 si lze pronajmout 30 x 64 = 1 920 kb/s. Pokud si pronajmeme celých 1 920 kb/s, pak se hovoří o „neděleném E1".

Fyzicky může být E1 přivedena dvěma páry kroucené dvojlinky (120 Ω), koaxiálním kabelem (75 Ω) nebo optickým kabelem.

Častější je případ, kdy si pronajímáme pouze n x 64 kb/s. Vlastní E1 linku pak ani nevidíme, ta končí u poskytovatele, kde je připojena na multiplexor. Připojeni jsme pak metalickým spojem z multiplexo-



Obr. 3.17 Super-rámec E1 rozděleny na 32 slotů po 64 kb/s

ru. Jelikož je připojení od nás k multipexoru provedeno metalickým spojem, tak je možné použít modemy s přenosem v základním pásmu (*Base Band*), které jsou běžně konstruovány pro rychlosti 2 Mb/s (tj. pojmou i „celou E1“). Na naší straně je pak modem s přenosem v základním pásmu (*Base Band*) s konektorem V.35, V.24 či X.21.

3.3.3 IP bez linkové vrstvy

Všimněte si, že digitální okruhy již na fyzické vrstvě zabezpečují některé služby běžné spíše pro linkové protokoly. Jedná se zejména o zajištění integrity přenášených super-rámců.

Jako jedna z technologií budoucnosti se jeví přímé vkládání IP-datagramů do super-rámců fyzické vrstvy. Vždyť na okruzích spojujících dva vzdálené směrovače už linková vrstva nezvýší příliš komfort, ale zato zvýší režii.

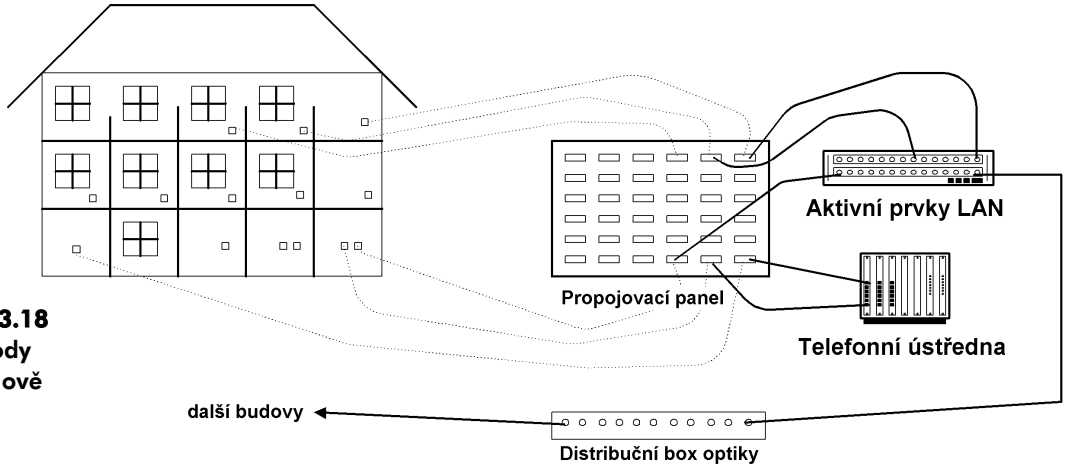
3.4 LAN

Lokální sítě jsou určeny pro propojení počítačů na kratší vzdálenosti (stovky metrů až kilometry). U lokálních sítí závisí volba fyzického rozhraní na volbě linkového protokolu. V dnešní době přicházejí v úvahu zejména čtyři typy linkových protokolů: Ethernet, Fast Ethernet, Gigabitový Ethernet a FDDI. Protokoly Arcnet a Token Ring jsou v praxi málo běžné.

3.4.1 Strukturovaná kabeláž

Strukturovanou kabeláží se rozumí komplexní řešení nízkonapěťových rozvodů v budově. Zahrnuje zejména telefonní rozvody a rozvody pro LAN. Většinou zahrnuje i další rozvody jako jsou bezpečnostní a jiné signalizace.

V jednotlivých místnostech budovy jsou umístěny telefonní zásuvky, zásuvky LAN a jiné vývody. Z těchto zásuvek vedou rozvody na propojovací panel budovy. V případě optických rozvodů jsou optická vlákna vyvedena na distribuční box optiky.



Obr. 3.18
Rozvody v budově

Propojovací panel a distribuční box optiky bývají uzavřeny v jedné skříni (*RackMount*) spolu s aktivními prvky LAN či dokonce i s telefonní ústřednou. Propojení mezi propojovacím panelem a aktivními prvky se provádí propojovacími kabely (*Patch Cord*).

Rozvod od zásuvek na propojovací panel je poměrně drahou záležitostí, protože se mnohdy jedná i o stavební úpravy. Snahou je proto rozvod provést maximálně kvalitně, aby se rozvody nemusely často předělávat. Základní filozofií nových protokolů je pak v maximální míře využít stávající kabeláže. Proto také kvalitním rozvodům původně vytvořeným pro Ethernet 10Base-T nedělal problémy přechod na 100Base-TX, pokud byly rozvody prováděny v kategorii 5. A snahou normy 1000Base-CX je využít týchž rozvodů i pro Gigabitový Ethernet.

Existují normy pro rozvody – tzv. kategorie. Dnes jsou aktuální kategorie:

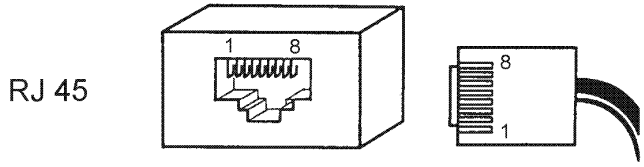
- Kategorie 5, kdy dodavatel garantuje práci v šířce pásma do 100 MHz nezávisle na použitém protokolu (Etherent, Token Ring, CDDI atd.).
- Rozšířená kategorie 5 (nebo také 5+), pracuje rovněž v šířce pásma do 100 MHz, avšak vyžaduje nové způsoby měření parametrů a v některých parametrech je přísnější. Cílem je provozovat Gigabitový Ethernet.
- Kategorie 6 s šířkou pásma do 200 MHz.
- Kategorie 7 s šířkou pásma do 600 MHz.

Dříve existovaly i kategorie 3 a 4. Rozvody dle těchto kategorii je dnes většinou nutné předělat.

3.4.1.1 Měděné rozvody

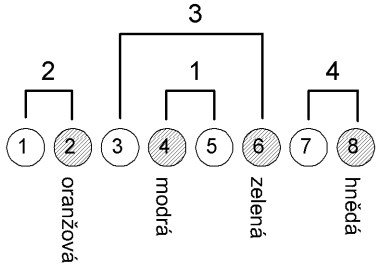
Měděné rozvody se provádí pomocí svazků kroucených dvoulinek. Jednotlivé místnosti se opatřují zásuvkami pro konektor RJ 45 (viz obr. 3.19).

Obr. 3.19
Konektor RJ 45



Konektor RJ 45 („kostka cukru“) obsahuje 8 vývodů pro 4 páry. Nejčastěji se používá zapojení dle EIA 568B (viz obr. 3.20). Toto zapojení umožňuje např. pár číslo 1 použít pro telefon (analogový) a páry 2 a 3 např. pro Ethernet (pár 4 zůstává v tomto případě volný).

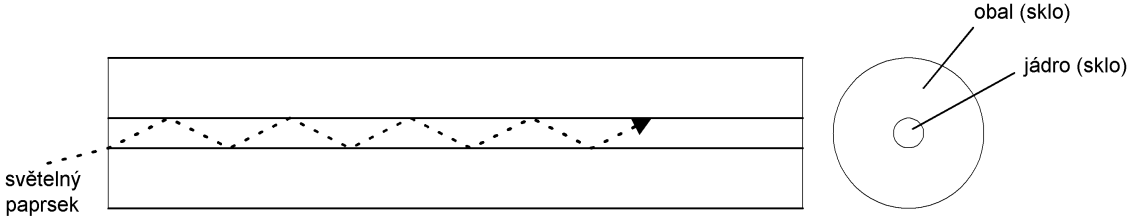
Obr. 3.20
EIA 568B – zapojení jednotlivých párů



3.4.1.2 Optická vlákna

Optická vlákna jsou tvořena dvěma vrstvami skla. Jeden typ skla je použit pro jádro vlákna a jiný typ skla pro obal vlákna. V jádře vlákna je veden optický paprsek, který se postupně odráží od rozhraní mezi dvěma druhy skla (viz obr. 3.21).

Obr. 3.21
Optické vlákno



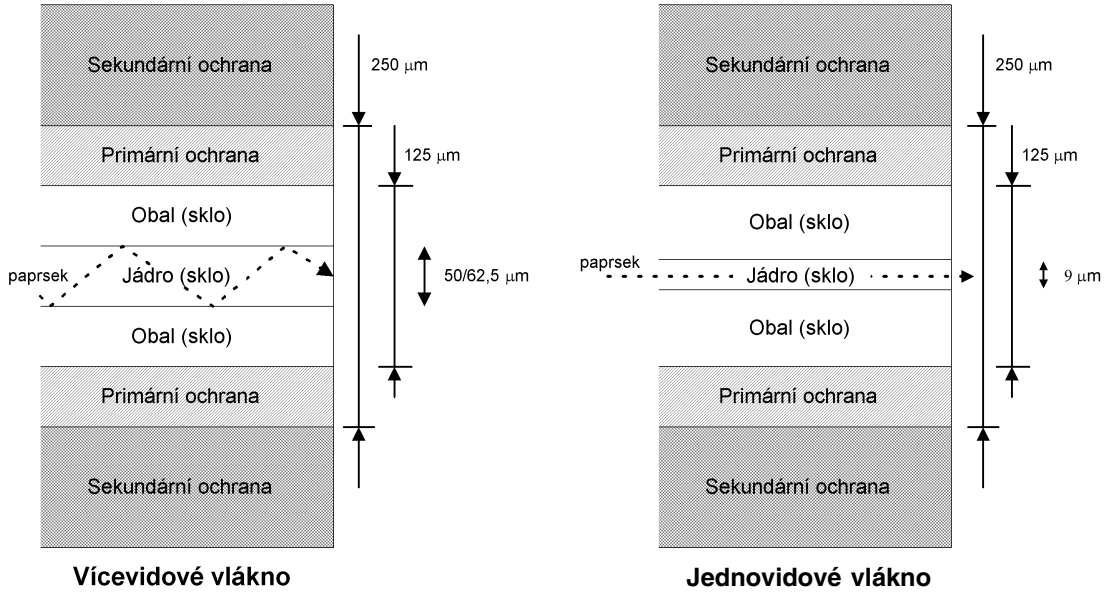
Sklo má nízký optický odpor pouze pro tři vlnové délky světla: 850 nm, 1300 nm a 1500 nm, proto se vždy k buzení optického signálu používá jedna z těchto vlnových délek.

Optické vlákno je vždy simplexní spoj, tj. na jedné straně je vysílač a na druhé straně přijímač. Pro duplexní spoje (což je téměř vždy) je nutná dvojice vláken – pro každý směr jedno vlákno.

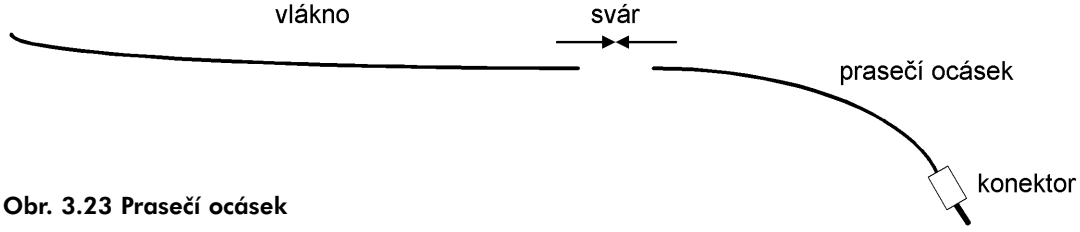
I když vlákno má zpravidla průměr 125 μm, tak jádro vlákna máme dvojího průměru:

- O průměru 50 μm (resp. 62,5 μm), pak hovoříme o vícevidovém vlákně (*multi mod*). Vicevidová vlákna se budí pomocí LED, avšak v poslední době se např. u gigabitového Ethernetu již setkáváme také s buzením laserem.
- O průměru 9 μm, pak hovoříme o jednovidovém vlákně (*single mod*). Jednovidová vlákna mají již tak úzké jádro, že paprsek se šíří jádrem vlákna rovnoběžně, tj. neodráží se od rozhraní mezi oběma druhy skel. Jednovidová vlákna se zásadně budí laserem. Jednovidová vlákna jsou určena pro spoje na velké vzdálenosti.

Na obr. 3.22 je znázorněna ochrana optických vláken. Optická vlákna jsou nejprve obalena tzv. primární ochranou, která zajišťuje pružnost vlákna. Bez primární ochrany je vlákno velice křehké. Sekundární ochrana pak zvyšuje ochranu vlákna. S odstraněnou sekundární ochranou se již setkáváme u optických propojovacích kabelů.



Obr. 3.22 Vícevidové a jednovidové vlákno



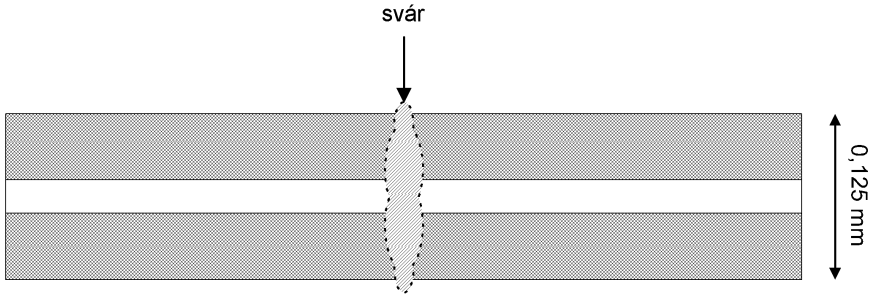
Obr. 3.23 Prasečí ocásek

S optickými kabely, které mají odstraněnu sekundární ochranu se v běžných firemních podmínkách obtížně pracuje. V této sféře jsou populární optická vlákna s tzv. těsnou sekundární ochranou (průměr 900 μm = 0,9 mm), která integruje primární i sekundární ochranu. Takové kabely jsou o něco dražší (proto se nehodí na propojování velkých vzdáleností), ale na druhou stranu je možné na tyto kabely přímo nasadit optické konektory.

Pokud se použijí kabely s primární ochranou, tak se musí používat továrně připravené optické konektory nasazené na kus optického vlákna, tzv. prasečí ocásky (*pig tail*). Prasečí ocásek se pak navařuje na vlákno.

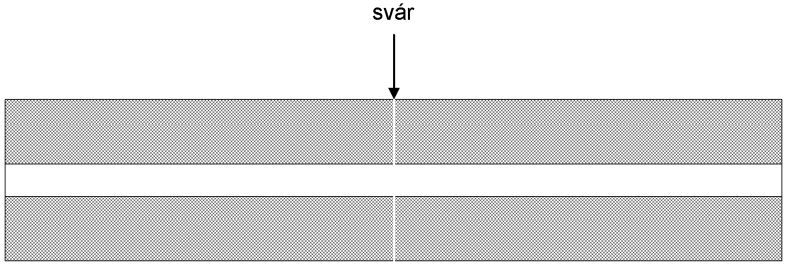
Svár dvou optických kabelů (tj. i navaření prasečího ocásku) je „věda“. Stačí si představit, že se jedná o navaření vlákna skládajícího se ze dvou skel (obalu a jádra), při prostém navaření by se sklovina obou vláken slila a vznikla by tak neprostupná překážka – viz obr. 3.24.

Obr. 3.24
Chybný svár optického vlákna způsobí neprostupnou překážku světelnému paprsku.



Vlákna je třeba svařit tak, aby překážka nevznikla. Kvalitní svár lze provést pouze pomocí nákladného zařízení.

Obr. 3.25
Správně provedený svár netvoří procházejícímu paprsku překážku



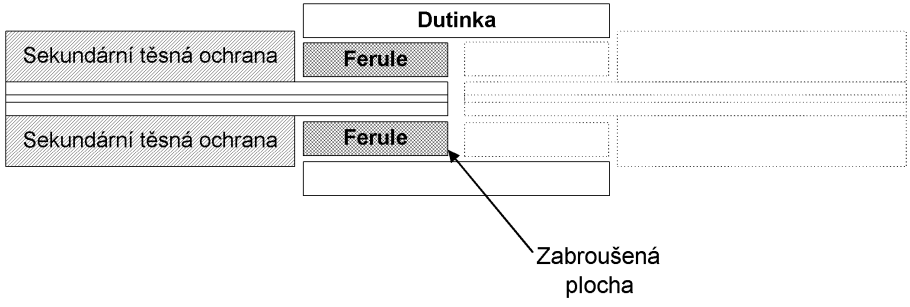
V mnoha případech se jeví efektivnější se svařování vláken vyhnout, použít vlákna s těsnou sekundární ochranou, na konce vláken přímo nasadit optické konektory a vlákna propojit pomocí optických konektorů.

Optická vlákna včetně primární a sekundární ochrany (resp. včetně těsné ochrany) jsou zpravidla uložena v optických kabelech. V optickém kabelu jsou svazky vláken opředeny kevlarem a vše je zapouzdřeno do vnějších obalů kabelu. Vnější obaly jsou pak provedeny podle toho, zdali je kabel určen pro uložení v budově, mezi budovami, či na mořském dně.

Problém je i se zlomením optického vlákna. Pokud by se vlákno zlomilo v ruce, pak dojde k roztříštění jeho konce, proto se lámání provádí speciálním přípravkem, který roztříštění konců omezuje. Přesto se zlomené konce musí ještě zabrousit a zkontrolovat mikroskopem, zdali byly všechny praskliny odbroušeny.

Na obr. 3.26 je na optické vlákno se sekundární těsnou ochranou nasazen optický konektor. Z konce vlákna byla odstraněna ochrana. Na místo ochrany byl na konec vlákna nasazen keramický kroužek – ferule. Konec vlákna i s ferulí byl zabroušen a zabroušená plocha zkontrolována pod mikroskopem.

Obr. 3.26
Optický konektor (bez mechanismu uchycení)



Dvě takto zakončená vlákna se vkládají proti sobě do dutinky, aby se zabránilo posunutí vláken. V dutince přiléhá jádro jednoho vlákna na jádro druhého vlákna a paprsek tak může projít z jednoho vlákna do druhého.

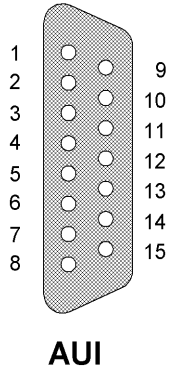
Princip optického konektoru je znázorněn na obr. 3.26. Na tomto obrázku není znázorněn mechanismus, kterým je vlákno přitlačováno do dutinky.

3.4.2 Ethernet (10 Mb/s)

Ethernet používá čtyři typy rozhraní: AUI, BNC, TP nebo optický spoj.

AUI (označované též jako 10BASE-5) je rozhraní (konektor CANNON 15), na které se připojuje kabel propojující počítač s tzv. *transceiverem*. Transceiver je zařízení, které vysílá/přijímá původně na tlustý koaxiální kabel rozvodu LAN. Existují však i transceivery pro rozvod tenkým koaxiálním kabelem („redukce AUI/BNC") i transceivery pro kroucenou dvojlinku („redukce AUI/TP"). Rozhraní AUI je tedy univerzální, protože je v něm napájení případných „redukcí". Naopak redukci TP/BNC je nutné realizovat pomocí samostatného opakovače s vlastním napájením, takže je to dražší záležitost.

Obr. 3.27
Zapojení rozhraní AUI



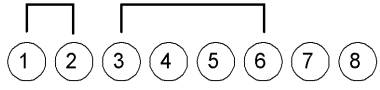
Vývod	Funkce	Vývod	Funkce
1	Kolize – stínění	9	Kolize -
2	Kolize +	10	Vysílání -
3	Vysílání +	11	Vysílání – stínění
4	Příjem – stínění	12	Příjem -
5	Příjem +	13	Napájení +12 V
6	Napájení -	14	Napájení – stínění
7	-	15	-
8	-		

BNC (označované též jako 10BASE-2) je rozhraní pro připojení na tenký koaxiální kabel. Koaxiální kabel je v místě připojení přerušen. Na oba konce přerušení se speciálními kleštěmi připevní BNC-konektory. Oba BNC-konektory se připojí na BNC T-konektor, který je připojen do počítače.

Kroucená dvojlinka (zkratkou TP, označovaná též jako 10BASE-T) se připojuje konektorem RJ45 („kostka cukru"). Kroucená dvojlinka vede zpravidla společně s telefonním rozvodem na centrální propojovací panel.

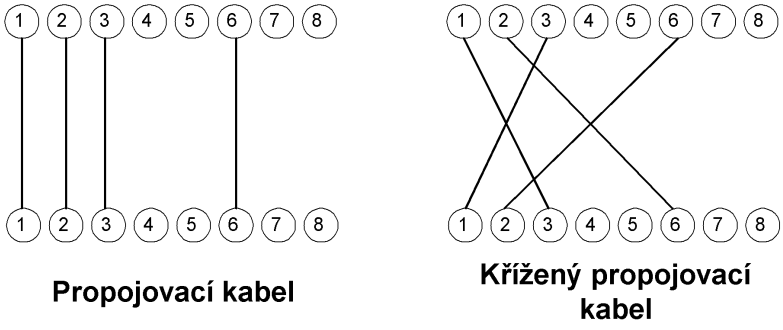
TP používá dva páry v konektoru RJ45, jak je znázorněno na obr. 3.28. (Všimněte si, že vývody 4 a 5 zůstávají volné, takže je lze použít pro telefon (analogový).

Obr. 3.28
Zapojení vývodů pro 10BASE-T (resp. 100BASE-TX)



V konektoru RJ45 se používají pro Ethernet dva páry. Jeden pár pro vysílání, druhý pár pro příjem. V případě, že ethernetový segment sdílejí pouze dvě stanice, které jsou propojeny přímo propojovacím kabelem, pak musí být páry překříženy (tj. překřížen příjem s vysíláním) – viz obr. 3.29.

Obr. 3.29
Propojovací kabely pro TP



Segment tvořený pouze dvěma stanicemi je velice zajímavým segmentem. Síťová rozhraní se na takovémto segmentu přepínají do plně duplexního provozu (***Full Duplex***), kdy se zcela oddělí vysílání od příjmu. Na takovémto segmentu nemůže dojít ke kolizím (vysílání je přímo napojeno na příjem a nikdo třetí do spojení nemůže vstoupit), proto zde lze dosáhnout přenosových rychlostí blížících se teoretickému maximu (10 Mb/s pro Ethernet a 100 Mb/s pro Fast Ethernet) a to samostatně v každém směru! Na tomto principu je založen tzv. přepínaný Ethernet.

Ethernet na optických vláknech se označuje též jako **10BASE-F**. Zásadně se vždy používá pár optických vláken – pro každý směr komunikace jedno vlákno.

3.4.3 Fast Ethernet (100 Mb/s)

Fast Ethernet se připojuje kroucenou dvojlinkou (označení 100BASE-TX) nebo optickým spojem (označení 100BASE-FX). Rozdíl oproti klasickému Ethernetu je pouze v kvalitě vedení. Současné rozvody se většinou staví minimálně kategorie 5, takže nasazení Fast Ethernetu jim nečiní potíže.

3.4.4 Gigabitový Etherent (1 Gb/s)

Gigabitový Etherent je standardizován pro optické spoje a pro kroucenou dvojlinku (4 páry).

Pro jednovidová vlákna je určen standard pod označením 1000BASE-LX buzený laserem o frekvenci 1300 nm s maximální délkou segmentu do 2 km (jednovidová vlákna na plně duplexních segmentech až do 40 km). Pro vícevidová vlákna může týž stadard (1000BASE-LX) pracovat až do vzdíálenosti 450 m.

Pouze pro vícevidová vlákna je určen standard 1000BASE-SX, který je buzen laserem o frekvenci 850 nm a je určen pro vzdálenosti do 250 m.

Standard pro metalické spoje 1000BASE-CX bude využívat současných rozvodů kategorie 5+ (100 MHz), avšak využije všechny čtyři páry kroucené dvojlinky (tj. všech 8 vývodů konektoru RJ 45).

3.4.5 FDDI

FDDI existují dvě varianty: na optickém vlákně (FDDI) nebo na kroucené dvoulince (CDDI). Na jedné LAN je možné obě eventuality i kombinovat. Přednost se dává kroucené dvojlince a pro připojení vzdálenějších uzlů se použije světelné vlákno. Vývody opět zpravidla vedou na distribuční box optiky v případě optických rozvodů a na propojovací panel v případě měděných rozvodů.

U FDDI je nutné poznamenat, že do počítače se vkládají dva typy karet. Nemyslím tím nyní pro kroucenou dvojlinku nebo pro optické vlákno, ale podle toho mají-li jedno nebo dvě rozhraní pro LAN. Karty se dvěma páry vývodů jsou určeny pro počítače, které jsou umístěny přímo na páteřním kruhu FDDI. Karty s jedním párem vývodů jsou určeny pro počítače, které budou připojeny přes koncentrátor. Koncentrátor je pak připojen na páteřní kruh.

3.5 GSM

Dnes je již více uživatelů sítě GSM než uživatelů Internetu. Uživatelé GSM se často v souvislosti s Internetem ptají na dvě otázky:

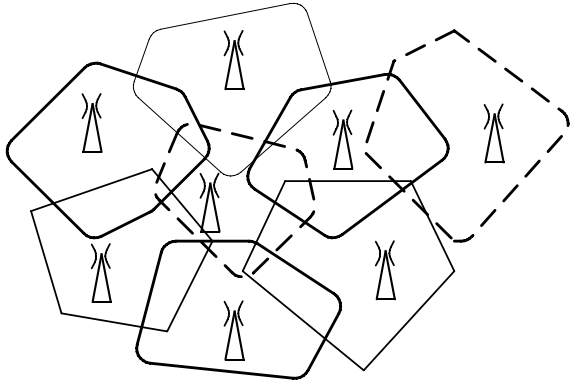
- Jak využít síť GSM pro připojení počítače k Internetu.
- Jak přímo přistupovat k informacím v Internetu pomocí mobilního telefonu, tj. např. jak mobilním telefonem brouzdat po Internetu.

Odpovědím na tyto otázky je věnována tato kapitola.

Normy pro GSM vydává ETSI (*the European Telecommunications Standards Institute*). Bližší informace jak zakoupit tyto normy např. na CD lze nalézt na http://www.etsi.org.

Systém GSM pokrývá území. Území je tak rozděleno do řady buněk. Každá buňka je obsluhována jedním zařízením BTS (*Base Transceiver Station*). Jednotlivé buňky se vzájemně překrývají jak je znázorněno na obr. 3.30. Pokud se uživatel se svým mobilním telefonem pohybuje, pak si jej jednotlivé BTS postupně předávají.

Obr. 3.30
Pokrytí území vzájemně překrývajícími se buňkami

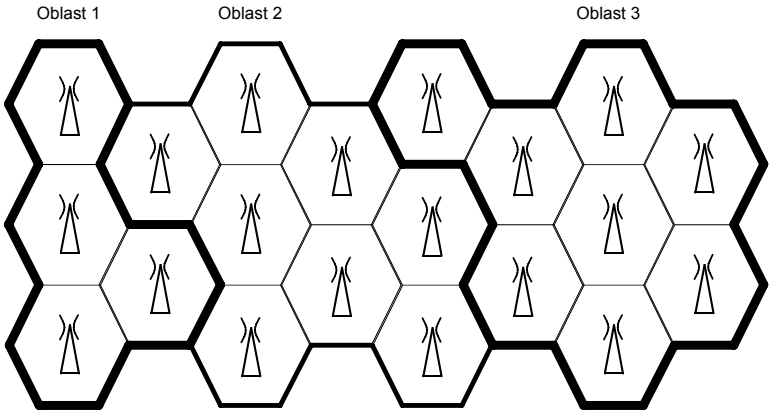


Nadále si však budeme rozdělení území na jednotlivé buňky představovat zjednodušeně jak je znázorněno na obr. 3.31.

Z hlediska řízení sítě je nutné udržovat informaci o tom, kde se konkrétní uživatel nachází. Z tohoto pohledu je vždy několik buněk začleněno do oblasti. Síť si udržuje informaci ve které oblasti se uživatel nachází. Je-li třeba vyhledat uživatele v síti (např. pro příchozí hovor), pak je vyhledáván ve všech buňkách dané oblasti.

Základním problémem pro takovéto množství vysílačů je počet přidělených vysílacích frekvencí. Díky omezenému rozsahu je možné po určité vzdálenosti opět použít stejné frekvence. Jestliže pro celou síť je alokováno N frekvencí, pak je možné stejné frekvence používat ve vzdálených buňkách. Cca N/9 frekvencí je možné použít v jedné buňce.

Obr. 3.31
Zjednodušené schéma pokrytí znázorňující oblasti a buňky



GSM používá dvě frekvence:

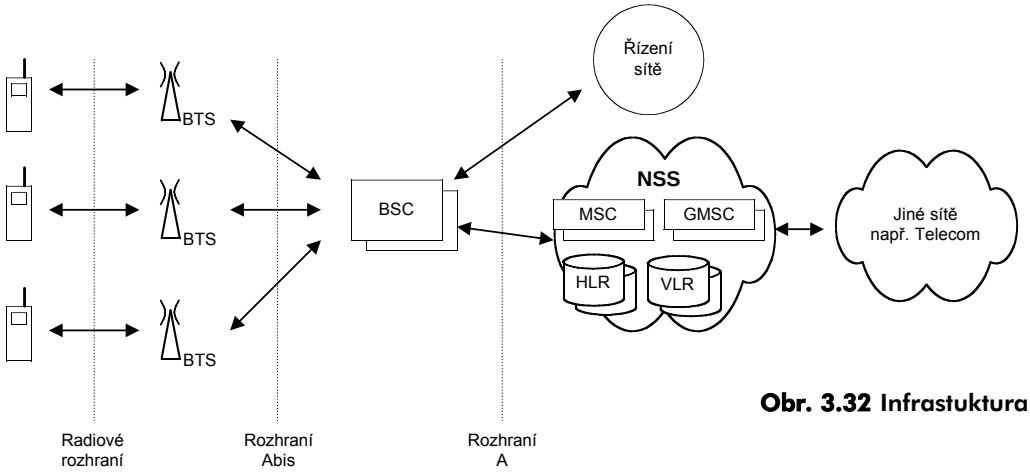
- Primární frekvence 900 MHz, kdy je operátorovi přiřazen rozsah zpravidla o šířce 25 MHz, a to buď 890-915 MHz nebo 935 až 960 MHz.
- Sekundární frekvence 1800 MHz, která používá zpravidla také dva rozsahy, a to 1710-1785 MHz a 1805-1880 MHz. Šířka rozsahu je tak 75 MHz, tj. třikrát větší než u frekvence 900 MHz.

Rozsah je pak rozdělen po 200 kHz, které se používají pro vysílače v BTS i v mobilních telefonech. Primární frekvence tak teoreticky obsahuje 124 frekvencí. Jelikož se krajní frekvence zpravidla nepoužívají, tak obsahuje reálně 122 frekvencí.

V jedné buňce je tedy možné použít 122/9 frekvencí, což je 13. V praxi se tak používají BTS s jedním, čtyřmi či maximálně 12 vysílači (frekvencemi).

Infrastruktura GSM znázorněná na obr. 3.32 se skládá z:

- Radiového rozhraní, pomocí kterého komunikuje mobilní telefon s BTS.
- BTS (*Base Transceiver Station*)
- BSC (*Base Station Controler*), který řídí několik BTS.
- NSS (*Network and Switching Subsystem*) zabývající se přepínáním hovorů (okruhů). Každý hovor i v rámci jedné buňky je přepínán NSS. NSS může přepínat okruhy i do jiných sítí, používá signalizaci SS7, o které jsme se zmínili u ISDN. NSS se skládá z:
  - MSC (*Mobile services Switching Centre*), které řídí několik BSC.
  - HLR (*Home Location Register*) což je databáze uživatelů, kde jsou uloženy informace o uživatelích jako je jméno uživatele, poskytované služby atd. Součástí HLR je též Autentizační centrum, které provádí autentizaci uživatelů.
  - VLR (*Visitors Location Register*) obsahuje databázi cizích uživatelů (uživatelů, kteří nejsou v HLR)
  - GMSC, což je gateway, na kterou jsou směrovány příchozí hovory. Tato gateway nalezne informace o uživateli v HLR a přepojí hovor na konkrétní MSC.
- Řízení sítě



Obr. 3.32 Infrastuktura GSM

Pro komunikaci mezi mobilním telefonem a BTS se používají komunikační kanály. Základnám kanálem, který používá uživatel pro komunikaci je kanál TCH (*Traffic Channel*). Rozeznáváme tři typy kanálu TCH:

1. Kanál TCH/F (F znamená *full rate* pro „plnou rychlost“).
2. Kanál TCH/H (H znamená *half rate* pro „poloviční rychlost“).
3. Kanál TCH/8 (1/8 rychlosti).

Kanálu TCH/F i kanálům TCH/H a TCH/8 je vždy ještě přiřazen pomalý kanál SACCH. Tímto pomalým kanálem mohou být přenášeny cca 2 zprávy za vteřinu nezávisle na kanálu TCH/F (resp. TCH/H).

Každá vysílací frekvence je časově rozdělena do osmi slotů (tj. jedna vysílací frekvence může být rozdělena mezi 8 uživatelů – např. mezi 8 kanálů TCH/F). Každý slot může přenášet jeden kanál TCH/F, tj. hlas frekvencí 13 kb/s či data rychlostí do 12,6 kb/s. Jak je uvedeno v kap. 3.3.1.1, do synchronních 12,6 kb/s se kóduje standardních 9,6 kb/s asynchronních.

Do příslušného slotu pro kanál TCH/F se „vejde“ ještě kanál SACCH. Oba kanály tak tvoří jeden spojený (asociovaný) kanál, který se označuje TACH/F. Obdobná situace je i u kanálu TCH/H, kde vznikne spojený kanál TACH/H.

Pokud by se (teoreticky) nepoužívaly žádné služební kanály, tak by bylo možné, aby se o jednu frekvenci dělilo až 8 uživatelů, tj. frekvence byla rozdělena mezi 8 hovorů.

Kromě kanálů TACH/F, TACH/H sloužících pro přenos uživatelských informací používá GSM několik služebních kanálů:

- Synchronizační kanál SCH a kanál frekvenční korekce FCCH. Oba kanály zajišťují pomocí oběžníků synchronizaci v rámci buňky (nezapomínejme, že GSM používá synchronní komunikaci).
- Kanál BCCH (*Broadcast Control Channel*) signalizuje buňku. Každá buňka se indikuje kanálem BCCH. Uživatel, který se pohybuje tak může zjišťovat informace pro volbu buňky (uživatel tak může zjistit i přítomnost buňky cizí sítě). Tento kanál je zpracováván mobilním telefonem i když je mobilní telefon nečinný.

- Kanálem PAGCH (*Paging and Access Channel*) signalizuje síť příchozí hovor pro mobilní telefon v oblasti. Signalizace se provádí oběžníky v rámci oblasti. Součástí této signalizace je i přiřazení kanálu pro komunikaci.
- Kanál RACH (*Random Access Channel*) slouží pro komunikaci z mobilního telefonu do sítě (všechny předchozí kanály odesílaly oběžníky směrem od BTS k mobilnímu telefonu). Tento kanál se používá např. v případě, že uživatel z mobilního telefonu chce vytvořit okruh („chce telefonovat"). Kanál RACH je kanál s náhodným přístupem, což s sebou přináší potenciální možnost kolizí.
- Kanál CBCH (*Cell Broadcast Channel*) používá mobilní telefon, když je nečinný. V takovém případě musí mobilní telefon odeslat zprávu dlouhou cca 80 bajtů kanálem CBCH každé dvě minuty, aby síť věděla, že je telefon k dispozici.

Služební kanály FCCH, SCH, BCCH a PAGCH lze zkombinovat tak, že zaberou část jednoho slotu ve směru od BTS k mobilnímu telefonu, tj. zaberou část pásma jako jeden kanál TACH/F.

Jak již bylo zmíněno buňka GSM je obsluhována BTS, která může obsahovat 1, 4 nebo 12 transceiverů. Zpravidla se používá následující organizace kanálů v buňce:

- V případě jednoho vysílače (celkem 8 slotů):
  - Jeden slot pro kanály: FCCH, SCH, BCCH, PAGCH, RACH a 4x TACH/8
  - Sedm slotů pro TACH/F, tj. buňka může obsloužit až 7 hovorů současně.
- V případě čtyř vysílačů (celkem 4x8=32 slotů):
  - Jeden slot pro kanály: FCCH, SCH, BCCH, PAGCH, RACH
  - Dva sloty po 8x TACH/8
  - 29 slotů pro TACH/F, tj. buňka může obsloužit až 29 hovorů současně.
- V případě dvanácti vysílačů (celkem 12x8=96 slotů)
  - Jeden slot pro kanály: FCCH, SCH, BCCH, PAGCH, RACH
  - Tři sloty pro kanály: BCCH, PAGCH a RACH
  - Pět slotů pro 8x TACH/8
  - 87 slotů pro TACH/8, tj. buňka může obsloužit až 87 hovorů současně.

3.5.1 Připojení počítače k Internetu přes datový okruh GSM

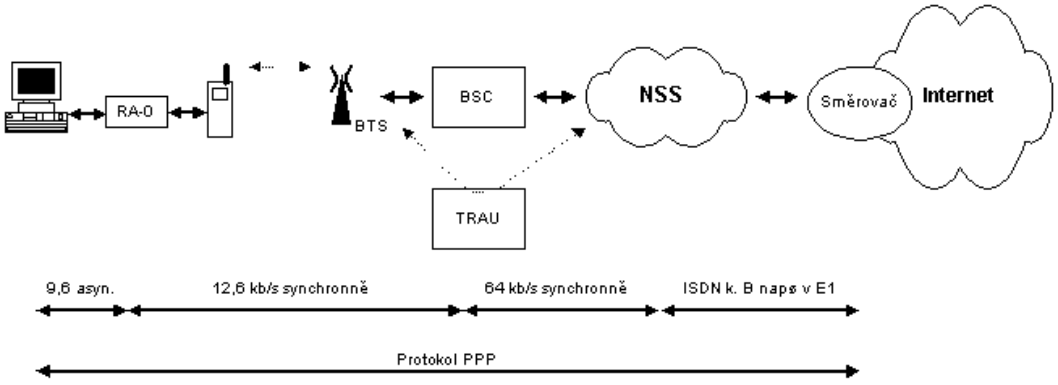
Dnes se používají pro připojení počítače k Internetu přes síť GSM datové služby GSM. Tj. kanálem TCH/F se přenášejí data.

Počítač je s mobilním telefonem propojen pomocí zařízení RA-0, které je buď součástí mobilního telefonu nebo součástí počítače. Zařízení RA-0 převádí asynchronní signál rychlosti až 9,6 kb/s na synchronní signál 12,6 kb/s, který je vložen do kanálu TCH/F.

Jelikož NSS přepojuje okruhy o rychlosti 64 kb/s („ISDN okruhy"), tak je třeba zařízením TRAU (*Transcoder/Rate Adapter Unit*) doplnit signál 12,6 kb/s výplní na signál 64 kb/s (viz kap. 3.3.1.1). Zařízení TRAU se vkládá před BSC, za BSC nebo může být i součástí zařízení BSC (jak je znázorněno na obr. 3.33).

Signál z NSS přichází jako jeden ISDN B kanál na směrovač poskytovatele Internetu. Poskytovatel Internetu je zpravidla propojen s NSS více kanály B, tj. např. linkou E1 či E3, aby mohlo velké množ-

ství uživatelů současně přistupovat do Internetu. Pro sestavení virtuálního okruhu na směrovač poskytovatele Internetu je nutné, aby směrovač měl přiděleno své telefonní číslo, protože v telefonní síti jsou účastníci adresováni telefonním číslem (v tomto případě je účastníkem směrovač poskytovatele).



Obr. 3.33 Využití datových služeb mobilního telefonu pro připojení PC k Interentu

Pokud se chce počítač připojit k Internetu, tak je nutné, aby předal mobilnímu telefonu telefonní číslo směrovače. Telefon pak požádá GSM síť o vytvoření datového okruhu se směrovačem poskytovatele Internetu. V takto vytvořeném virtuálním okruhu je možné, aby počítač komunikoval se směrovačem linkovým protokolem. Zpravidla se zde používá protokol PPP, kterým se uživatel i autentizuje poskytovateli Internetu.

3.5.2 SMS

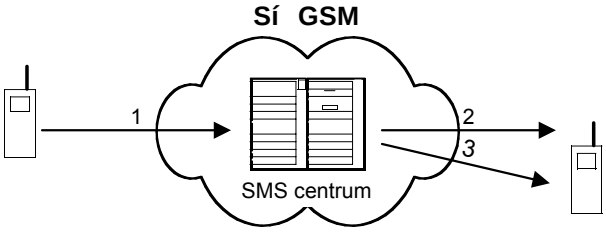
Komunikace pomocí krátkých textových zpráv (SMS zpráv) mezi uživateli sítě GSM je služba připomínající e-mail v Internetu. Z hlediska uživatele je zásadním rozdílem mezi zprávou SMS a e-mailem skutečnost, že zpráva SMS může být maximálně 160 znaků dlouhá. (Uvažuje se i o rozšířených SMS zprávách, kde jednu delší logickou zprávu bude možné přenášet pomocí několika fyzických zpráv dlouhých 160 znaků.)

Další rozdílnou vlastností SMS zpráv je, že u nich je možná tzv. notifikace přijetí zprávy, tj. je možné označit zprávu tak, aby ji síť GSM sledovala, zdali byla zpráva doručena adresátovi nebo nikoliv.

Zmínili jsme se o vlastnostech, kterými se zprávy SMS liší od e-mailu. Nyní je třeba naopak objasnit vzájemnou podobnost. U elektronické pošty není nutné, aby byli odesílatel i adresát současně připojení k Internetu. Odesílatel odešle e-mail, který Internetem doputuje až na poštovní server adresáta, zde se uloží a čeká, až se adresát připojí k Internetu a e-mail si vyzvedne.

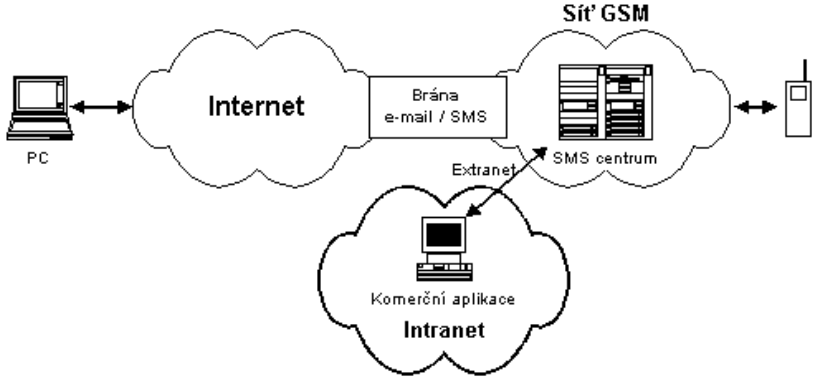
Zprávy SMS putují od adresáta na tzv. SMS centrum (obdoba poštovního serveru). SMS centrum se snaží doručit zprávu adresátovi, pokud není možné navázat spojení s adresátem, tak zpráva zůstává uložena v SMS centru (obdobně jako e-mail na poštovním serveru).

Obdobná je i adresa adresáta SMS zprávy. Např. v Internetu má adresát e-mailovou adresu *novak@firma.cz*, kde *firma.cz* určuje poštovní server a řetězec *novak* určuje konkrétního uživatele v rámci tohoto poštovního serveru. U SMS zpráv máme místo názvu poštovního serveru telefonní číslo SMS centra a místo jména uživatele se používá číslo uživatele.



Obr. 3.34
Odeslaná SMS zpráva se uloží v SMS centru (1); SMS centrum se pokouší doručit zprávu adresátovi (2), pokud se doručení nepodaří, tak zpráva zůstává v SMS centru; Při dalším pokusu (3) se zprávu podařilo doručit

Mnozí operátoři GSM poskytují pro nekomerční použití SMS bránu do Internetu (viz obr. 3.35). Použití této brány je jednoduché. Např. do sítě Paegas se z Internetu odešle e-mail na adresu *tel.cislo@sms.paegas.cz* (uživatel si přes WWW server operátora může tuto službu povolit či zakázat). Tento e-mail je Internetem dopraven na SMS bránu, která jej transformuje na SMS zprávu a odešle do SMS centra, které se pokouší SMS zprávu doručit adresátovi. Není třeba asi zdůrazňovat, že při transformaci e-mailu na SMS zprávu lze z e-mailu vzít pouze 160 bajtů. Je možná komunikace i v opačném



Obr. 3.35
SMS brána a komunikace SMS centra s komerčními aplikacemi.

směru, tj. SMS zpráva se doplní o internetovou adresu příjemce a odešle se do SMS centra, které zajistí předání zprávy SMS bráně.
Pokud je třeba odesílat velké množství SMS zpráv z počítače do sítě GSM (zejména pokud se jedná o komerční aplikace), pak se využije skutečnosti, že SMS server je také počítač. Mezi oběma počítači se vybuduje spojení a aplikace předává/přebírá SMS zprávy přímo do/z SMS centra. Jelikož se jedná o propojení intranetu firmy s intranetem operátora GSM, tak se kladou velké nároky na bezpečnost tohoto propojení.

Důležitou vlastností SMS komunikace je skutečnost, že SMS zprávy mohou být přenášeny nezávisle na hlasové/datové komunikaci uživatele mobilního telefonu, tj. uživatel může současně telefonovat a nezávisle na tom přenášet SMS zprávy.

Pro SMS komunikaci se nesestavuje virtuální okruh. SMS zpráva se předá GSM síti, která ji samostatně dopravuje adresátovi.

Příkladem využití SMS zprávy v aplikaci je služba PaegasInfo, kde adresátem SMS zprávy je aplikace (program). Pokud odešlete správně formátovanou zprávu SMS centru +420603052000 na telefonní číslo 4616 (v internetové terminologii bychom napsali odeslat „mail“ na 4616@+420603052000, protože 4616 je obdobou poštovní schránky na serveru +420603052000). Tak tuto zprávu zpracuje program, který odpoví též SMS zprávou.

Např. pro překlad slova „by“ z angličtiny do češtiny odešlete SMS zprávu: `SLO AC by`

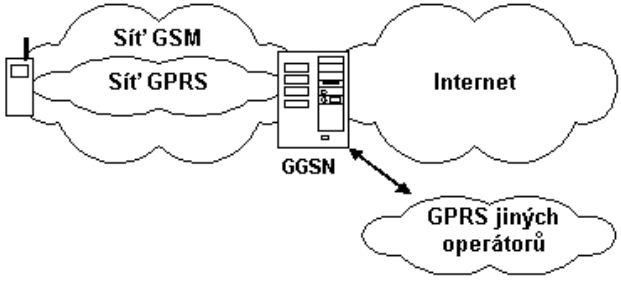
3.5.3 GPRS

Datový okruh má dvě zásadní nevýhody:

- Má poměrně malou přenosovou rychlost – pouze do 9,6 kb/s.
- Na počátku se musí okruh sestavit, což jistou dobu trvá. To je nepraktické např. při brouzdání po Internetu, kdy uživatel si stáhne z Internetu nějaké informace, které si čte a za chvíli stahuje další informace. Mezi tím je úsek, kdy není třeba žádná datová komunikace. Buďto by se datový okruh v tomto úseku přerušil, ale pak zase je nutné jej po chvíli opět sestavit; nebo by okruh byl ponechán, pak se ale musí za něj zbytečně platit, i když není využíván.

Tyto nevýhody odstraňuje GPRS (*General Packet Radio Service*). GPRS nesestavuje virtuální okruhy, ale používá paketový přenos podobně jako Internet. Mobilní telefon se tak jeví jako by byl neustále připojen k síti. GPRS by se tak dalo přirovnat k připojení počítače k LAN (naopak využití datového okruhu lze přirovnat k připojení počítače přes modem).

Obr. 3.36
GPRS tvoří datovou síť uvnitř sítě GSM



GPRS teoreticky může k přenosu dat využít až všech 8 slotů a dosáhnout teoretické přenosové rychlosti až 171,2 kb/s. V praxi se však předpokládá, že budou využity maximálně 4 sloty. Předpokládá se, že se nejprve bude používat rychlost cca 28 kb/s, později rychlost cca 56 kb/s a v budoucnu rychlost 112 kb/s (viz http://www.gsmworld.com).
Jak je znázorněno na obr. 3.36, tak GPRS tvoří datovou síť uvnitř sítě GSM. Tato síť je *pomocí Gateway GPRS Service Node* (GGSN) propojena s ostatními sítěmi – např. Internetem nebo GPRS jiných operátorů.

Pro komunikaci v Internetu může mít mobilní telefon přiřazenu IP-adresu. Mobilní telefon pak vkládá IP-datagramy do GPRS. Uzel GGSN pak pracuje jako směrovač.

3.5.4 SIM

GSM je založen na myšlence, že každý mobilní telefon se skládá ze dvou částí:

- Mobilního telefonu
- SIM (*Subscriber Identity Module*) – což je čipová karta vložená do mobilního telefonu (viz obr. 3.37). SIM karta je v podstatě počítač řídící mobilní telefon. SIM karta kromě procesoru obsahuje paměť RAM a trvalou paměť (s uloženým softwarem, ale i např. telefonním seznamem).

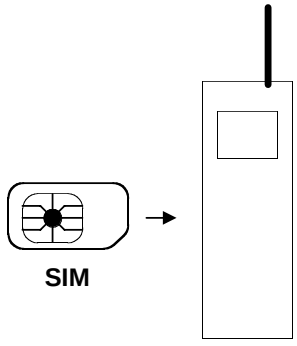
SIM karta obsahuje osobní údaje klienta (telefonní seznam, seznam posledních volání atd.); SIM karta neobsahuje osobní údaje o klientovi (např. jméno uživatele) – ty jsou uloženy v databázi HLR.

Po zapnutí mobilního telefonu je uživatel autentizován pomocí PIN. Po zadání správného PIN jsou uživateli zpřístupněny informace na SIM kartě a mobilní telefon se pokusí vyhledat příslušnou síť GSM.

Každá SIM karta má svou devatenáctimístnou identifikaci ICCID (je nejenom uložena v SIM kartě, ale bývá i na kartě vytištěna). Tato identifikace je přístupovým klíčem do databáze HLR. Zajímavé je, že na

SIM kartě není trvale uloženo telefonní číslo uživatele či seznam poskytovaných služeb, to se získá až z HLR. Toto zdánlivě komplikované řešení přináší uživatelům velké výhody. Např. pokud si chce uživatel nechat změnit telefonní číslo či přikoupit další služby, pak se to udělá pouze změnou v HLR. Prakticky to znamená, že pokud chce uživatel změnit telefonní číslo, tak nemusí nikam chodit se SIM kartou, ale mnohdy stačí zavolat na Call Centrum operátora, kde mu tuto změnu provedou telefonicky (uživatel musí pouze svůj mobilní telefon vypnout a znovu zapnout, aby se z HLR načetlo nové číslo).

Obr. 3.37
SIM karta se vkládá do mobilního telefonu



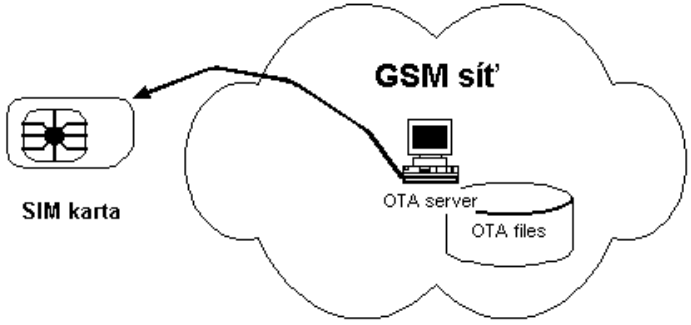
Software na SIM kartě je zodpovědný nejenom za identifikaci uživatele, přihlášení uživatele k síti, ale např. i za zobrazovaná menu na displeji mobilního telefonu. Škála menu mobilního telefonu nám dává škálu poskytovaných služeb. Praktická by byla možnost modifikace menu mobilního telefonu. Jenže modifikace menu mobilního telefonu znamená modifikaci softwaru uloženého na SIM kartě, protože za jednotlivými položkami menu se ukrývají jednotlivé funkčnosti, které jsou realizované softwarem uloženým v SIM kartě.

Pro modifikaci některých datových oblastí (např. oblasti s telefonním seznamem nebo programů realizujících některá menu) slouží systém OTA (*Over The Air*). Jak je znázorněno na obr. 3.38, v GSM síti operátora GSM je umístěn OTA server, na kterém jsou uloženy tzv. OTA soubory, které je možné nahrávat do SIM karet.

Celý mechanismus je zabezpečen, aby nebylo útočníkům umožněno nahrávat software do cizích SIM karet.

Jelikož při nahrávání do SIM karty slouží mobilní telefon pouze jako čtečka čipových karet, tak je možné protokol OTA využít lokálně na PC s čtečkou čipových karet. Uživatel přinese svou SIM kartu k PC se čtečkou čipových karet. Na PC se spustí program, který nahraje protokolem OTA soubor z lokálního disku do SIM karty. I tento zjednodušený postup využívá zabezpečení protokolu OTA. Na SIM kartu je možné data nahrát až poté, co uživatel zadá svůj PIN (zadáním PIN dává uživatel svolení k nahrání dat). Dalším typem zabezpečení je, že nahrávaná data musí být „digitálně podepsána“ operátorem GSM – jiná data SIM karta neakceptuje.

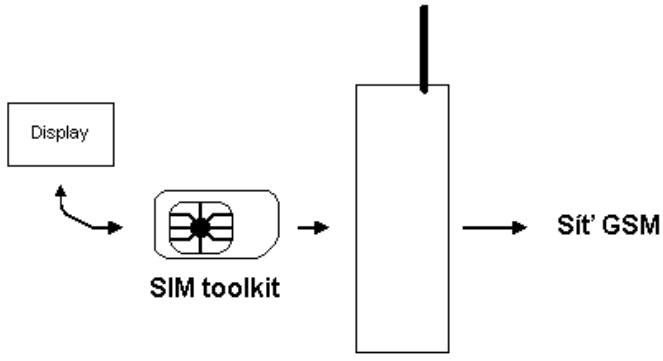
Obr. 3.38
Download softwaru z OTA serveru do SIM karty



Nyní už víme, jak software na SIM kartu nahrát. Ale otázkou je, jak takový software vyvinout. K tomuto účelu lze využít SIM toolkit, což je aplikační programové rozhraní, pomocí kterého lze využívat (upravovat) některé funkčnosti mobilního telefonu.

Nejlepším způsobem jak si představit funkčnost SIM toolkitu je představit si jej jako černou skříňku mezi displejem (s klávesnicí) mobilního telefonu a zbytkem mobilního telefonu, jak je znázorněno na obr. 4.39.

Obr. 3.39
SIM toolkit



V kapitole 3.5.2 jsme překládaly slovo „by“ z angličtiny do češtiny pomocí služby PaegasInfo. Tato služba měla nevýhodu v tom, že jsme si museli pamatovat klíčové slovo „SLO“. Pomocí SIM toolkit je možné v mobilním telefonu vytvořit menu 'PaegasInfo' -> 'Praktické' -> 'Slovníček', které vygeneruje správně formátovanou zprávu, do které pouze vložíme vstupní data (do jakého jazyka překládat a překládané slovo) – nemusíme se tak zabývat správnou strukturou zprávy – to obstará SIM toolkit.

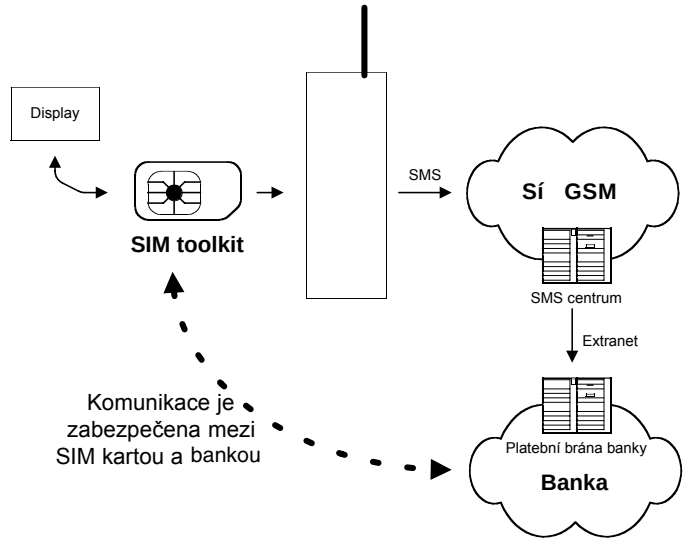
Klasickou aplikací, na které lze velice pěkně objasnit funkci SIM toolkitu je populární úloha IPB GSMbanking. Klient chce odeslat platební příkaz ze svého mobilního telefonu do banky. Jako nosič použije SMS zprávu jejíž délka tomuto účelu postačuje. SMS zpráva přenáší textový řetězec. Pokud by uživatel pořizoval platební příkaz jako běžnou textovou zprávu, pak by musel na přesná místa v přenášeném řetězci uvést čísla účtu, kódy bank, částku a další informace, které se uvádějí v platebním příkazu. To by bylo velice nepohodlné, takže se využije funkčnost SIM toolkitu, která umožňuje vybudovat soustavu menu tak, že uživatel je postupně dotazován na číslo účtu příkazce, číslo účtu příjemce, částku atd. SIM toolkit následně sám uloží tyto údaje do SMS zprávy na správné pozice.

To však není vše. SIM toolkit datovou část SMS zprávy před odesláním zašifruje (zašifrování bychom bez SIM toolkitu těžko prováděli). Šifrovací klíč je uložen na SIM kartě a pochopitelně ve vaší bance, takže nikdo se k obsahu zprávy během jejího přenosu GSM sítí nedostane (ani zaměstnanci operátora GSM). Přístup k šifrovacímu klíči na SIM kartě je chráněn dalším PINem, tzv. bankovním PINem (nebo BPINem) – viz obr. 3.40.

Obdobný systém se používá i pro jiné aplikace, jako je např. prodej letenek. Na tomto systému je oceňovaná zejména bezpečnost, která využívá zabezpečení mezi SIM kartou a koncovou aplikací. Takovéto zabezpečení např. neposkytuje ani dnes aktuální verze 1.1 protokolu WAP.

Jiným typem aplikace je platba v telefonickém virtuálním obchodním domě pomocí elektronické peněženky realizované na čipové kartě. Tento systém je určen pro mobilní telefony vybavené druhým slotem pro čtení čipových karet, tzv. *dual slot*. Zatímco do prvního slotu mobilního telefonu se vkládá SIM karta, tak do druhého slotu lze vložit jinou čipovou kartu, např. právě elektronickou peněženku. (Je vcelku škoda, že dnes nelze do mobilního telefonu vložit platební kartu (např. VISA či MasterCard/EuroCard), protože tyto karty jsou realizovány na magnetickém proužku a jejich přechod na čipovou kartu (resp. hybridní kartu, tj. kartu obsahující jak magnetický proužek, tak i čip) se teprve plánuje.

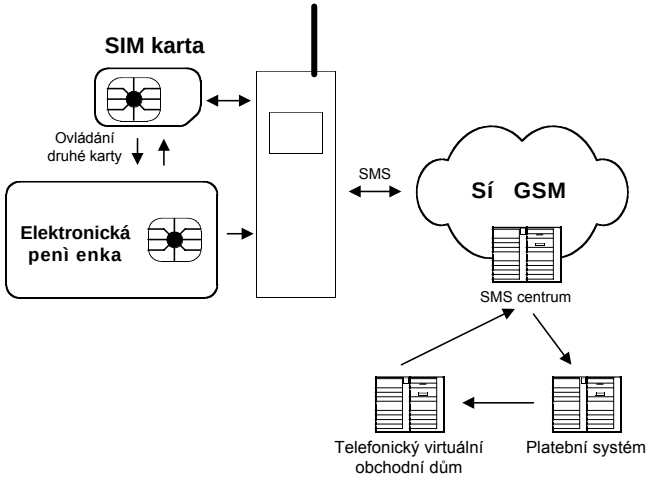
Obr. 3.40
Zabezpečení komunikace za využití SIM toolkit



Uživatelská představa je taková, že uživatel jede autem a vzpomene si, že chce koupit nějaké zboží. Zatelefonuje do telefonického virtuálního obchodního domu, kde si zboží vybere. V zápětí zboží zaplatí z elektronické peněženky vložené do druhého slotu mobilního telefonu. Zboží je pak již jen dodáno na určené místo.

Důležitý je systém platby. Z telefonického virtuálního obchodního domu přijde požadavek na platbu elektronickou peněženkou (viz obr. 3.41). Aplikační programové rozhraní umožňuje vyvinout a na SIM kartu nahrát software, který umí komunikovat s vloženou čipovou kartou (elektronickou peněženkou) tak, že elektronická peněženka vygeneruje ve vhodném okamžiku příslušnou platbu. Tato platba je zašifrována a vložena do SMS zprávy. SMS zpráva s platbou doputuje přes SMS centrum do platebního systému, který zpravidla provozuje vydavatel elektronických peněženek. Nakonec platební systém potvrdí platbu telefonnímu virtuálnímu domu a jeho operátorka může uživateli poděkovat za projevený zájem.

Obr. 3.41
Platba elektronickou peněženkou



3.5.5 WAP

V předchozím příkladu jsme nakupovali v telefonickém virtuálním obchodním domě. Podstatně zajímavější je brouzdat po Internetu s případnou možností nákupu v internetovém obchodním domě.

Protokol WAP (*Wireless Application Protocol*) umožňuje uživatelům mobilních telefonů přistupovat k informacím a službám umístěným na Internetu (resp. intranetu), aniž by k tomu potřebovali počítač – místo počítače použijí mobilní telefon. WAP je pro mobilní telefony to, co WWW pro počítače. Takže pokud máte k dispozici mobilní telefon podporující WAP, tak se již při brouzdání po Internetu můžete obejít bez PC. V mobilním telefonu je implementován tzv. WAP microbrowser, který je analogií webového prohlížeče pro PC. Vzhledem k omezeným zobrazovacím možnostem mobilního telefonu má microbrowser pouze omezené možnosti oproti browseru na PC.

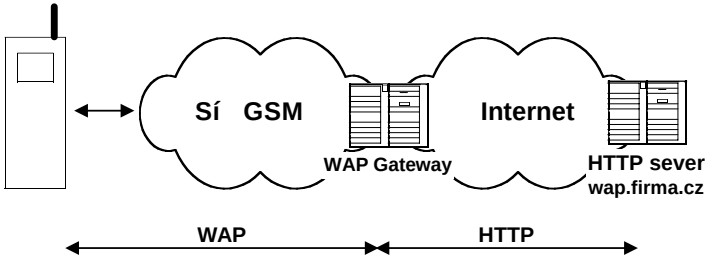
Vývojem standardu pro WAP se zabývá WAP Forum (viz http://www.wapforum.org).

WAP používaný v sítích GSM může použít jako přenosové médium:

- SMS pouze teoreticky, protože SMS je pomalou komunikací, která je pro tento účel poměrně drahá. WAP přes SMS lze přirovnat k WEBmailu, tj. stahování webových stránek přes e-mail.
- Datový okruh GSM, který lze přirovnat k připojení PC přes modem k Internetu. Datový okruh je v NSS vždy převeden do tvaru ISDN, proto se o tomto typu připojení někdy říká, že se jedná o připojení typu ISDN.
- GPRS, který lze přirovnat k připojení PC k Internetu pomocí pevné linky.

Základem WAP komunikace je WAP gateway (viz obr. 3.42). Uživatel balí informace ve WAP protokolu do SMS zprávy resp. datového okruhu resp. paketů GPRS a odesílá je na WAP gateway. WAP gateway tyto informace vybalí a na aplikační vrstvě je transformuje do protokolu HTTP. Pakety HTTP protokolu vkládá do TCP spojení v Internetu, které je navázáno s příslušným webovým serverem. Webový server pak vrací příslušné webové stránky zpět přes WAP gateway do microbrowseru mobilního telefonu. Microbrowser pak informace interpretuje na displeji mobilního telefonu.

Obr. 3.42
Komunikace s HTTP serverem



Na HTTP serveru se vystavují stránky v jazyce WML, který je jakýmsi zjednodušením jazyka HTML. Můžeme také prohlížet stránky v jazyce HTML, neboť WAP gateway zpravidla umí i konverzi z HTML do WML, avšak to není příliš praktické, proto většina firem má kromě http://www.firma.cz i http://wap.firma.cz. Nepraktičnost spočívá v tom, že firmy mají http://www.firma.cz vyšperkované mnohými velkými obrázky a jinými komplikovanými konstrukcemi, které jsou sice pěkné v nejposlednější verzi prohlížeče na PC, ale v microbrowseru nepůsobí nikterak úchvatně.

Celková architektura protokolu WAP se skládá z pěti vrstev a vrstvy nosičů, jak je znázorněno na obr. 3.43. Jako nosiče mohou sloužit protokoly GSM, ale WAP obecně není omezen jen pro GSM, ale může být použit nad téměř libovolnou sítí.

Obr. 3.43
Vrstvy protokolu WAP

Transportní vrstva WDP (*Wireless Datagram Protocol*) je obecným datagramovým protokolem, jehož datagramy mohou být vkládány do téměř libovolného nosiče.

Transakční vrstva WTP (*Wireless Transaction Protocol*) zabezpečuje spojení (je spojovanou službou), tj. zabezpečuje doručování dat.

Relační vrstva WSP (*Wireless Session Protocol*) zabezpečuje funkčnost na úrovni protokolu HTTP verze 1.1. Relační vrstva dokáže pracovat nejen nad vrstvou WTP, ale dokonce i pouze nad vrstvou WDP.

Aplikační vrstva WAE (*Wireless Application Envirnoment*) definuje:

- Manipulační jazyk WML (*Wireless Markup Language*) pro popis WAP stránek, který je obdobou jazyka HTML.
- WML skripty, které jsou obdobou (zjednodušením) Java skriptů.
- WTA služby (*Wireless Telephony Application*) a programovací rozhraní pro ovládání těchto služeb.
- Formáty přenášených dat, obrázků, telefonních seznamů a kalendářů.

Bezpečnostní vrstva WTLS (*Wireless Transport Layer Security*) je vcelku nešťastně vložena mezi transportní a transakční vrstvu. Bezpečnostní vrstva WTLS je odvozena od protokolu TLS. Protokol TLS je obdobou v Internetu populárního protokolu SSL. I když protokoly TLS a SSL jsou velice podobné, tak jsou vzájemně nekompatibilní (tj. např. čistý klient TLS nedokáže komunikovat se serverem SSL). Když se k tomu připočte ještě vložení bezpečnostní vrstvy mezi transportní a transakční vrstvu (nikoliv až pod aplikační vrstvu, jak je tomu v případě SSL), tak z toho vyplývá, že WAP gateway nemůže pracovat obdobně jako SSL proxy, jak jsme zvyklí u protokolu HTTPS. Jinými slovy: pokud se použije protokol WTLS, tak je možné zabezpečení pouze mezi mobilním telefonem a WAP gateway. Na WAP gateway se vše musí dešifrovat. Případné zabezpečení mezi WAP gateway a WWW serverem (aplikací) je možné např. protokolem SSL, ale jako klient v tomto zabezpečení nevystupuje mobilní telefon, ale celá WAP gateway, která použije jeden společný klíč pro všechny své uživatele.

Jediným bezpečným řešením je aplikaci umístit přímo na WAP gateway, nikoliv do Internetu. Nehovoříme pak o WAP gateway, ale o WTA serveru (*Wireless Telephony Application*), který je umístěn přímo v GSM síti (resp. jiné telefonní síti, která je se sítí GSM propojena).

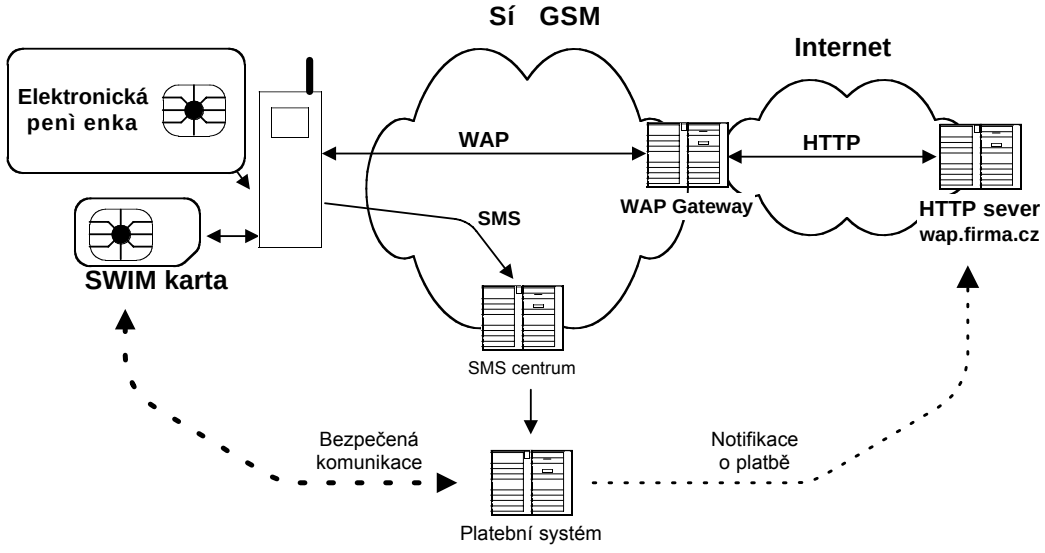
V dnes aktuální verzi 1.1 protokolu WAP není navíc WAP realizován na SIM kartě, ale přímo v hardwaru mobilního telefonu.

Připravovaná verze 1.2 protokolu WAP bude obsahovat modul WIM (*Wireless Identity Module*), který zajistí autentizaci klienta, uchování privátního klíče, ochranu PINem a digitální podpis. Předpokládá se, že bude využit internetový bezpečnostní standard PKI (*Private Key Infrastructure*), tj. práce s certifikáty vydanými pro WAP server a případně i pro WAP klienta.

Modul WIM bude implementován již na čipové kartě. Implementace modulu WIM je možná dvojím způsobem:

- Implementace WIM na samostatné čipové kartě, což je možné pouze u mobilních telefonů, které mají druhý slot.
- Implementace WIM přímo na SIM kartě. Takovéto karty se označují jako SWIM (SIM+WIM=SWIM). V současné době však SIM karty nemají dostatečnou kapacitu paměti RAM pro tuto implementaci.

Komerční aplikace může v budoucnu pracovat jak je znázorněno na obr. 3.44. Uživatel brouzdá po Internetu přes WAP gateway. V případě, že chce zaplatit nějaké zboží, tak pomocí elektronické peněženky provede platbu. Tato platba je zabezpečena mezi SWIM kartou a platebním systémem. Již dnes existují microbrowsery, které umí integrovat menu z WAP stránky s položkami generujícími platbu s příslušným zabezpečením.



Obr. 3.44 Aplikace se SWIM kartou

Vytváření WAP aplikací

WAP aplikace jsou velmi podobné WWW aplikacím. Pro vytváření stránek je definován jazyk WML. Zdrojové informace jsou uloženy v jednotlivých kartách (*card*), tyto karty jsou sdružovány do balíčků (*deck*).

WAP podporuje obrázky ve formátu WBMP. WAP podporuje i scriptování na straně mobilního telefonu. Pro scriptování se používá jazyk WMLS, jedná se o analogii java scriptů na WWW. Funkčnosti WMLS scriptů jsou omezenější oproti Java scriptům, vzhledem k omezené velikosti paměti na telefonu. Z WTLS scriptu lze využít v aplikaci vlastnosti koncového zařízení např. vytáčení telefonních čísel, práci s telefonním seznamem na SIM kartě, odesílání textových zpráv apod.

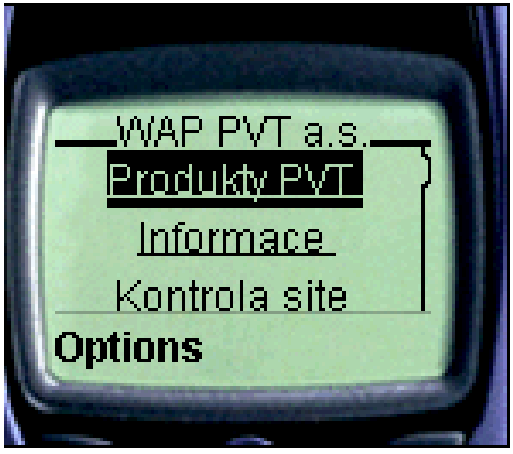
Podobně jako u WWW není potřeba žádný zvláštní nástroj na tvorbu zdrojového textu WAP stránek. Existují však i speciální editory WML stránek.

Pokud chce autor obohatit svou WAP aplikaci obrázkem, může použít některý editor obrázků pro WAP, který umožňuje vytvoření obrázku, nebo konverzi existujícího obrázku do formátu WBMP.

Pro kontrolu nebo prohlížení WAP aplikací bez použití mobilního telefonu existují WAP prohlížeče pro PC.

Příklad zobrazení WAP stránek na display mobilního telefonu:

Příklad zdrojového WML kódu, který odpovídá předchozím dvěma obrazovkám:




```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">

<wml>
<card id="index" ontimer="#home" title="WAP PVT a.s">
<timer value="20"/>
<p>

<br/>
<img src="logo.wbmp" alt="WAP server PVT"/>
</p>
</card>

<card id="home" title="WAP PVT a.s.">
<!– uvodni stana –>
<p align="center">
```

```
<anchor title="Produkty PVT">Produkty PVT
  <go href="#produkty"/>
</anchor>
<anchor title="Informace">Informace
  <go href="#prvni"/><br/>
</anchor>
<anchor title="Kontrola site">Kontrola site
  <go href="input.wml"/><br/>
</anchor>
<anchor title="WAP servery">WAP servery
  <go href="#servery"/>
</anchor>
</p>
</card>

<card id="produkty" title="Produkty PVT">
<do type="prev" label="Back">
<go href="#home"/>
</do>
<p>
Bankovni produkty:<br/>
---------<br/>
IPB GSMB<br/>
IPB HB<br/>
Ostatni produkty:<br/>
---------<br/>
RMS<br/>
SCP<br/>
CNZP<br/>
GSM<br/>
ATEC<br/>
Regioninfo<br/>
Skoleni<br/>
</p>
</card>


  <card id="prvni" newcontext="true">

<do type="prev" label="Back">
<go href="#home"/>
</do>
    <p align="center">
         <anchor title="Pocasi">Pocasi
      <go href="#pocasi"/>
    </anchor>
 <anchor title="Kurzy">
      <go href="#kurzy"/>
    </anchor>
    </p>
  </card>
```

```
<card id="pocasi">
 <do type="prev" label="Back">
      <go href="#prvni"/>
    </do>
<p>
    Ve dne svetlo v noci tma<br/>
<b>-10 az +20 stupnu C</b>
<i>dalsi informace na tel: 0603 001001</i>
</p>
 </card>
<card id="kurzy">
 <do type="prev" label="Back">
      <go href="#prvni"/>
    </do>
<p>
    <b>US$$</b> 15 Kc<br/>
    <b>DM</b>  5 Kc
</p>
 </card>
<card id="servery" title="WAP servery">
<p>
<anchor title="atlas">WAP.ALTLAS.CZ
<go href="http://wap.atlas.cz"/>
</anchor>
<anchor title="uzdroje">WAP.UZDROJE.CZ
<go href="http://wap.uzdroje.cz"/>
</anchor>
</p>
</card>
</wml>
```

4

Linková vrstva

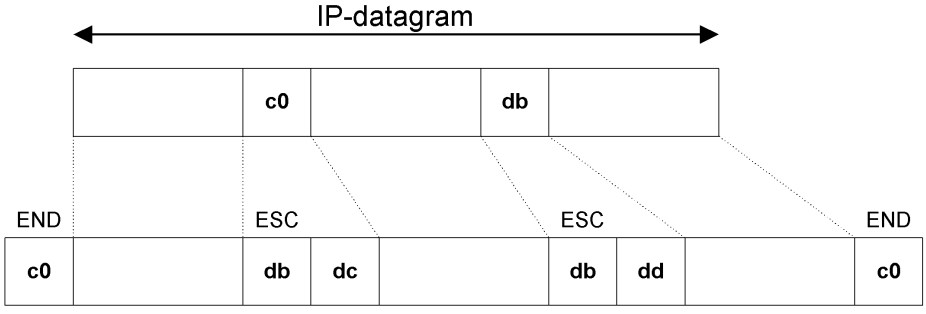
Linkových protokolů je velké množství. Zastavíme se u protokolů SLIP, CSLIP, HDLC, PPP, Frame Relay, Ethernet, FDDI a ATM. ATM, ač protokol síťové vrstvy, tak v se Internetu často používá jako linkový protokol.

4.1 SLIP

Protokol **Serial Line IP** (dále jen SLIP) vkládá IP-pakety přímo do sériové linky. Pro řízení linky jsou mezi data vkládány tzv. Esc-sekvence (analogicky jako při komunikaci počítače s terminálem či tiskárnou).

Protokol SLIP je specifikován normou RFC-1055. SLIP je velice jednoduchý protokol, který je určen pro přenos paketů síťových vrstev.

Obr. 4.1 Rámec protokolu SLIP



Každý rámec protokolu SLIP je ukončen tzv. Esc-sekvencí END ($c0_{16}$). Většina implementací protokolu SLIP však Esc-sekvenci END umísťuje navíc i na počátek rámce. Jestliže se vyskytne znak $c0_{16}$ v přenášených datech, pak je nahrazen tzv. SLIP Esc-sekvencí: dvojicí db_{16},dc_{16} (ASCII Esc-sekvence je $1b_{16}$). A konečně znak db_{16} je nahrazován dvojicí db_{16},dd_{16}.

Protokol SLIP je velice jednoduchý, ale nezabezpečuje:

- **Detekci chyb při přenosu**. Je proto výhodné použít detekci chyb alespoň na úrovni modemů např. podle doporučení V.42, protože jinak by detekce chyb nemusela být zabezpečena vůbec (IP-protokol má kontrolní součet pouze ze záhlaví a kontrolní součet v protokolu UDP je nepovinný). Je proto nebezpečné umísťovat za linky s protokolem SLIP např. DNS-servery nebo NFS-servery, které nemají zapnut kontrolní součet v UDP-datagramu.
- Rámec protokolu **SLIP nenese informaci o přenášeném protokolu** síťové vrstvy. Je proto možné přenášet vždy pouze jeden síťový protokol, tj. není možné na jedné lince mixovat např. pakety protokolu IP s pakety protokolu IPX.
- **Není možné, aby se oba konce např. informovaly o své IP-adrese či jiných konfiguračních parametrech**.
- **Nelze jej použít pro synchronní linky**.

Protokol SLIP má díky své jednoduchosti i jednu výhodu. Díky tomu, že neposkytuje téměř žádné služby, tak přenáší minimum služebních informací, takže na méně poruchových pomalých sériových linkách je poměrně oblíben.

4.2 CSLIP

Varianta protokolu SLIP s kompresí se označuje jako CSLIP (*Compressed SLIP*). Protokol CSLIP, specifikovaný RFC-1144, redukuje 40 bajtů záhlaví protokolů TCP a IP (20 bajtů z IP-záhlaví a 20 bajtů z TCP-záhlaví) na 3 až 16 bajtů. Komprimuje se IP a TCP záhlaví, nikoliv data. I když byly činěny úvahy o kompresi dat, tak v praxi se ukázalo výhodnější kompresi dat ponechat na modemech.

Stejnou kompresi IP a TCP záhlaví lze též použít u protokolu PPP. Na rozdíl od protokolu CSLIP, kde oba konce spojení musí být předem nakonfigurovány na kompresi záhlaví, tak u protokolu PPP jeden konec spojení nabízí druhému konci možnost použití komprese záhlaví – pokud se oba konce dohodnou, pak kompresi záhlaví použijí.

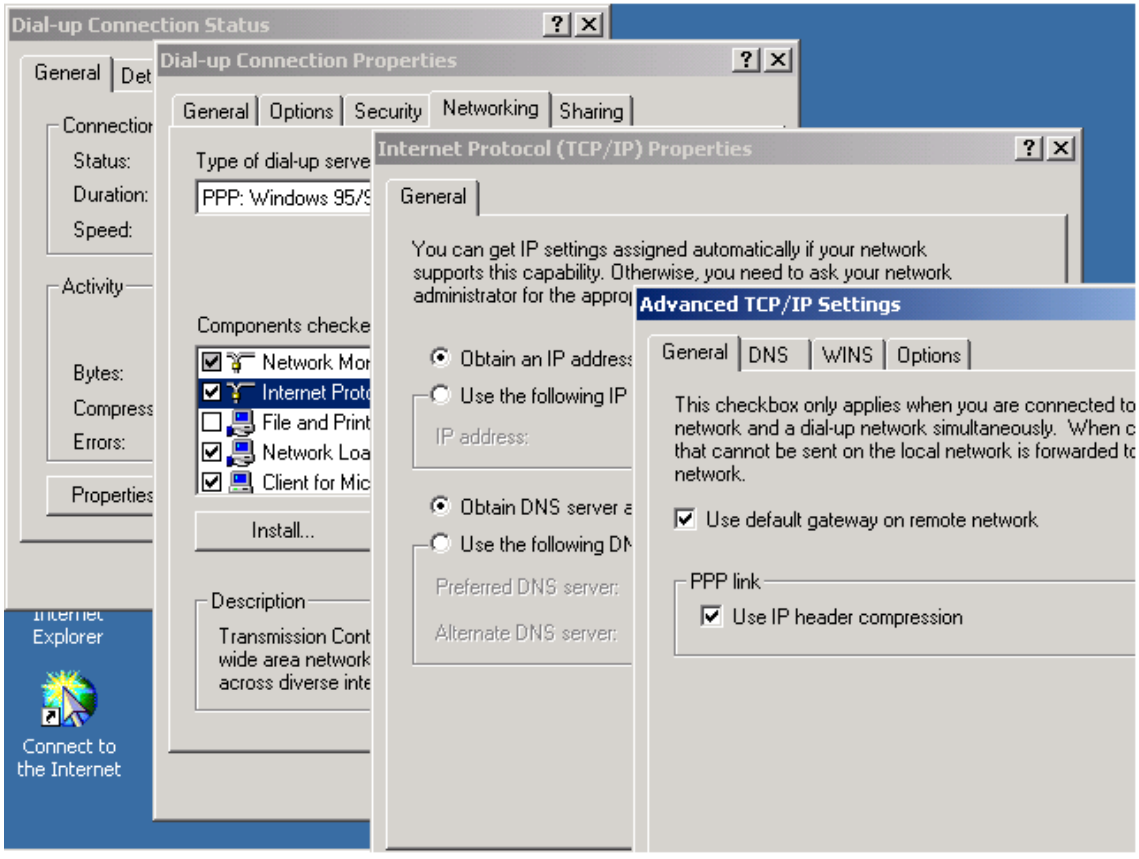
I když se hovoří o kompresi záhlaví, tak tato komprese v podstatě není kompresí jak ji známe např. u programů ZIP apod.

Myšlenka spočívá v tom, že autor se zamyslel nad IP a TCP záhlavím (viz obr. 4.3).

A zjistil, že mnohé údaje v těchto záhlavích se během TCP spojení nemění nebo se mění jen málo, takže stačí přenášet jen změněné položky IP a TCP záhlaví nebo dokonce jen přírůstky těchto položek. V podstatě se mění pouze položky: identifikace IP-datagramu, pořadové číslo odesílaného bajtu, pořadové číslo přijatého bajtu, některé příznaky, délka okna, kontrolní součet TCP záhlaví a ukazatel naléhavých dat. Změny v ostatních položkách jsou ojedinělé. Položky: celková délka IP-datagramu a kontrolní součet IP-záhlaví jsou zase postradatelné.

Komprese záhlaví komprimuje záhlaví pouze v případě, že se jedná o TCP protokol a v záhlavích se mění pouze uvedené položky. V opačném případě (např. je odeslán ICMP paket, je odeslán UDP datagram, jedná-li se o fragment IP-datagramu, je-li nastaven některý z příznaků RST, SYN, FIN nebo naopak nenastaven příznak ACK atp.) se komprese neprovede a linkou je přenesen nekomprimovaný (nezměněný) rámec.

Pokud odesílatel chce přenést TCP/IP paket, pak je paket na straně odesílatele předán komponentě označované jako kompresor (viz obr. 4.4). Kompresor buď paket zkomprimuje, nebo jej nezměněný propustí. Na příjemcově straně je pak dekompresor, který z komprimovaného paketu postaví původní paket.



Obr. 4.2 Ve Windows 2000 je v případě konfigurace protokolu PPP též možné nastavit kompresi IP a TCP záhlaví

Kompresor komprimuje jednotlivá spojení. Pro každé spojení si udržuje slot, ve kterém má všechny informace z IP i TCP záhlaví nutné pro kompresi i pro dekompresi, tj. zpětné sestavení obou záhlaví.

Nyní si představme, že odesílatelův paket dorazil na kompresor. Kompresor nejprve zkoumá, zdali je paket komprimovatelný nebo nikoliv. V případě, že se jedná o nekomprimovatelný paket (např. je odeslán ICMP paket, je odeslán UDP datagram, jedná-li se o fragment IP-datagramu, je-li nastaven některý z příznaků RST, SYN, FIN nebo naopak nenastaven příznak ACK atp.) je paket propuštěn bez komprese. V opačném případě, tj. v případě, že paket je komprimovatelný, tak se provede komprese. Tj. kompresor začne prohledávat své sloty, nemá-li v některém slotu IP+TCP záhlaví spojení, kterému by tento paket mohl příslušet.

Mohou nastat dvě situace:

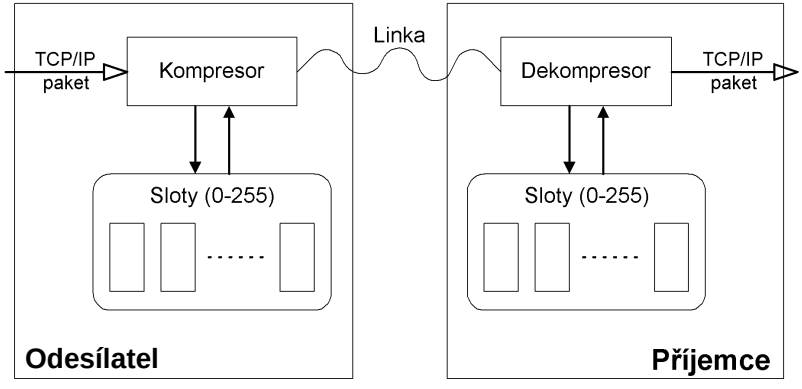
1. V žádném slotu se nenajde odpovídající IP+TCP záhlaví. Jedná se tedy o první komprimovatelný paket nového spojení (úplně první paket nového TCP spojení má nastaven příznak SYN, takže jej nelze komprimovat). V tomto případě vloží IP a TCP záhlaví tohoto paketu do prvního

volného slotu. Pokud žádný slot již není volný, pak použije nejdéle nepoužívaný slot. Kompresor tento paket nekomprimuje, pouze v poli "Protokol vyšší vrstvy *(Protocol)*" změní hodnotu 6 (pro protokol TCP) na číslo použitého slotu.

0	8	16	24	31
Verze IP 4 bity	Délka záhlaví	Typ služby 8 bitů	Celková délka IP-datagramu 16 bitů	
Identifikace IP-datagramu 16 bitů		Příznaky (*flags*)	Posunutí fragmentu od počátku (*fragment offset*) - 13 bitů	
Doba života datagramu (*TTL*) - 8 bitů	Protokol vyšší vrstvy (*protocol*) - 8 bitů	Kontrolní součet z IP-záhlaví (*checksum*) 16 bitů		
IP-adresa odesilatele (*source IP-adress*) 32 bitů				
IP-adresa příjemce (*destination IP-adress*) 32 bitů				
Zdrojový port (*source port*) 16 bitů		Cílový port (*destination port*) 16 bitů		
Pořadové číslo odesílaného bajtu (*sequence number*) 32 bitů				
Pořadové číslo přijatého bajtu (*acknowledgment number*) 32 bitů				
Délka záhlaví 4 bity	Rezereva 6 bitů	**URG** ACK **PSH** RST SYN FIN	**Délka okna (*window size*)**	
Kontrolní součet (*TCP checksum*) 16 bitů			**Ukazatel naléhavých dat (*urgent pointer*) 16 bitů**	

Obr. 4.3 IP a TCP záhlaví

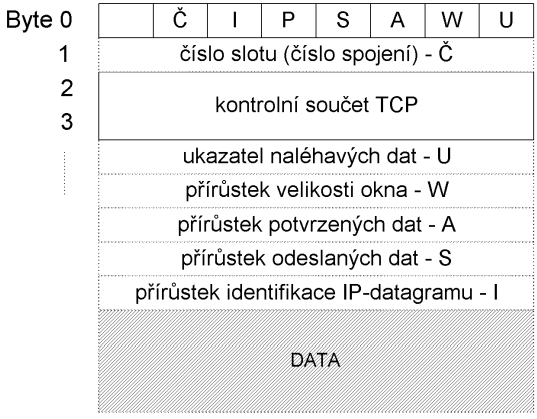
Obr. 4.4
Kompresor a dekompresor



2. Ve slotu číslo *č* našel IP + TCP záhlaví, které odpovídá předchozímu paketu spojení. Budeme říkat, že kompresor zjistil, že paket odpovídá spojení *č*. V takovém případě kompresor provede kompresi paketu.

Struktura komprimovaného paketu je na obrázku 4.5 (nepovinné položky jsou znázorněny tečkovaně). Komprimovaný paket má komprimované IP a TCP záhlaví. Komprese se netýká datové části paketu.

Obr. 4.5
Komprimované záhlaví



Komprimované záhlaví obsahuje v prvním bajtu tzv. masku. Jednotlivé bity masky specifikují, které položky v záhlaví originálního paketu se změnily, a proto celé položky nebo jejich přírůstky musí být přenášeny i v komprimovaném záhlaví. Je-li příznak nastaven, pak v komprimovaném záhlaví je uvedena konkrétní položka komprimovaného záhlaví, pokud není nastaven, pak příslušná položka není v komprimovaném záhlaví přítomna.

Vždy se přenáší kontrolní součet z TCP-záhlaví.

Jednotlivé bity masky:

- Č – označuje číslo slotu. Číslo slotu není povinné, pokud není uvedeno, pak se předpokládá, že je shodné s číslem slotu předchozího komprimovaného paketu přenášeného linkou. Číslo slotu je dlouhé jeden bajt (tj. nabývá hodnoty v intervalu 0-255), protože číslo slotu se mezi kompresorem a dekompresorem přenáší v poli "Protokol vyšší vrstvy *(Protocol)*", které je dlouhé právě jeden bajt.

  Na lince je tedy možné v jednom okamžiku komprimovat max. 255 spojení. Proto je komprese určena spíše pro linky připojující PC k Internetu a není příliš vhodná pro propojení páteřních směrovačů.
- U – ukazatel naléhavých dat. Signalizuje, že je v paketu vyplněno pole obsahující ukazatel naléhavých dat.
- W – přírůstek velikosti okna. V komprimovaném záhlaví se nepřenáší hodnota celého okna, ale pouze její přírůstek. Pokud by přírůstek byl záporný nebo větší než 64K (tj. nevešel by se do dvou bajtů), pak se paket nekomprimuje. Obdobně je tomu i u bitů A, S a I.
- A – přírůstek potvrzených dat.
- S – přírůstek odeslaných dat.

- I – přírůstek identifikace IP-datagramu.
- P – příznak PUSH. Tento příznak se odlišuje od ostatních příznaků tím, že mu neodpovídá konkrétní položka komprimovaného záhlaví. Je-li tento příznak nastaven, pak originální paket má nastaven příznak PUSH v záhlaví TCP segmentu.

Předpokládali jsme, že položka komprimovaného záhlaví je dlouhá jeden bajt, tj. může nabývat hodnoty 0 až 255. U některých položek je třeba přenášet i větší hodnoty než 255. V komprimovaném záhlaví se u položek nepřenáší hodnota nula. Např. pokud se nezmění přírůstek odeslaných dat, tj. hodnota přírůstku by byla nula, pak se položka přírůstek odeslaných dat vůbec v komprimovaném záhlaví neobjeví. Skutečnost, že se nula nepřenáší, je využita k signalizaci, že pro položku jsou použity tři bajty. První bajt je 0 a v dalších dvou bajtech je hodnota položky nebo přírůstku položky. Např. nejvyšší možná hodnota 65535 je hexadecimálně uložena ve třech bajtech 00 ff ff.

I když TCP spojení je plně duplexní, tak komprimace IP + TCP záhlaví se provádí pro každý směr zcela samostatně, tj. jako by šlo o dva samostatné simplexní spoje.

V případě, že by v komprimovaném záhlaví byly současně nastaveny příznaky A, W a U, pak se přenášený paket nekomprimuje – jedná o výjimku připravenou pro protokoly telnet, rlogin apod. Pro tyto protokoly se komprimované záhlaví skládá pouze z masky s nastavenými příznaky A, W a U a kontrolního součtu, tj. komprimované záhlaví se zkracuje na 3 bajty. V tomto případě se opravdu při stisknutí jedné klávesy na terminálu místo 41 bajtů přenáší pouze čtyři bajty (3 bajty komprimovaného záhlaví a 1 bajt dat). Blíže viz RFC-1144.

4.3 HDLC

Protokol HDLC provádí detekci chyb i řízení toku dat. HDLC je normalizován mj. normami: ISO-3309, ISO-4335, ISO-7776, ISO-7809, ISO-8471 a ISO-8885.

Protokol HDLC vznikl z protokolu SDLC firmy IBM. Protokol SDLC byl určen pro synchronní přenos. Dnes se protokol SDLC vesměs chápe jako podmnožina protokolu HDLC, i když ne všechny možnosti protokolu SDLC byly do HDLC zahrnuty.

Později byla norma HDLC rozšířena i pro asynchronní přenos. Odlišnosti asynchronní varianty si ukážeme na příkladu protokolu PPP, který je podmnožinou protokolu HDLC a zpravidla se s ním setkáváme na asynchronních linkách. Nadále budeme v této kapitole předpokládat synchronní bitově orientovaný přenos na fyzické úrovni.

Protokol HDLC je velice rozsáhlá norma mající velké možnosti (mnohé jsou volitelné či se dokonce vzájemně vylučují). Jednotliví výrobci zpravidla realizují část této normy a mnohé detaily si upraví podle sebe. Výsledkem je skutečnost, že realizace protokolu HDLC např. firmou Digital a firmou CISCO (či jinými firmami) jsou vzájemně nekompatibilní. Proto dnes většina firem dodává programy nejen pro vlastní realizaci protokolu HDLC, ale i pro implementace svých nejdůležitějších konkurentů. Takže se můžete setkat s tzv. CISCO HDLC, DEC HDLC atp.

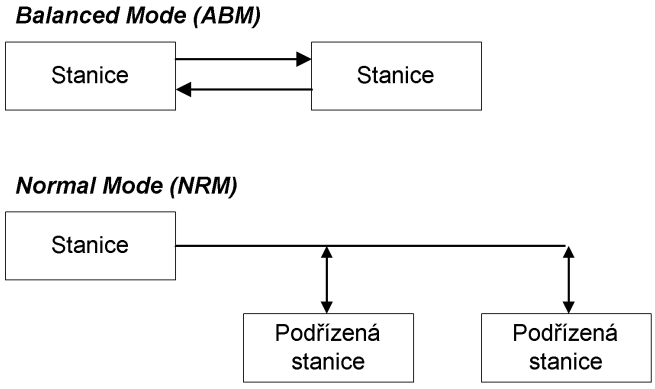
Protokol HDLC rozeznává tzv. módy:

- **ABM (*Asynchronous balanced mode*)**, který je určen pro propojení dvou stanic plně duplexním spojem, tj. obě stanice mohou vysílat současně, aniž by si vzájemně na lince překážely. Většinou se dnes setkáváme právě s tímto módem.

- **NRM (*Normal response mode*).** Tento mód odpovídá víceméně protokolu SDLC. Jedná se o situaci, kdy je propojeno více stanic na polo-duplexním spoji (tzv. přepínaný duplex mezi vysíláním a příjmem). Pro vysílání i příjem slouží společné přenosové médium, tj. v jednom okamžiku lze buď přijímat, nebo vysílat. Jedna stanice je označena jako řídící stanice a ostatní jako podřízené stanice. Je definován tzv. *pooling*, tj. řízení, kdy která stanice smí vysílat. Bez povolení smí vysílat pouze řídící stanice. Ostatní stanice mohou vysílat jen tehdy, když jim to řídící stanice povolí. Povolení se nastavuje P/F bitem v řídícím poli HDLC-rámce. Pouze řídící stanice může vydávat příkazy – podřízené stanice mohou jen odpovídat.

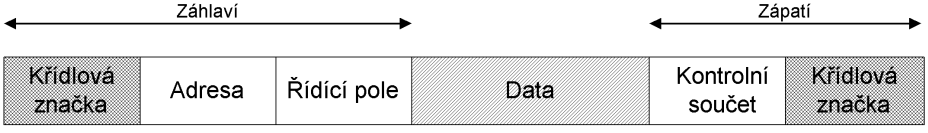
  Tento mód je v Internetu používán jen výjimečně. Nebudeme se jím zabývat – uvádíme jej zejména proto, abychom vysvětlili význam P/F bitu.
- **ARM (*Asynchronous response mode*)** je dnes málo běžný režim.

Obr. 4.6
Módy ABM a NRM



Formát HDLC rámce je znázorněn na obr. 4.7.

Obr. 4.7
Rámec protokolu HDLC



4.3.1 Křídlová značka (Flag)

Křídlová značka uvozuje datový rámec, tj. každý HDLC-rámec začíná a končí právě křídlovou značkou. Na přenosové lince se mohou vyskytovat posloupnosti křídlových značek (např. v klidovém stavu). Jdou-li dvě křídlové značky po sobě, pak uvozují prázdný rámec, který se nezpracovává.

Křídlová značka se skládá z osmi bitů: 0111 1110. Šest po sobě následujících jedniček určuje právě křídlovou značku. Okamžitě můžete namítnout: Vždyť přenášený znak se může skládat i z více jak šesti jedniček. Jenže právě bitově orientovaná synchronní verze HDLC používá trik. Na vstupu, kdykoliv, když data obsahují více jak pět jedniček za sebou, tak se za těchto pět jedniček automaticky vloží jedna nula. Analogicky na výstupu, je-li v přenášených datech za pěti jedničkami 0, pak se tato nula vypouští. Není-li za pěti jedničkami nula, ale jednička, pak se jedná o křídlovou značku. Tato technika se též označuje jako *bit stuffing*.

Tato technika je možná jen u bitově orientovaného přenosu, kde se přenáší řada bitů, u znakově orientovaného přenosu tato technika není možná, protože počet přenášených bitů musí být dělitelný délkou znaku (zpravidla 7 nebo 8 bitů). Vložení bitu by pak toto pravidlo porušilo.

4.3.2 Adresní pole

Adresní pole je dlouhé 8 bitů. Vyjadřuje adresu stanice, které je paket určen. Je vcelku evidentní, že toto pole má své opodstatnění u módu NRM (resp. protokolu SDLC), kdy spolu komunikují často více jak dvě stanice. Je však striktně vyžadováno, proto je přítomno ve všech mutacích protokolu HDLC. Pro úplnost uveďme, že protokol PPP používá např. hodnotu 1111 1111, tj. oběžník.

Adresní pole v rámci HDLC nesouvisí s IP-adresou. Jedná se o linkovou adresu!

4.3.3 Kontrolní součet

Z přenášených dat, adresního a řídícího pole se počítá kontrolní součet, který je zpravidla buď 32bitový nebo 16bitový. Adresát z přijatého rámce rovněž spočte kontrolní součet, který porovná s kontrolním součtem v přijatém rámci. Jsou-li shodné, pak považuje přijatý rámec za správně přenesený. V opačném případě si u číslovaných rámců může vyžádat zopakování přenosu.

Určení jaký kontrolní součet se bude používat je součástí úvodního dialogu stanic při inicializaci spojení (pomocí příkazu XID).

4.3.4 Datové pole a typ přenášeného protokolu

Datové pole obsahuje přenášená data. Všimněte si, že záhlaví HDLC-rámce neposkytuje možnost specifikace protokolu vyšší vrstvy. Tj. neumožňuje např. mixovat rámce protokolu IP s protokolem IPX. Volba protokolu se přitom určuje v počátečním inicializačním dialogu.

Toto omezení platí pro tzv. číslované rámce. U nečíslovaných rámců je možné na počátek datového pole zadat specifikaci protokolu.

4.3.5 Řídící pole

Řídící pole je nejsložitější pole. Podle nejnižších dvou bitů řídícího pole rozlišujeme 3 typy HDLC-rámců:

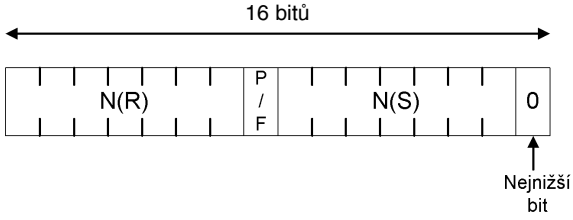
- **Informační rámce neboli I-rámce (v nejnižším bitu je 0)**, které jsou primárně určeny pro přenos dat. Mohou však ve svém řídícím poli přenášet i některé řídící informace (např. pozitivní potvrzení přijatých rámců).
- **Nečíslované rámce neboli U-rámce (v nejnižších dvou bitech je 11)**, které se používají nejen pro přenos dat, ale i pro mnohé řídící funkce (úvodní inicializační dialog, řízení linky a diagnostiku).
- **Rámce supervizoru neboli S-rámce (v nejnižších dvou bitech je 10)**. Používají se pro řízení toku dat (požadavek na vysílání, potvrzování I-rámců atd.). S-rámce mohou být používány až když je linka inicializována, tj. když mohou být používány I-rámce. S-rámce zpravidla neobsahují datové pole.

Řídící pole je u U-rámců osmibitové. U I-rámců a S-rámců může být buď osmibitové nebo šestnáctibitové. Na následujících obrázcích je použito šestnáctibitové řídící pole, tj. jedná se o tzv. rozšířený mód. Označení módů ABM a NRM je většinou míněno jako příslušný mód s osmibitovým řídícím polem. Rozšířené módy s šestnáctibitovými řídícími poli se někdy označují jako ABME a NRME.

Co se v osmibitovém řídícím poli ušetří? Použijí se pouze tři bity pro číslování N(S) i N(R), tj. rámce se nečíslují modulo 128, ale jen modulo 8.

4.3.5.1 I-rámec

Obr. 4.8
I-rámec



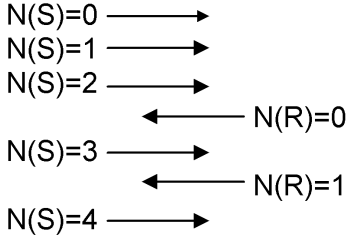
V I-rámci slouží pole N(S) a N(R) pro číslování rámců. Čísluje se od nuly do 127 (nejvyšší možné číslo v sedmi bitech), po dosažení čísla 127 se opět pokračuje od nuly. N(S) určuje číslo odesílaného rámce. Naopak pole N(R) slouží pro potvrzení přijatého rámce. Jelikož je komunikace obousměrná, potvrzují se v protisměru správně přijaté rámce.

V případě, že není třeba v protisměru posílat data, pak se k potvrzení přijatých dat použije S-rámec (s příkazem RR). V případě, že přijatý rámec byl po přepočítání kontrolního součtu shledán jako chybný, pak je pomocí S-rámce (příkazem REJ) vyžádáno opakování přenosu – tzv. negativní potvrzení. Tento S-rámec ve svém poli N(R) zopakuje číslo posledního správně přijatého rámce.

Je možné potvrzovat rámce postupně po jednom. To ovšem prodlužuje odezvu, protože se musí u každého rámce čekat na jeho potvrzení. Proto se zpravidla potvrzují rámce pomocí tzv. okna.

Je-li např. okno rovno třem (viz obr. 4.9), pak až po odeslání tří paketů se čeká na potvrzení prvního z nich. Po potvrzení prvního se odešle čtvrtý, po potvrzení druhého se odešle pátý, ... Ve vyrovnávací paměti odesílatele je nutné udržovat celé okno nepotvrzených rámců pro případ vyžádání chybného nebo ztraceného rámce.

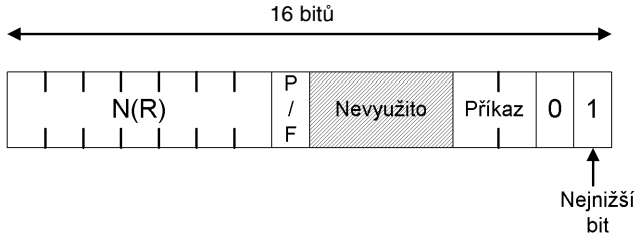
Obr. 4.9
Okno o velikosti 3



P/F bit je důležitý pro NRM mód protokolu HDLC. V NRM módu řídící stanice nastavením tohoto bitu na P (=Pool) povoluje podřízené stanici vysílat data. Podřízená stanice nechává při vysílání tento bit nastaven. Tím signalizuje, že chce ve vysílání pokračovat. U posledního vysílaného rámce tento bit shodí (nastaví F=Final).

4.3.5.2 S-rámec

Obr. 4.10
S-rámec

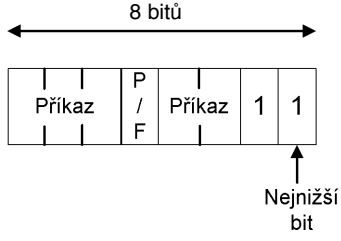


S-rámec může potvrzovat správně přijatý rámec. Dále v poli příkaz může nést následující příkazy resp. odpovědi:

- **RR** (*Receiver Ready* = přijímač připraven). Informuje protějšek, že přijímač je připraven akceptovat I-rámce. Dále se používá jako signalizace, že linka je opět volná (poté co tomu tak nebylo). Ještě musíme připomenout, že se též může použít (jak bylo uvedeno u popisu I-rámce) k potvrzení čísla správně přijatého rámce.
- **RNR** (*Receiver Not Ready* = přijímač nepřipraven). Informuje protějšek o dočasné neschopnosti přijímat I-rámce, zároveň potvrzuje dosud přijaté rámce.
- **REJ** (*Reject* = odmítnutí). Přijetí chybného rámce, tj. používá se jako příkaz nebo jako odpověď pro zopakování vysílání.

4.3.5.3 U-rámec

Obr. 4.11
U-rámec

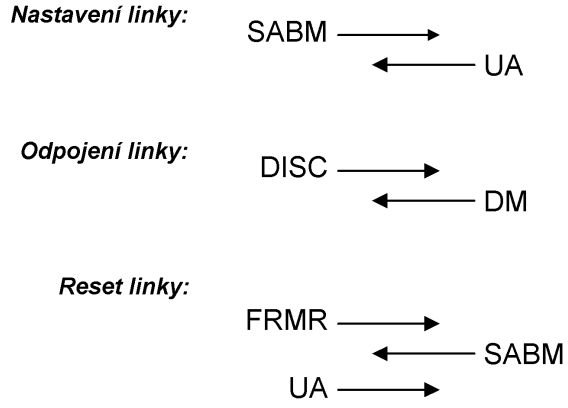


U-rámce mohou jak přenášet data, tak i příkazy a odpovědi:

- **SABM** (*Set Asynchronous Mode* = nastavení módu ABM). Příkaz nastavuje linku do módu ABM s osmibitovým řídícím polem.
- **SABME** (*Set Asynchronous Mode* = nastavení módu ABM). Příkaz nastavuje linku do módu ABM s šestnáctibitovým řídícím polem – jedná se tedy o tvar protokolu HDLC zobrazovaný v této kapitole.
- **SNRM** (*Set Normal Response Mode* = nastavení módu NRM). Příkaz nastavuje linku do módu NRM s osmibitovým řídícím polem.
- **SNRME** (*Set Normal Response Mode* = nastavení módu NRM). Příkaz nastavuje linku do módu NRM s šestnáctibitovým řídícím polem.
- **UA** (*Unnumbered Acknowledgment* = nečíslované potvrzení). Používá se pro potvrzení SABM, SABME, SNRM, SNRME a DISC.

- **DISC** (*Diconnect* = odpojení). U komutovaných linek znamená požadavek na zavěšení. U pevných linek umožňuje zrušit aktuální operační mód (aktuální nastavení).
- **DM** (*Disconnect Mod*). Pozitivní potvrzení příkazu DISC.
- **FRMR** (*Frame Reject* = odmítnutí rámce). Používá se k indikaci přijetí vadného rámce. Přitom není možné dosáhnout opravy opakováním vysílání (retransmisí). Po obdržení FRMR se začíná znovu od nastavení módu linky, tj. jedním z příkazů SABM, SABME, SNRM či SNRME. Počátek datové části paketu obsahuje 2-3 bajtové pole s kódem chyby (např. chyba v řídícím poli rámce, chyba v informačním poli, překročení kapacity rámce, chyba v sekvenci přijatých rámců atp.)
- **XID** (*Exchange Station Identification* = výměna konfiguračních informací). Příkazy a odpovědi XID se používají k úvodní inicializační sekvenci, kdy se stanice domlouvají na délce kontrolního součtu, přenášeném protokolu vyšší vrstvy atd.
- **UI** (*Unnumbered Information* = nečíslované datové rámce). Používá se pro přenos nečíslovaných datových rámců. Tyto rámce mohou na počátku datového pole obsahovat specifikaci přenášeného protokolu, takže je pak možné na přenosové lince mixovat různé protokoly (např. IP a IPX).

Obr. 4.12
Jednoduché dialogy protokolu HDLC



4.3.6 Závěr k protokolu HDLC

Protokol HDLC je vyspělý linkový protokol umožňující:

- Pomocí kontrolního součtu zjišťovat chybné rámce.
- Při přijetí chybného číslovaného rámce je možné vyžádat zopakování vysílání rámce (nečíslované chybné rámce se zahazují).
- Pomocí nečíslovaných rámců je možné na lince mixovat více síťových protokolů.

Linka se v protokolu HDLC nachází v:

- Odpojeném stavu.
- Ve stavu nastavování linky, kdy se mohou používat pouze U-rámce.
- Ve stavu přenosu informací, tj. za běžných okolností se přenášejí pouze I a S-rámce (nepoužívají-li se k přenosu dat U-rámce).
- Odpojování linky, opět se přenáší jen U-rámce.

4.4 PPP

Protokol PPP využívá rámce tvaru protokolu HDLC. Nevyužívá však zdaleka všechny možnosti protokolu HDLC.

- Na fyzické úrovni je schopen používat rozhraní podle doporučení V.24, V.35 atp. Nevyžaduje žádné řídící signály (RTS, CTS, DCD, DTR atp.). Řídící signály však mohou být využity pro zvýšení efektivity.
- Může používat jak asynchronní, tak bitově či znakově synchronní přenos dat.
- Pro asynchronní přenos použije 1 start bit, 8 datových bitů a 1 stop bit (bez parity).
- Vyžaduje plně duplexní dvojbodové spoje (*point-to-point*), které mohou být pevné i komutované.
- Využívá zpravidla 16 nebo 32 bitů pro kontrolní součet, aby mohl zjistit, zda nebyl rámec během přenosu poškozen.
- Cílem protokolu PPP je umožnit po jedné lince přenos více síťových protokolů současně (mixovat protokoly). Nepoužívá I-rámce, ale přenos provádí pouze pomocí U-rámců. Nemůže tedy použít číslování rámců, a tedy ani možnost opakování rámce v případě zjištění chybného rámce.
- Na počátku datového pole umísťuje osmi nebo šestnáctibitovou identifikaci přenášeného síťového protokolu.

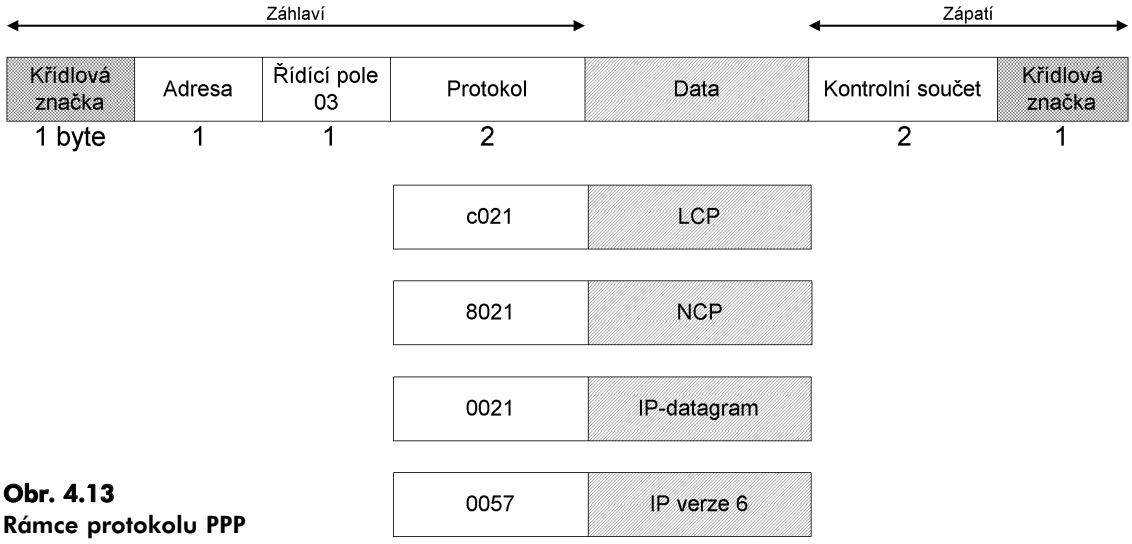
Protokol PPP specifikuje RFC-1661. Tvar rámce PPP-protokolu specifikuje RFC-1662. Na obr. 4.13 jsou znázorněny tvary rámců protokolu PPP.

Rámec protokolu PPP obsahuje v poli adresa ff_{16} (oběžník) a v řídícím poli vždy 03_{16} (U-rámce s nastaveným P/F bitem na 0). Pokud se na lince vyskytují rámce pouze s těmito adresami a řídícími poli, pak oba konce linky mohou použít kompresi (*Address-and-Control-Field-Compression*). Při této kompresi se prostě při vysílání obě pole vypustí.

Nyní ke křídlové značce (flag). Křídlovou značkou je uvozen i ukončen každý rámec protokolu PPP. Křídlová značka obsahuje binárně 0111 1110, tj. $7e_{16}$. Co ale když je třeba znak 7e přenášet v datech? U binárně synchronních linek jsme si popsali techniku *bit stuffing*.

Pro asynchronní spoje (též pro znakově synchronní linky) se použijí Esc-sekvence (podobně jako u protokolu SLIP). Znak 7e se nahradí dvojicí 7d 5e. A znak 7d se nahradí dvojicí 7d 5d.

Implicitně se uvozují escape sekvencí 7d i všechny řídící znaky ASCII (tj. znaky s kódem desítkově menším než 32). Navíc se k hodnotě těchto znaků připočte desítkově 32 (tj. 20 šestnáctkově). Např. místo znaku 03 se přenáší 23_{16}. Takže ani terminálový ovladač nemůže přenášeným znakům uškodit tím, že by je chybně interpretoval např. jako zvonek, BACKSPACE atp. Možná vás zarazilo slovo implicitně



Obr. 4.13
Rámce protokolu PPP

v úvodu tohoto odstavce. Ale obě stanice se mohou pomocí příkazu *Async-Control-Character-Map* (ACCM) dohodnout na tabulce znaků, které se budou uvozovat escape sekvencí.

Na binárně synchronní lince se escape sekvence nepoužívají. Ale není tomu tak vždy. S escape sekvencemi je možné se setkat i na binárně synchronních linkách. Proč? V případě, že je třeba konvertovat přenos z asynchronních na bitově synchronní (a naopak), pak escape sekvence přejdou z asynchronního přenosu do synchronního jako znaky. Při odpovědi musí naopak synchronní strana obohatit synchronní data o escape sekvence, aby bylo po konverzi možné komunikovat s protějškem. Takže i synchronní stanice mohou používat příkaz *Async – Control – Character – Map*. Takováto konverze se používá např. když máme synchronní linku připojenu do PC a používáme autosynchronní modem. Tj. komunikace vychází z PC asynchronně a v autosynchronním modemu se mění na synchronní.

Součástí protokolu PPP jsou dva služební protokoly:

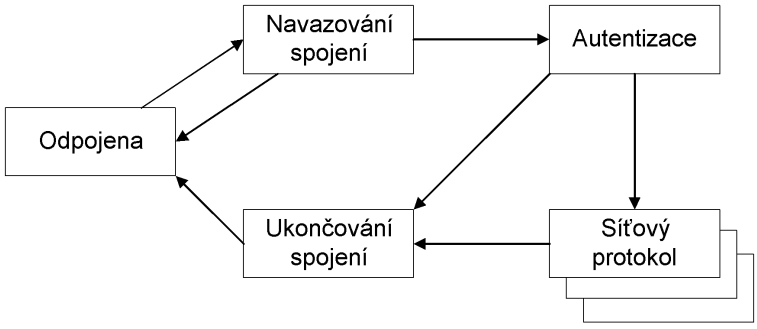
- Protokol LCP sloužící k navázání spojení, autentizaci stanic apod.
- Skupina protokolů NCP. Důležité je množné číslo. Každý síťový protokol, který bude využívat linkový protokol PPP má definovánu vlastní normu pro protokol NCP. Součástí této normy je vždy i číslo protokolu, které se použije v poli protokol rámce, a to jak pro příslušný protokol NCP (číslo začíná číslicí 8), tak i pro datové rámce (číslo začíná číslicí 0). Máme např.
  - **Protokol IPCP** (číslo protokolu 8021_{16}), je variantou protokolu NCP pro IP-protokol verze 4, IPCP je specifikován RFC-1332. Datové rámce používají v poli protokol hodnotu 0021_{16}.
  - **Protokol IPV6CP** (číslo protokolu 8057_{16}), je variantou protokolu NCP pro IP-protokol verze 6 (RFC-2023). Datové rámce přenášené protokolem IP verze 6 používají číslo protokolu 0057_{16}.
  - **Protokol SNACP** (číslo protokolu 804d), tj. protokol NCP pro IBM SNA (RFC-2043). Datové rámce používají číslo protokolu 004d.
  - **Protokol DNCP** (číslo protokolu 8027), tj. protokol NCP pro DECnet Pahse IV (RFC-1762). Datové rámce používají číslo protokolu 0027.

- **Protokol IPXCP** (číslo protokolu 802b), tj. protokol NCP pro IPX (RFC-1552). Datové rámce používají číslo protokolu 002b.
- **Protokol OSINLCP** (číslo protokolu 8023), tj. protokol NCP pro OSI protokoly, tj. např. protokoly ES-IS, IS-IS atd. (RFC-1377). Datové rámce používají číslo protokolu 0023.

4.4.1 Protokol LCP

Protokol LCP se používá ještě před tím, než se vůbec uvažuje o tom, jaký síťový protokol na lince poběží. LCP je společný protokol (na rozdíl od protokolů NCP) pro všechny síťové protokoly. Protokol LCP je určen pro navázání spojení, ukončení spojení, výměnu autentizačních informací apod. Linka se nachází postupně ve fázi navazování spojení, autentizace, síťový protokol a ukončování spojení, jak je znázorněno na obr. 4.14.

Obr. 4.14
Jednotlivé fáze linky v protokolu PPP



Linka odpojena je fáze, ze které se vždy začíná a končí. Když dojde k nějaké externí události (např. modemy ztratí mezi sebou spojení nebo síťový administrátor vydá příkaz k ukončení spojení), přechází linka do této fáze.

Z fáze linka odpojena se přechází do fáze **navazování spojení**. Navazování spojení se provádí výměnou konfiguračních paketů. Během navazování spojení se žádné datové pakety nepřenáší (tj. pakety síťového protokolu – např. IP). V případě výskytu datového paketu během navazování spojení se takový paket zahazuje.

Autentizace je fáze, kdy klient prokazuje svou totožnost. Slovo klient jsem použil záměrně. Asi jste si položili otázku, kdo je to klient? Klientem je ta strana (stanice), která je vyzvána k prokázání své totožnosti. Po prokázání totožnosti jedné stanice si mohou stanice svou roli vyměnit a k prokázání své totožnosti může být vyzvána druhá strana. V praxi většinou prokazuje svou totožnost jen jedna strana (např. uživatel PC proti poskytovateli Internetu).

Autentizace je nepovinná, tj. může být přeskočena. Během autentizace opět nemohou být ještě přenášeny datové pakety (síťový protokol).

Autentizace pouze přenáší data používaná k vlastnímu prokazování totožnosti. Tj. protokol LCP nepopisuje žádný autentizační algoritmus, pouze přenáší data, která pak následně využijí autentizační protokoly. Jako autentizační protokol se používá buď protokol PAP, nebo protokol CHAP. Navíc ještě je zpravidla možná terminálová autentizace.

Fáze, která je na obr. 4.14 označena jako **síťový protokol** v sobě může obsahovat celou řadu kroků. V tomto okamžiku přicházejí ke slovu jednotlivé protokoly NCP. Každý síťový protokol, který chce lin-

ku využívat si musí přivést pomocí svého protokolu NCP linku do otevřeného stavu pro tento protokol. Pokud se objeví datové pakety síťového protokolu, pro který není linka otevřena, pak se tyto pakety zahodí.

Např. mají-li se na lince přenášet pakety protokolu IP verze 4 a protokolu IS-IS, pak se musí linka otevřít dvakrát, jednou pomocí protokolu IPCP a podruhé pomocí OSINLCP. Po otevření linky pro konkrétní síťový protokol se teprve mohou začít přenášet data konkrétního síťového protokolu. Linka může být otevřena pro více síťových protokolů současně.

Poslední fází je fáze **ukončování spojení**. Během této fáze jsou všechny jiné pakety než protokolu LCP již zahazovány. Fyzické vrstvě je signalizováno ukončení spojení. Fyzická vrstva může reagovat např. zavěšením komutované linky.

Formát rámce LCP protokolu je vyjádřen na následujícím obrázku 4.15.

Obr. 4.15
Protokol LCP

Osmibitové pole kód specifikuje typ příkazu (resp. odpovědi) protokolu LCP:

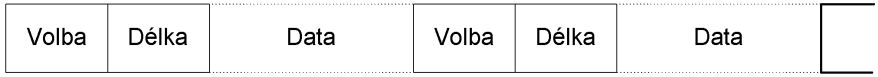
kód	Název (anglicky)	Význam
1	Configure-Request	Konfigurační paket nesoucí požadavky na změnu implicitních parametrů linky.
2	Configure-Ack	Konfigurační paket s kladným potvrzením požadavků na změnu implicitních parametrů linky. Tj. všechny požadované změny parametrů jsou akceptovány.
3	Configure-Nak	Konfigurační paket s odpovědí. Avšak protější strana neakceptuje všechny požadavky na změnu parametrů linky. Ty, které neakceptuje, jsou v tomto paketu specifikovány. Ostatní požadavky jsou akceptovány (tj. nespecifikované požadavky v paketu Configure-Nak jsou akceptovány).
4	Configure-reject	Konfigurační paket odmítající všechny požadavky. Může být důsledkem i chybného kódu požadavku.
5	Terminate-Request	Požadavek na ukončení spojení.
6	Terminate-Ack	Potvrzení požadavku na ukončení spojení.
7	Code-Reject	Odmítnutí požadavku z důvodu neznámého kódu. Může být způsobeno i tím, že protější stanice používá jinou verzi protokolu.
8	Protokol-Reject	Protější strana nepodporuje uvedený protokol.
9	Echo-Request	Podpora testovací smyčky na linkové úrovni.
10	Echo-Reply	Povinná odpověď na Echo-Request.
11	Disckard-Request	Zahoď tento paket. Používá se pro testování linky při zátěži, tj. odesílatel generuje pomocí těchto paketů umělou zátěž linky.

Osmibitové pole **ID** je identifikace požadavku. Odesílatel zvolí identifikaci do tohoto pole a adresát ji zkopíruje do své odpovědi. Pomocí tohoto pole se určuje příslušnost odpovědi k danému požadavku.

Šestnáctibitové pole **délka** obsahuje číslo udávající součet délek polí: kód, ID, délka a volby.

Pole **volby** obsahuje jednotlivé požadavky (resp. odpovědi) na změnu implicitních parametrů linky. Toto pole se skládá z jedné nebo více voleb. Jednotlivé volby jsou ukládány sekvenčně za sebou jak je znázorněno na obr. 4.16.

Obr. 4.16
Struktura voleb



Následující tabulka uvádí některé volby. Pole volba a délka jsou osmibitová.

kód	Název volby (anglicky)	Význam
1	Maximum-Receive-Unit	Pomocí této volby se stanice mohou dohodnout na délce rámce (MTU) delší než 1500 bajtů (všechny stanice jsou povinny přenášet rámce délky alespoň 1500 bajtů)
3	Authentication-Protocol	Pole data je uvozeno dvojbajtovým polem specifikujícím typ autentizačního protokolu (typ autentizačního protokolu je např. pro protokol PAP šestnáctkově c023 a pro protokol CHAP c223). Za tímto polem následují vlastní autentizační data.
7	Protocol-Field-Compression	Rámec protokolu PPP obsahuje dvoubajtové pole specifikující typ protokolu (např. 0021 pro IP-datagramy). Při zahájení komunikace se musí toto pole používat vždy dvoubajtové. Po potvrzení Protocol-Field-Compression se používá toto pole jenom jednobajtové.
8	Adress-and-Control -Field-Compression	V rámci protokolu PPP je vždy v poli adresa hodnota ff a v řídícím poli hodnota 03. Po potvrzení Adress-and-Control-Field-Compression odesílatel tato pole vypouští a příjemce je zpět automaticky doplňuje.

Příklad LCP odchyceného MS Network Monitorem ve Windows 2000 (Terminate-Ack):

```
Frame: Base frame properties
      Frame: Time of capture = 3/15/2000 9:37:11.400
      Frame: Time delta from previous physical frame: 130187 microseconds
      Frame: Frame number: 17
      Frame: Total frame length: 18 bytes
      Frame: Capture frame length: 18 bytes
      Frame: Frame data: Number of data bytes remaining = 18 (0x0012)
  PPP: Unknown Frame (0x0)
      PPP: Destination Address =  RECV_
      PPP: Source Address =  RECV_
      PPP: Protocol = Link Control Protocol
```

```
  LCP: Terminate Ack Packet, Ident = 0x08, Length = 4
      LCP: Code = Terminate Acknowledgement
      LCP: Identifier = 8 (0x8)
      LCP: Length = 4 (0x4)
      LCP: Data: Number of data bytes remaining = 0 (0x0000)

00000:  20 52 45 43 56 05 20 52 45 43 56 05 C0 21 06 08    RECV. RECV.Ŕ!..
00010:  00 04                                             ..
```

4.4.2 Autentizace

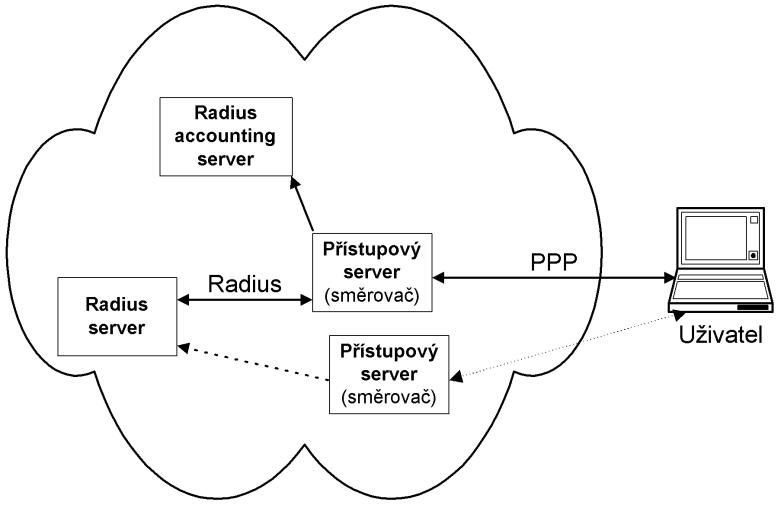
Prokazovat totožnost lze tč. v případě protokolu PPP trojím způsobem (neuvažujeme-li jako čtvrtou možnost eventualitu, že autentizace je zcela vynechána):

- **Terminálový dialog**. Terminálový dialog nesouvisí s protokolem PPP, nýbrž s jeho implementací. Zpravidla se totiž uživatel přihlašuje po sériové lince k serveru. Na serveru sedí na této lince terminálový proces vyžadující jméno uživatele a heslo. Teprve ze jména uživatele pozná, že se nejedná o běžného uživatele terminálu, ale uživatele, pro kterého má na lince startovat protokol PPP (např. proces pppd). Pokud je takováto autentizace na serveru možná a je dostačující, pak je možné autentizační fázi protokolu PPP přeskočit.
- **Protokol Password Authentication Protocol (PAP).** Tento protokol je obdobou autentizace pomocí terminálového dialogu. Tj. uživatel prokazuje svou totožnost také pomocí jména uživatele a hesla. Pro výměnu autentizačních informací se ale použije protokol LCP, tj. jméno uživatele a heslo se nevkládá přímo na linku, ale balí se do protokolu LCP.
- **Challenge Handshake Authentication Protocol (CHAP)**. Je považován za dokonalejší. Oba konce sdílí stejné tajemství (v podstatě je to šifrovací klíč symetrické šifry). Stanice, která autentizaci inicializuje, vygeneruje náhodný řetězec jako dotaz (challenge), který odešle druhé straně. Druhá strana tento řetězec zašifruje pomocí sdíleného tajemství a odešle zpět. Stanice, která autentizaci inicializovala, tak obdržela zašifrovaný řetězec, který dešifruje. Porovná oba řetězce. Jsou-li oba řetězce stejné, pak protějšku potvrdí úspěšný výsledek autentizace. V opačném případě odpoví, že autentizace proběhla neúspěšně a může se začít znovu s navazováním spojení.

Výhodou protokolu CHAP je skutečnost, že oba konce znají sdílené tajemství – je tak snadno možné provádět oboustranně autentizaci. Sdílení tajemství je současně nevýhodou protokolu CHAP, nelze zabránit zneužití tohoto tajemství druhou stranou (na rozdíl od autentizace heslem, kde druhá strana má přístup pouze k zašifrovanému heslu). Protokol CHAP specifikuje RFC-1994.

Další problém s autentizací spočívá v tom, že se klient bude chtít přihlašovat nikoliv stále na jeden přístupový server, ale na různé přístupové servery. Klasickým případem je připojení k poskytovateli Internetu, který má své přístupové body v různých městech. V takovém případě by autentizační informace musely být udržovány na každém přístupovém serveru. Myšlenka spočívá v centralizaci autentizačních informací. V síti je jeden (nebo i více záložních) serverů, které udržují autentizační informace o každém uživateli. Kromě autentizačních informací mohou být udržovány i konfigurační informace (např. IP-adresa uživatele, přístupové filtry atd.). Přístupový server pak vůči takovémuto serveru vystupuje jako klient, který požaduje službu: prověření autentizační odpovědi či poskytnutí IP-adresy, kterou má protokolem IPCP předat uživateli atd. Jako protokol mezi přístupovým serverem a serverem s autentizačními a konfiguračními informacemi se dnes často používá protokol RADIUS nebo protokol TACAC+. Protokol RADIUS je aplikační protokol.

Obr. 4.17
Radius a radius accounting protokol

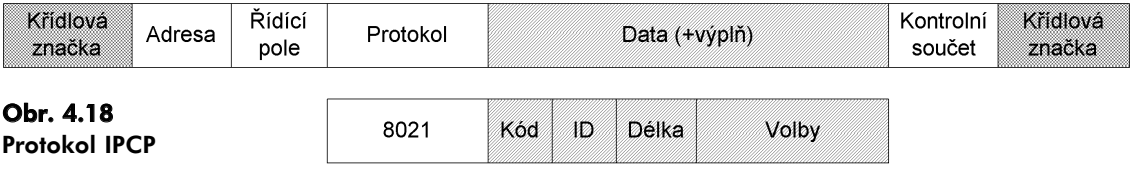


Kromě protokolu RADIUS existuje ještě RADIUS Accounting Protocol. Tímto protokolem mohou přístupové servery předávat informace o připojování a odpojování uživatelů. RADIUS Accounting Server pak soustřeďuje tyto informace, které pak mohou být využity např. pro zúčtování přístupu uživatelů do Internetu.

4.4.3 Protokol IPCP

Protokol IPCP je protokol NCP pro protokol IP verze 4.

Formát rámce IPCP protokolu je vyjádřen na následujícím obrázku:



Obr. 4.18
Protokol IPCP

Osmibitové pole kód specifikuje typ příkazu (resp. odpovědi) protokolu IPCP:

kód	Název kódu (anglicky)	Význam
1	Configure-Request	Konfigurační paket nesoucí požadavky na změnu implicitních parametrů.
2	Configure-Ack	Konfigurační paket s kladným potvrzením požadavků na změnu implicitních parametrů. Tj. všechny požadované změny parametrů jsou akceptovány.

3	Configure-Nak	Konfigurační paket s odpovědí. Avšak protější strana neakceptuje všechny požadavky na změnu parametrů linky. Ty které neakceptuje, jsou v tomto paketu specifikovány. Ostatní požadavky jsou akceptovány (tj. požadavky nespecifikované v paketu Configure-Nak jsou akceptovány).
4	Configure-reject	Konfigurační paket odmítající všechny požadavky. Může být i důsledkem chybného kódu požadavku.
5	Terminate-Request	Požadavek na ukončení spojení
6	Terminate-Ack	Potvrzení požadavku na ukončení spojení
7	Code-Reject	Odmítnutí požadavku z důvodu neznámého kódu. Může být způsobeno i tím, že protější stanice používá jinou verzi protokolu.

Osmibitové pole **ID** je identifikace požadavku. Odesílatel výplní nějaký údaj do tohoto pole a adresát jej zkopíruje do své odpovědi. Pomocí tohoto pole se určuje příslušnost odpovědi k požadavku.

Šestnáctibitové pole **délka** obsahuje číslo udávající součet délek polí: kód, ID, délka a volby.

Pole **volby** obsahuje jednotlivé požadavky (resp. odpovědi) na změnu implicitních parametrů linky. Toto pole se skládá z jedné nebo více voleb. Jednotlivé volby jsou ukládány sekvenčně za sebou.

Pole volba a délka jsou jednobajtové, jejich formát je obdobný formátu voleb protokolu LCP – viz obr. 4.16.

kód	Název kódu (anglicky)	Význam
2	IP-Compression-Protocol	Komprese TCP/IP záhlaví (viz protokol SLIP). Pole data obsahuje šestnáctkově 002d. Délka je 6, protože další 2 bajty obsahují parametry komprese.
3	IP Adress	Předání IP adresy protějšku. Takto je možno dynamicky přidělovat IP-adresy. Chce-li protějšek použít jinou IP-adresu, pak odpoví paketem Configure-Nak, kde tuto adresu specifikuje. Pole data obsahuje čtyřbajtovou IP-adresu a délka je 6.
129	Primary-DNS-Address	Specifikace primárního jmenného serveru pole data obsahuje (RFC-1877) čtyřbajtovou IP-adresu primarního jmenného serveru, délka je 6.
131	Secondary-DNS-Address	Specifikace sekundárního jmenného serveru, pole data obsahuje (RFC-1877) čtyřbajtovou IP-adresu primárního jmenného serveru, délka je 6.

V rámci protokolu PPP se pro přenos datových paketů s protokolem IP verze 4 použije identifikace protokolu 0021_{16}. V případě komprese je situace komplikovanější, protože ne všechny pakety mají komprimované záhlaví (ne všechny IP pakety nesou protokol TCP, např. pakety ICMP se nekomprimují). Je tedy nutné rozlišovat v přenášených paketech pakety s komprimovaným TCP/IP záhlavím a pa-

kety s nekomprimovaným záhlavím. Proto v záhlaví PPP-rámce v poli protokol mají nekomprimované pakety identifikaci 0021_{16} a pakety s komprimovaným IP-záhlavím identifikaci $002d_{16}$.

Příklad rámců protokolu IPCP (Configure Request a Configure Ack) odchycených MS Network Monitorem ve Windows 2000:

```
  Frame: Base frame properties
      Frame: Time of capture = 3/15/2000 9:39:21.948
      Frame: Time delta from previous physical frame: 0 microseconds
      Frame: Frame number: 37
      Frame: Total frame length: 42 bytes
      Frame: Capture frame length: 42 bytes
      Frame: Frame data: Number of data bytes remaining = 42 (0x002A)
  PPP: Unknown Frame (0x0)
      PPP: Destination Address =  SEND_
      PPP: Source Address =  SEND_
      PPP: Protocol = Internet Protocol Control Protocol
  IPCP: Configuration Request, Ident = 0x07
      IPCP: Code = Configuration Request
      IPCP: Identifier = 7 (0x7)
      IPCP: Length = 28 (0x1C)
      IPCP: Option: Compression Protocol = 0x002D (Van Jacobson Compressed TCP/IP)
          IPCP: Option Type = Compression Protocol
          IPCP: Option Length = 6 (0x6)
          IPCP: Compression Protocol = Van Jacobson Compressed TCP/IP
          IPCP: Max Slot ID = 15 (0xF)
          IPCP: Comp Slot ID = Slot Identifier may be compressed
      IPCP: Option: Address = 195.47.37.205
          IPCP: Option Type = Address
          IPCP: Option Length = 6 (0x6)
          IPCP: Source Address = 195.47.37.205
      IPCP: Option: Primary DNS Server Address = 194.149.105.18
          IPCP: Option Type = Primary DNS Server Address
          IPCP: Option Length = 6 (0x6)
          IPCP: Primary DNS Server Address = 194.149.105.18
      IPCP: Option: Secondary DNS Server Address = 194.149.103.201
          IPCP: Option Type = Secondary DNS Server Address
          IPCP: Option Length = 6 (0x6)
          IPCP: Secondary DNS Server Address = 194.149.103.201

00000:  20 53 45 4E 44 07 20 53 45 4E 44 07 80 21 01 07    SEND. SEND.^!..
00010:  00 1C 02 06 00 2D 0F 01 03 06 C3 2F 25 CD 81 06   .....-....Ă/%Í£.
00020:  C2 95 69 12 83 06 C2 95 67 C9                     Â•i.ı.Â•gÉ

*******************************************************************************
Frame: Base frame properties
      Frame: Time of capture = 3/15/2000 9:39:22.69
      Frame: Time delta from previous physical frame: 120172 microseconds
      Frame: Frame number: 38
      Frame: Total frame length: 42 bytes
      Frame: Capture frame length: 42 bytes
```

```
      Frame: Frame data: Number of data bytes remaining = 42 (0x002A)
  PPP: Unknown Frame (0x0)
      PPP: Destination Address =  RECV_
      PPP: Source Address =  RECV_
      PPP: Protocol = Internet Protocol Control Protocol
  IPCP: Configuration Acknowledgement, Ident = 0x07
      IPCP: Code = Configuration Acknowledgement
      IPCP: Identifier = 7 (0x7)
      IPCP: Length = 28 (0x1C)
      IPCP: Option: Compression Protocol = 0x002D (Van Jacobson Compressed TCP/IP)
          IPCP: Option Type = Compression Protocol
          IPCP: Option Length = 6 (0x6)
          IPCP: Compression Protocol = Van Jacobson Compressed TCP/IP
          IPCP: Max Slot ID = 15 (0xF)
          IPCP: Comp Slot ID = Slot Identifier may be compressed
      IPCP: Option: Address = 195.47.37.205
          IPCP: Option Type = Address
          IPCP: Option Length = 6 (0x6)
          IPCP: Source Address = 195.47.37.205
      IPCP: Option: Primary DNS Server Address = 194.149.105.18
          IPCP: Option Type = Primary DNS Server Address
          IPCP: Option Length = 6 (0x6)
          IPCP: Primary DNS Server Address = 194.149.105.18
      IPCP: Option: Secondary DNS Server Address = 194.149.103.201
          IPCP: Option Type = Secondary DNS Server Address
          IPCP: Option Length = 6 (0x6)
          IPCP: Secondary DNS Server Address = 194.149.103.201

00000:  20 52 45 43 56 07 20 52 45 43 56 07 80 21 02 07    RECV. RECV.^!..
00010:  00 1C 02 06 00 2D 0F 01 03 06 C3 2F 25 CD 81 06   .....-....Ă/%Í£.
00020:  C2 95 69 12 83 06 C2 95 67 C9                     Â•i.ı.Â•gÉ
```

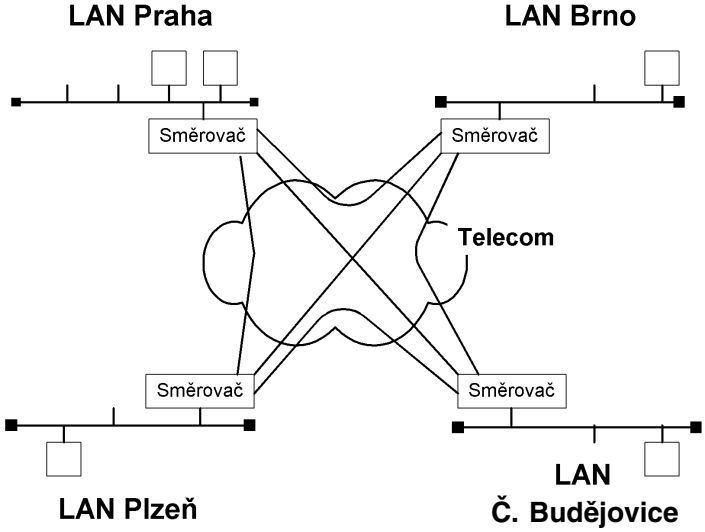
4.5 Frame Relay

Frame Relay je protokol linkové vrstvy pro rozsáhlé sítě. Je specifikován zejména v normách: I.122, I.441 a ANSI TI.606.

Frame Relay používá virtuální okruhy. Virtuální okruh je obdobou pevné linky. Uživatel si od provozovatele sítě Frame Relay pronajímá virtuální okruhy mezi svými jednotlivými lokalitami.

Frame Relay je datagramová nespojovaná služba, tj. jsou jí přenášeny nečíslované rámce. Doručení rámce obecně není provozovatelem garantováno. Každý rámec obsahuje kontrolní součet, lze tedy ověřovat došlo-li během přenosu k narušení paketu. Narušený paket se zahazuje.

Základním parametrem virtuálního okruhu je množství dat, které může uživatel v časovém intervalu T_c předat virtuálnímu okruhu. Tuto veličinu budeme dále označovat jako šíři přenosového pásma (*bandwidth interval*) a označovat ji budeme B_c. Častěji se však používá poměr B_c/T_c, který se označuje jako CIR (*Committed Information Rate*). CIR vyjadřuje kolik dat může uživatel předat sítí Frame Relay za jednotku času. CIR je abstrakce, protože nikdy nelze přesně za 1 vteřinu předat tolik bajtů, kolik vyjadřuje zprůměrovaný CIR. Data se totiž síti předávají v celých rámcích – nikoliv v jejich částech.



Obr. 4.19
WAN založená na pevných linkách

Zajímavou vlastností je tzv. šíře pásma podle potřeby. Jinými slovy: uživatel se dohodne s poskytovatelem na CIR (např. 64 kb/s). Jenže v případě špičky může tuto hranici i překročit. Vše je pochopitelně za určitý poplatek, takže kromě CIRu se uživatel dohodne i na další veličně B_e (*Excess Burst Rate*). B_e vyjadřuje o kolik bajtů je možné v časovém intervalu T_c veličinu B_c překročit.

Frame Relay je určen pro rychlosti od 56 kb/s do 2 Mb/s. Je však efektivní ještě při rychlostech okolo 100 Mb/s.

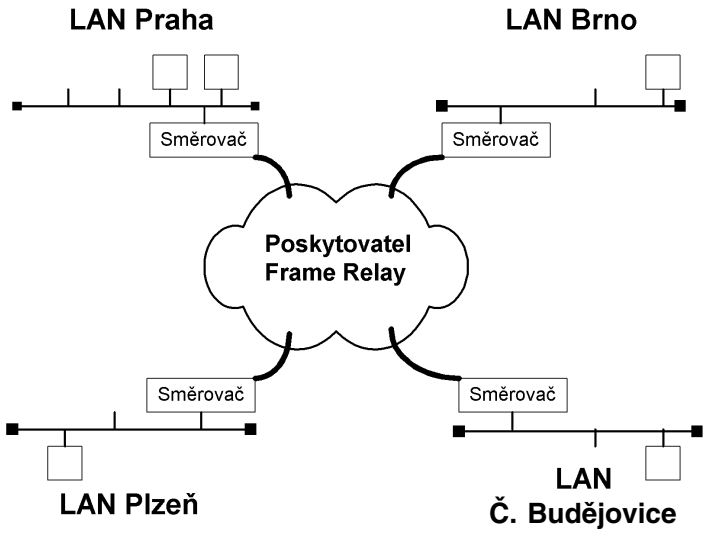
Pronajmete-li si pevné linky od provozovatele veřejné telefonní sítě, pak pro propojení např. čtyř lokalit vzájemně mezi sebou potřebujete čtyři pevné linky. V každé lokalitě budete mít tři modemy a obsazená tři rozhraní na směrovači. Obrázek 4.19 znázorňuje rozsáhlou síť hypotetické firmy, která je situována v Praze, Plzni, Č. Budějovicích a Brně.

Naproti tomu poskytovatel sítě Frame Relay provozuje vlastní datovou síť. Uživatel (zákazník) je na tuto síť napojen v jednotlivých lokalitách zpravidla jednou linkou o vyšší kapacitě. Po této lince svěřuje své datové rámce provozovateli sítě Frame Relay a očekává, že je obdrží v jiné své lokalitě. Provozovatelova síť se skládá z tzv. přepínačů Frame Relay, které si mezi sebou předávají svěřené rámce. Uživatele vcelku nezajímá přes jaké přepínače jde jeho rámec. Dokonce jej ani nezajímá, zdali uvnitř provozovatelova síť používá protokol Frame Relay nebo ne.

Pro propojení jednotlivých lokalit stačí v každé lokalitě jedno rozhraní na směrovači a jedna linka na nejbližší přístupový bod poskytovatele Frame Relay. Může jít např. o pevnou linku (např. E1), radioreleový nebo družicový spoj.

Mezi jednotlivými lokalitami uživatele vzniknou virtuální okruhy. Vraťme se k našemu obrázku: vzniknou virtuální okruhy Praha-Brno, Plzeň-Praha, Brno-Č. Budějovice, Praha-Č. Budějovice, Plzeň-Č. Budějovice a Plzeň-Brno. Vznikne síť s topologií shodnou se sítí, kdy byly jednotlivé lokality propojeny pevnými linkami. Avšak místo pevných linek máme virtuální okruhy. Fyzická linka spojující např. lokalitu Praha se sítí Frame Relay slouží současně třem virtuálním okruhům (do zbývajících lokalit).

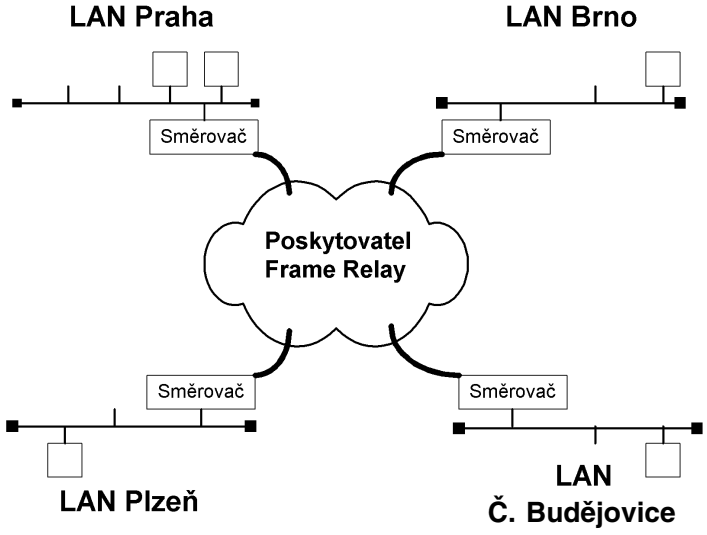
Obr. 4.20
WAN založená na Frame Relay



Provozovatel sítě Frame Relay vytvořil na přání zákazníka pomocí virtuálních okruhů rozsáhlou privátní síť. Avšak provozovatel Frame Relay má více zákazníků a pro každého na jeho přání vytvoří jinou privátní WAN.

Klasický případ připojení zákazníka k síti Frame Relay není, že by se k síti Frame Relay připojoval přímo počítač. K Frame Relay se připojuje směrovač a jednotlivé počítače jsou v lokalitě se směrovačem připojeny pomocí lokální sítě.

Obr. 4.21
Připojení k poskytovateli Frame Relay

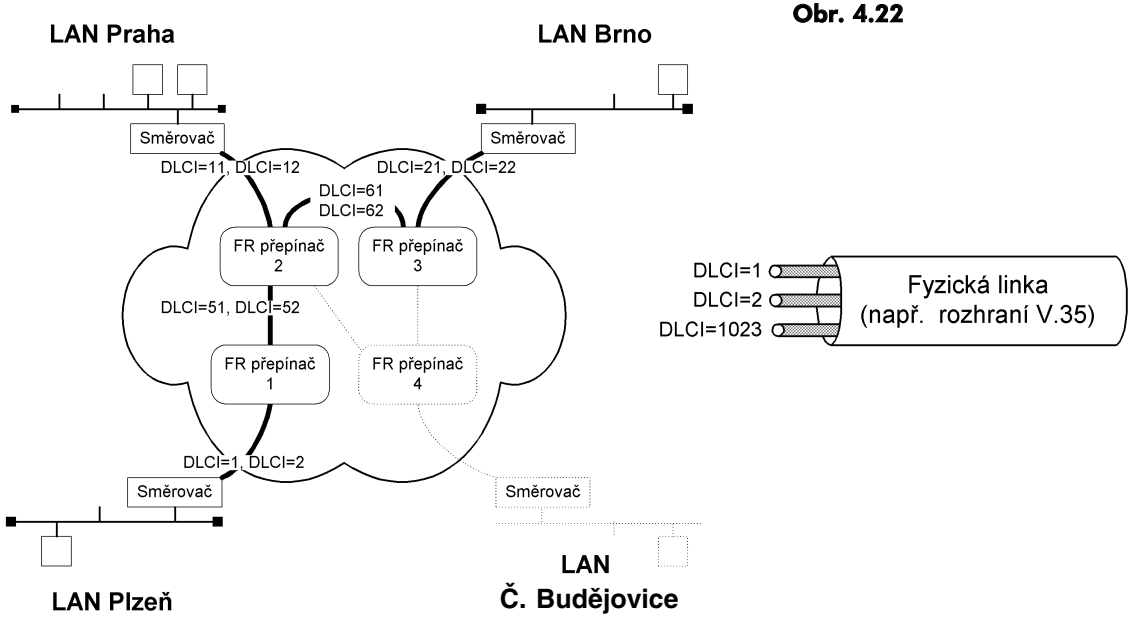


Mezi směrovačem uživatele a nejbližším přepínačem Frame Relay je definováno rozhraní uživatel-síť. Na tomto rozhraní svěřuje své rámce uživatel provozovateli sítě. Právě vůči tomuto rozhraní má Frame Relay vlastnosti, o kterých se zmiňujeme. Co je uvnitř, to uživatele nezajímá. Je to zcela obdobné jako u datových sítí na bázi protokolu X.25, X.25 je také jen rozhraní mezi uživatelem a provozovatelem sítě.

Na fyzické vrstvě se u sítí Frame Relay používá rozhraní V.35, X.21 apod. Používají se pevné okruhy se synchronním přenosem.

Rámec má na cestě po virtuálním okruhu od zdroje k cíli celou řadu linek, kterými prochází. Mezi uživatelem a provozovatelem je linka (rozhraní uživatel-síť). Dále má na cestě jednotlivé linky od jednoho přepínače Frame Relay ke druhému. Na druhém konci následuje opět rozhraní uživatel-síť. Každý virtuální okruh je identifikován tzv. DLCI. DLCI je součástí záhlaví rámce Frame Relay. Jdou-li z nějaké lokality uživatele např. dva virtuální okruhy do různých lokalit, pak jednotlivé lokality se určují v záhlaví rámce pomocí DLCI.

Obr. 4.22



Na obr. 4.22 rámec na cestě z Plzně do Brna postupně mění své DLCI. Uživatel jej v Plzni odevzdá poskytovateli sítě Frame Relay s vyplněným DLCI=2. Přepínač číslo 1 má nakonfigurováno ve své tabulce, že příchozí rámec s DLCI=2 má změnit DLCI=2 na DLCI=52 ("Brněnská lokalita našeho zákazníka přes přepínač 2"). Přepínač 2 je nakonfigurován tak, že mění DLCI=52 na DLCI=62 ("Brněnská lokalita našeho zákazníka přes přepínač 3"). Přepínač 3 je zase nakonfigurován tak, že změní DLCI=62 na DLCI=22 a rámec předá uživateli.

Z hlediska zákazníka je situace ale jednodušší. Když chce z Plzně poslat rámec do Brna, tak v Plzni vyplní v záhlaví rámce DLCI=2 a ví, že jej v Brně obdrží s DLCI=22. A odevzdá-li v Plzni rámec s DLCI=1, pak jej obdrží v Praze s DLCI=11. Obráceně, odevzdá-li v Praze rámec s DLCI=11, pak jej obdrží v Plzni s DLCI=1.

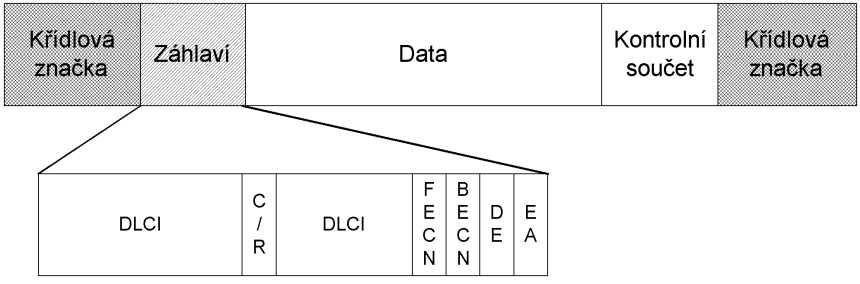
DLCI se mohou používat v síti Frame Relay:

- Jednoznačná v rámci celé sítě provozovatele Frame Relay (viz náš předchozí obrázek).
- Jednoznačná v rámci jednoho přepínače Frame Relay.
- Jednoznačná v rámci celé sítě provozovatele Frame Relay, ale jiná (např. delší) než směrem k uživateli (jiná uvnitř sítě a jiná na rozhraní uživatel-síť).
- Provozovatel provozuje síť na jiném protokolu, ale na rozhraní uživatel-síť se tváří jako Frame Relay, takže vůči uživateli musí použít nějaká DLCI, ale ta s jeho sítí více méně nesouvisí.

4.5.1 Rámec protokolu Frame relay

Rámec protokolu Frame Relay na rozdíl od protokolu HDLC nemá samostatné adresní a řídící pole. Má společné pole záhlaví, které obsahuje identifikaci virtuálního okruhu (DLCI) a další řídící informace. Záhlaví je dlouhé jeden až čtyři bajty. Na obrázku 4.23 je znázorněno dvoubajtové záhlaví.

Obr. 4.23
Formát rámce Frame Relay



Každý bajt záhlaví obsahuje bit **EA**, který určuje, zdali následující bajt je součástí záhlaví nebo přenášených dat. Je-li EA=0, pak i následující bajt je součástí záhlaví, je-li EA=1, pak je tento bajt posledním bajtem záhlaví.

Pole **DLCI** je identifikací virtuálního okruhu, jedná se o obdobu adresy v protokolu HDLC.

Bit **C/R** určuje, zdali jde o příkaz (C) nebo odpověď (R).

Nastavením bitu **DE** se signalizuje, že rámec se má zahodit.

Zbývají bity **FECN** a **BECN**. I když nastavování těchto bitů je nepovinné, tak se u nich zastavíme. Pomocí těchto bitů se řeší problém zahlcení virtuálního okruhu. Posílají-li se data virtuálním okruhem z jednoho konce do druhého, pak nevadí, když se nějaký paket ztratí (např. na úrovni protokolu TCP se přenos zopakuje). Problém je ale se zahlcením spoje, tj. je-li na cestě k cíli nějaké úzké místo, které není schopno rámce takovou rychlostí posílat dále. Rámce se takovému uzlu hromadí ve vyrovnávací paměti (ve frontě), až dojde k vyčerpání fronty a další rámce se musí zahazovat. Tuto situaci nazývám zahlcením linky.

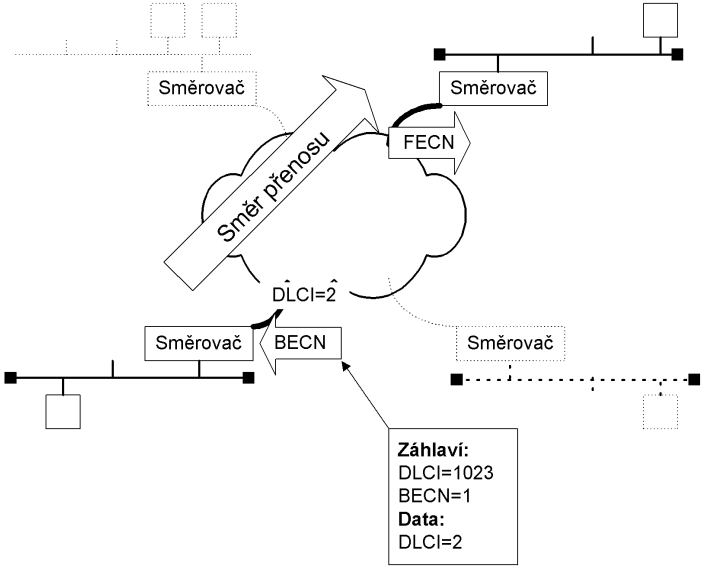
Ztráta rámců pro vyšší vrstvy znamená, že si musí vyžadovat opakování přenosu paketů vyšší vrstvy, tj. zopakování vysílání ztracených rámců. Nebo dokonce může znamenat ztrátu spojení, tj. spojení je nutné obnovovat. V každém případě to znamená zvýšení objemu přenášených dat.

Dojde-li k zahlcení linky, pak každé další i drobné zvýšení provozu zahlcení ještě prohlubuje a linka se stává čím dál méně průchodnou. Na úrovni protokolu TCP se musí stále znovu a znovu obnovovat

komunikace až postupně dojde k rozpadnutí spojení. Koncový uživatel je rozzloben – myslí si, že nastala porucha sítě.

Řešení spočívá v tom, že se zvýší doba odezvy na virtuálním okruhu, tj. virtuální okruh se už na vstupu bude tvářit, že má menší propustnost. Virtuální okruh tedy odebírá rámce od uživatele sice menší rychlostí, ale odebrané rámce se snaží doručit. Uživateli se virtuální okruh jeví jako pomalejší, ale spojení mu nepřipadá porouchané.

V případě zahlcení virtuálního okruhu signalizuje síť odesílateli zahlcení nastavením bitu BECN a příjemci signalizuje zahlcení nastavením bitu FECN (odesílatelem a příjemcem je myšlen uživatelův směrovač na rozhraní uživatel-síť). Odesílání rámců s nastaveným bitem BECN (resp. FECN) síť provádí v okamžiku, kdy je zahlcena, tj. kdy musí zahazovat rámce. Síť může však zahlcení i předpovídat, když při kontrole front nahromaděných rámců na uzlech sítě Frame Relay zjistí, že některá fronta je blízká vyčerpání.

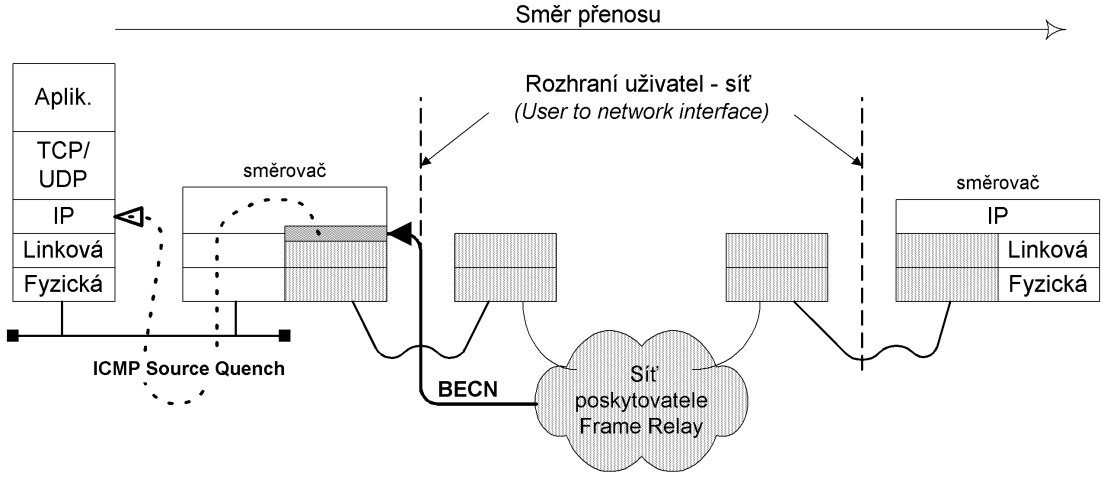


Obr. 4.24 Signalizace přehlcení virtuálního okruhu

Nastavení bitů BECN a FECN se neprovádí u rámců běžných datových okruhů (datových DLCI). Pro tuto signalizaci je rezervováno služební DLCI=1023, jehož rámce síť Frame Relay odesílá uživateli na rozhraní uživatel-síť. V datové části má takovýto služební rámec strukturu odvozenou od rámce XID protokolu HDLC (U-rámec XID protokolu HDLC slouží pro příkazy a odpovědi nesoucí konfigurační informace). V našem případě rámec XID nese číslo DLCI, které odpovídá zahlcenému virtuálnímu okruhu. Obrázek 4.24 vyjadřuje situaci, kdy postižený virtuální okruh má u odesílatele DLCI=2.

Problém je v tom, že uživatelův přístupový směrovač dostane pomocí bitu BECN informaci o tom, že konkrétní virtuální okruh je zahlcen. Jak ale může směrovač snižovat zátěž linky? Musí být dostatečně inteligentní, aby vyšší vrstvě signalizoval zahlcení.

V případě Internetu se jako vyšší vrstva používá protokol IP (Na obr. 4.25 to je nejvyšší vrstva směrovače). Směrovač v sobě musí obsahovat podporu, která je na pomezí Frame Relay a IP.



Obr. 4.25 Signalizace BECN

Podpora ošetření zahlcení okruhu může spočívat v různých mechanismech. Protokol IP má vlastní mechanismus pro ošetření zahlcení linek pomocí protokolu ICMP. Směrovač, který je nucen zahodit IP-paket z důvodu zahlcení, o tom informuje odesílatele IP-paketu ICMP-datagramem *Source Quench*. Příjemce po obdržení ICMP-datagramu *Source Quench* snižuje rychlost příslušného TCP spoje. (Poznamenejme, že signalizace *Source Quench* se u protokolu UDP nepoužívá.)

Směrovač rovněž v pravidelných intervalech zjišťuje na každém virtuálním okruhu podíl přijatých rámců s nastaveným bitem BECN. Je-li výskyt rámců s nastaveným bitem BECN významný, pak na příslušném virtuálním okruhu začne na příchozí pakety generovat odpovědi *Source Quench* v protokolu ICMP.

Jsou i jiné možnosti. Např. přístupový směrovač rámce s nastaveným příznakem BECN vůbec nezpracovává. Pak na úrovni protokolu TCP/IP může docházet až ke ztrátám TCP segmentů. Spoj se může jevit jako poruchový.

Další možností je, že přístupový směrovač uživatele umí pracovat nejen s třetí vrstvou (IP protokol), ale i se čtvrtou vrstvou (protokol TCP). Pak by mohl přímo v záhlaví TCP segmentů opravit délku okna nebo by mohl uměle pozdržet odpovědi od protější strany. Zdroj by si tedy myslel, že linka k cíli má delší odezvu a automaticky by snížil rychlost odesílání TCP-paketů. Zásah do protokolu TCP na směrovači se však obecně považuje za nekorektní.

4.5.2 Závěr k protokolu Frame Relay

Zákazník se s poskytovatelem Frame Relay zpravidla dohaduje na:

- Lokalitách, mezi kterými se vytváří jednotlivé virtuální okruhy.
- Na velikosti přenosových intervalů (CIR) na těchto okruzích a na veličině B_e určující, o kolik může ve špičkách CIR překračovat.
- Na tom, zdali poskytovatel nastavuje bity BECN a FECN. Uživatel si musí rozmyslet, jak je využije – zda uživatelovy směrovače umí tyto bity využít.
- Na lince spojující uživatele a poskytovatele (pevná linka, radioreleový spoj atd.). Většinou poskytovatel Frame Relay dodává i linky spojující uživatele s jeho sítí. Pak je velice důležité použití fyzického rozhraní (V.35, X.21, ...), abyste věděli, jakou si máte koupit propojovací šňůru k vašemu směrovači.

Častým dotazem je: jaký je rozdíl mezi Frame Relay a veřejnou sítí X.25. Frame Relay je protokol pouze linkové vrstvy (X.25 je protokol síťové vrstvy), tj. uživatelé Frame Relay nemají nějakou celosvětově jednoznačnou adresu. Druhou odlišností je, že Frame Relay je datagramovou službou, tj. obecně není garantováno doručení rámce. Protokol X.25 si ukládá data, která nestačí zpracovávat do paměti a postupně je předává dále. Společnou vlastností je, že oba protokoly vytvářejí pro uživatele virtuální okruhy.

4.6 ATM

Protokol ATM má velké ambice. Jeho cílem je být univerzálním linkovým a síťovým protokolem. Jako síťový protokol umožňuje síťovým rozhraním používat celosvětově jednoznačné (globální) síťové adresy. Jeho přívrženci sní o tom, že by mohla globální celosvětová síť založená na ATM nahradit Internet.

ATM se orientuje na virtuální okruhy. Umožňuje spojení na dvoubodových spojích (*point-to-point*) a na spojích, kdy jeden odesílatel adresuje více adresátů (*point-to-multipoint*).

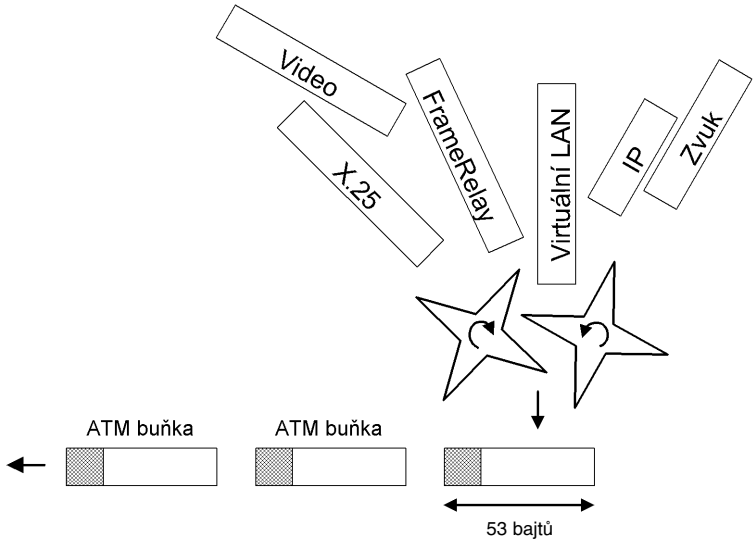
ATM přenáší: zvuk, video, rámce datových sítí tvořících virtuální okruhy (Frame Relay, X.25) i datových sítí netvořících virtuální okruhy, jako je např. IP-protokol.

Jednou z aplikací protokolu ATM je emulace lokálních sítí (tzv. LAN emulace), kdy ATM nahrazuje klasické protokoly pro LAN jako je Ethernet či FDDI. LAN emulace plně nahrazuje protokoly LAN, a to i včetně jejich nedostatků. Emulace LAN přináší ale i výhody – např. mohou se využít vysoké rychlosti ATM či emulovaná LAN může být tvořena geograficky vzdálenými stanicemi.

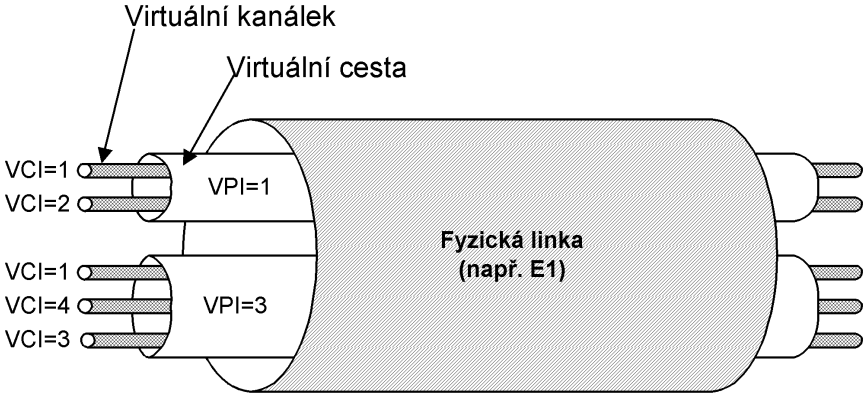
ATM využívá asynchronní přenos (viz kap. 1.3.3), veškerá přenášená data se na vstupu rozdělí do poměrně krátkých buněk, které jsou dlouhé 53 bajtů, jak je znázorněno na obr. 4.26. Každá buňka přenáší 48 bajtů dat. Veškerá vstupní data (zvuk, video, privátní sítě, spojované i nespojované sítě) se rozdělí do buněk (viz obr. 4.26). Síť ATM pak přenáší buňky směrem k příjemci. Příjemce pak z dopravených buněk skládá původní informaci.

ATM používá k přenosu dat fyzické linky (např. E1, E3 atp.). Logicky je fyzická linka rozdělena na virtuální cesty (*Virtual Path* – VP). Jak je znázorněno na obr. 4.27, každá virtuální cesta je dále rozdělena na virtuální kanálky *(Virtual Channel* – VC).

Obr. 4.26
ATM rozděluje pakety různých protokolů do jedné standardní ATM buňky o velkosti 53 bajtů



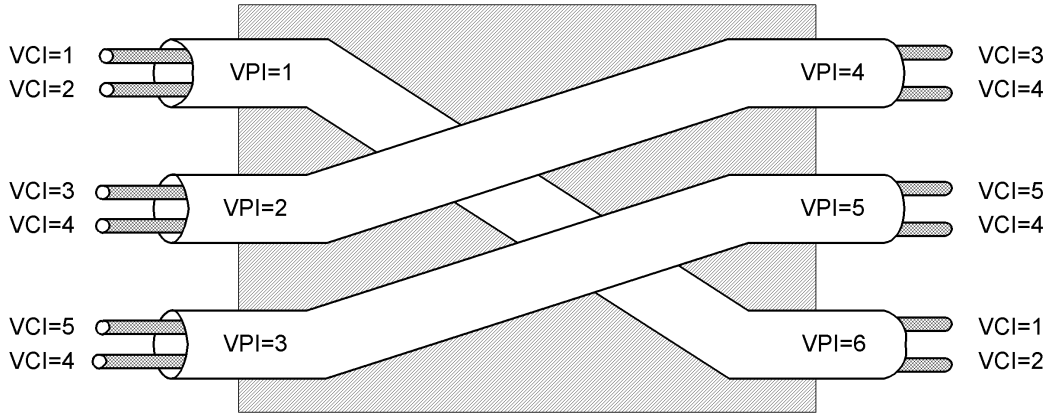
Obr. 4.27
Virtuální cesty a virtuální kanálky



Každá ATM buňka ve svém záhlaví nese identifikaci virtuální cesty (*Virtual Path Identifier* – VPI) a virtuálního kanálku (*Virtual Channel Identifier* – VCI). Virtuální cesty i virtuální kanálky jsou jednostranné cesty, tj. pro oboustrannou komunikaci musí být pro každý směr komunikace zřízena samostatná cesta.

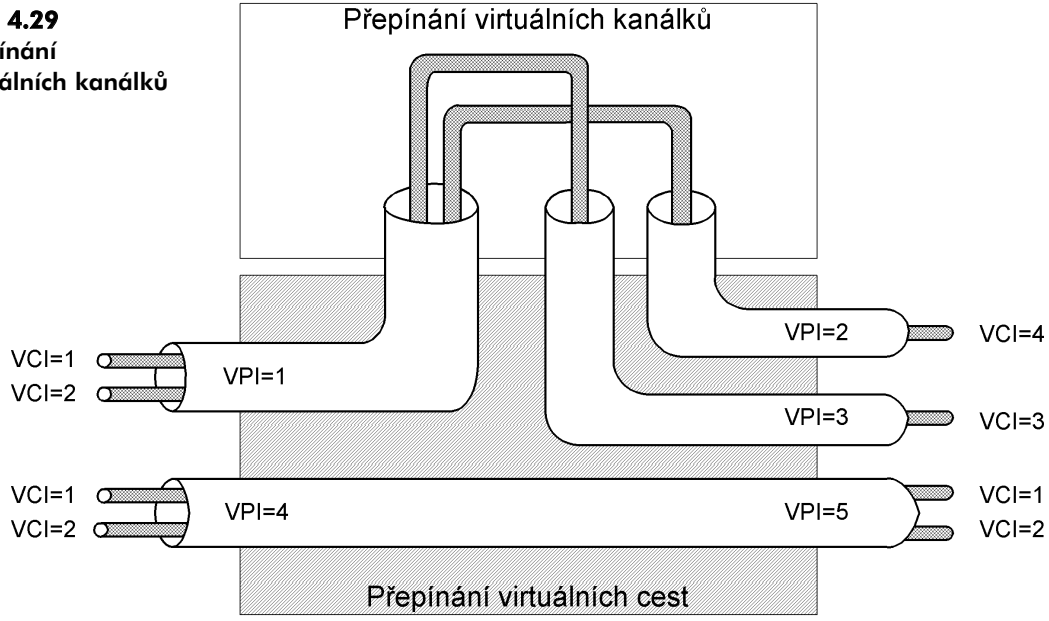
Jádrem sítě ATM jsou tzv. přepínače ATM, které přepínají virtuální cesty mezi sebou. Na obrázku 4.28 je schématicky znázorněn přepínač ATM přepínající virtuální cesty. Takovýto přepínač ATM přepisuje identifikaci VPI v záhlaví buňky. Na obr. 4.28 přepisuje např. VPI=1 na VPI=6. Identifikace virtuálních kanálků zůstávají zachovány.

Přepínače ATM mohou přepínat nejenom virtuální cesty, ale i virtuální kanálky, jak je znázorněno na obr. 4.29. Takový přepínač se skládá ze dvou vrstev. Spodní vrstva přepíná virtuální cesty, horní pak přepíná i virtuální kanálky.



Obr. 4.28 **Přepínač ATM přepínající virtuální cesty**

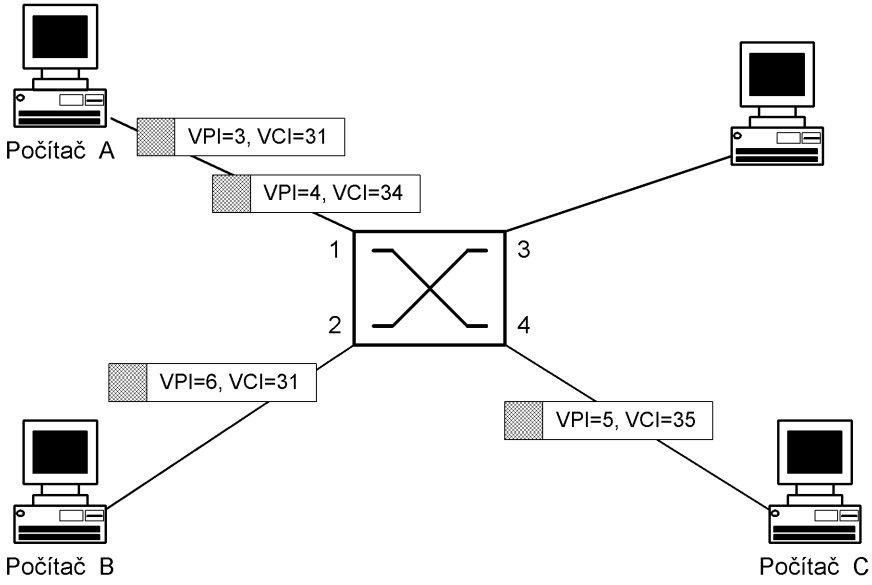
Obr. 4.29
Přepínání virtuálních kanálků



Příchozí (resp. odcházející) buňka ATM je v přepínači ATM identifikována třemi údaji: VPI, VCI a síťovým rozhraním (*interface*) přepínače ATM.

Pro přepínání ATM buněk udržuje přepínač ATM přepínací tabulku, která říká jaké buňky na vstupu se mají přepnout na jaké buňky na výstup. Např. na obr. 4.30 počítač A chce odesílat buňky jak počítači B, tak i počítači C. Počítač A používá pro odesílání buněk na počítač B identifikaci VPI=3/VCI=31; buňky jsou na počítač B doručovány s identifikací VPI=6/VCI=31, tj. dochází pouze k přepínání cesty z VPI=3 na VPI=6.

Obr. 4.30
Přepínání buněk



Přepínací tabulka pak pro náš případ musí mj. obsahovat následující údaje (je třeba si uvědomit, že spoj ATM je jednostrannou komunikací, takže tabulka musí mít minimálně další řádky pro komunikaci v opačném směru).

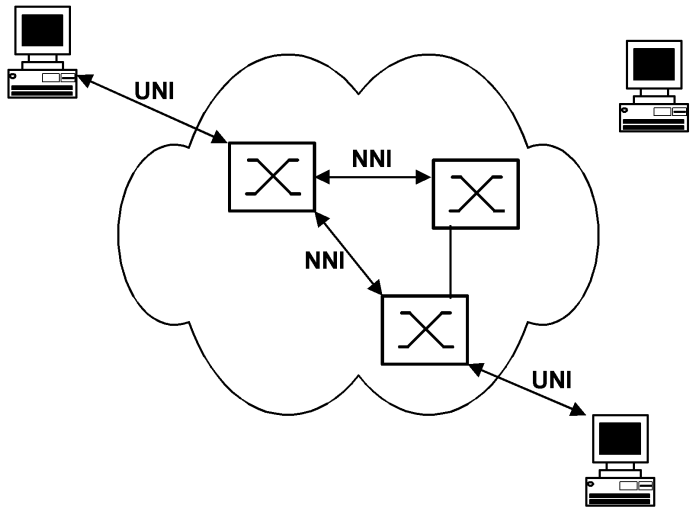
Vstup		Výstup	
Rozhraní	**VPI/VCI**	**Rozhraní**	**VPI/VCI**
1	3/31	2	6/31
1	4-34	4	5/35

Některé kombinace VPI/VCI jsou vyhrazeny pro služební účely.

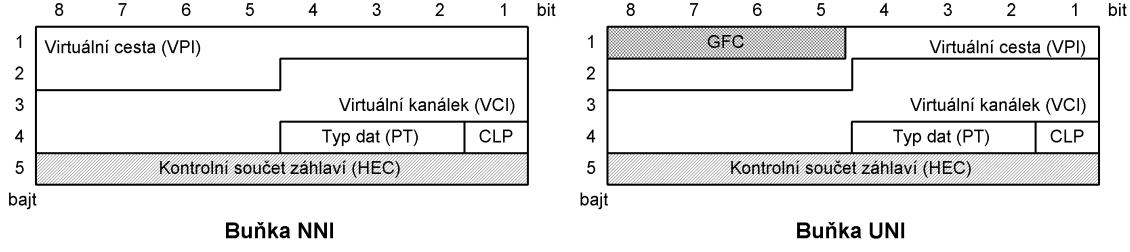
Rozlišujeme dva typy rozhraní (viz obr. 4.31):

- Rozhraní uživatel – síť (*User to Network Interface* – UNI), které je mezi uživatelovým počítačem a přilehlým ATM přepínačem.
- Rozhraní síť – síť (*Network to Network Interface* – NNI) mezi jednotlivými ATM přepínači téže sítě ATM.

Obr. 4.31
Rozhraní UNI a NNI



Struktura hlavičky ATM buňky závisí na skutečnosti, zdali se jedná o buňku na rozhraní UNI nebo NNI. Jak je vidět z obr. 4.32, rozdíl mezi těmito dvěma typy spočívá pouze v tom, že buňka UNI má na úkor pole VPI navíc pole GFC.



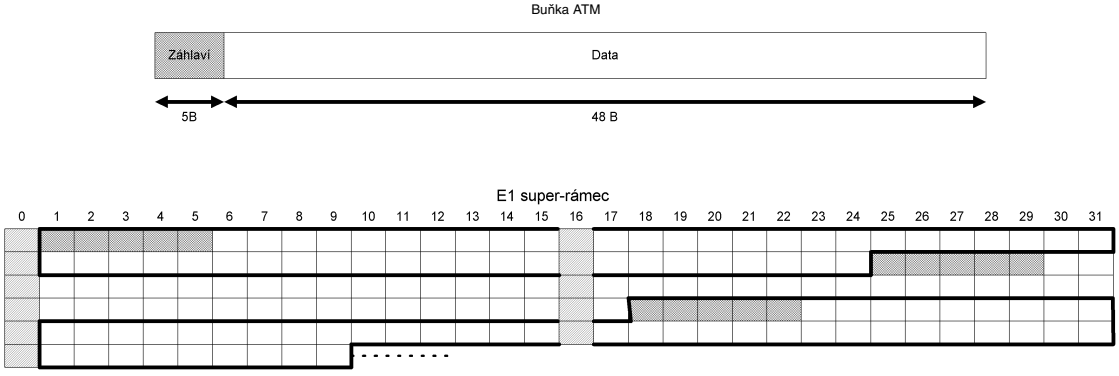
Obr. 4.32 Záhlaví ATM buněk

Význam jednotlivých polí záhlaví:

- GFC (*Generic Flow Control*) je určeno pro lokální komunikaci (např. řízení toku dat), v praxi se většinou nevyužívá – bývá nastaveno na hodnotu 0000_2.
- VPI a VCI obsahuje identifikace virtuální cesty a virtuálního kanálku, ke kterému buňka náleží. Pro buňky NNI je pole VPI o 4 bity delší, což umožňuje v komunikaci mezi ATM přepínači používat větší množství virtuálních cest.
- HEC (*Header Error Control*) obsahuje kontrolní součet. Existují dva druhy HEC:
  - Detekční mód, kdy je buňka při chybě odmítnuta.
  - Korekční mód, kdy je hlavička buňky opravena (předpokládá se chyba v jednom bitu hlavičky).

- CLP (*Cell Loss Priority*) umožňuje první stupeň správy. Buňky s CLP=1 jsou kandidáty na odmítnutí v případě přetížení spoje.
- PT (*Payload Type*) obsahuje informace o přenášených datech. Pole PT je dlouhé 3 bity, označme je *xyz*.
  - Bit *z* je informace o paketu vyšší vrstvy (vrstvy AAL), který je vsunut do buňky. Paket AAL byl vsunut do několika buněk. *z*=1 signalizuje, že se jedná o poslední fragment AAL paketu. První a prostřední fragmenty mají hodnotu *z*=0.
  - Bit *y* signalizuje přetížení spoje.
  - Bit *x* je určen pro další rozlišení buněk.

Buňka ATM se vkládá do super-rámce fyzické vrstvy. Na obr. 4.33 je znázorněna jedna z možností vkládání buňky do super-rámce linky E1 (sloty 0 a 16 slouží pro služební účely).



Obr. 4.33 Vkládání buněk ATM do super-rámce E1 (doporučení ITU G.804)

4.6.1 Vrstvy ATM

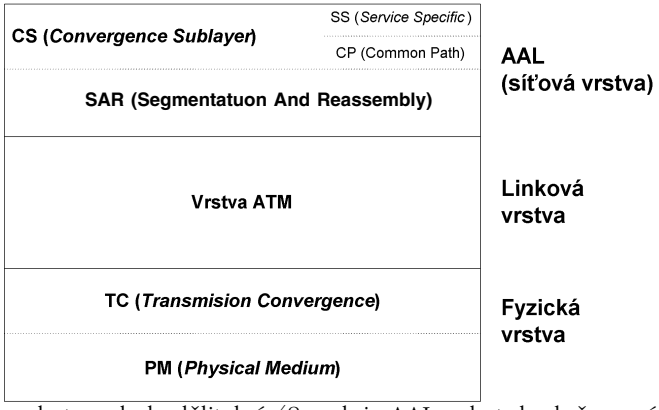
Z hlediska síťového modelu se problematika ATM skládá ze tří vrstev:

1. Fyzické vrstvy, která se skládá z vlastní fyzické vrstvy PM a vrstvy TC. Vrstva TC má na starosti vkládání buněk do super-rámců, vkládání prázdných buněk na nečinné linky, vytváření a ověřování kontrolního součtu záhlaví buňky (HEC) atd.
2. Vrstvy ATM, která je zodpovědná za konstrukci buněk, přepínání buněk, definuje rozhraní UNI a NNI atd.
3. Vrstvy AAL, která adaptuje (přizpůsobuje) služby vyšší úrovně pro vrstvu ATM.

4.6.1.1 AAL

Vrstva AAL balí pakety protokolů vyšších vrstev (např. IP-pakety) do AAL paketu. Paket AAL je posléze segmentován (rozdělen) do jednotlivých buněk ATM. Celý proces se děje ve dvou krocích. Nejprve vrstva CS zabalí paket vyšší vrstvy do AAL paketu jehož délka je dělitelná 48, a posléze vrstva SAR tento AAL paket rozdělí na buňky ATM.

Obr. 4.34 Jednotlivé vrstvy ATM

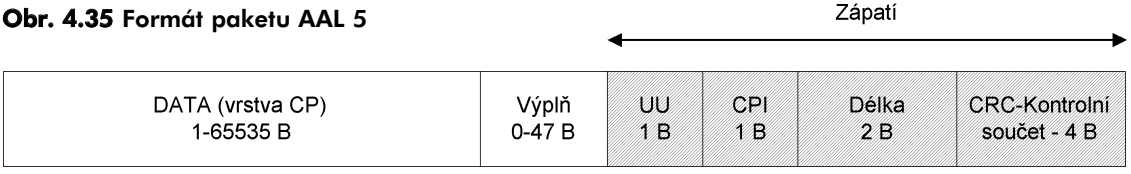


Pokud by délka AAL paketu nebyla dělitelná 48, pak je AAL paket doplněn o výplň – důvodem je skutečnost, že buňka ATM má právě 48 bajtů dlouhou datovou část.

Jelikož vrstva AAL musí zpracovávat nejrůznější typy dat (zvuk, video, datové pakety atd.), tak je vytvořeno několik typů vrstvy AAL:

1. AAL typu 1 pro služby s požadavkem na přenos toku bitů s konstantní šíří přenášeného pásma (např. telefonní hovory).
2. AAL typu 2 pro služby s požadavkem na přenos toku bitů s proměnnou šířkou přenášeného pásma (např. animace, video).
3. AAL typu 3 pro přenos datových paketů síťových služeb vytvářejících virtuální okruhy.
4. AAL typu 4 pro přenos datových paketů síťových služeb nevytvářejících virtuální okruhy.
5. AAL typu 5, které vzniklo zjednodušením AAL 3 a 4. Dále se budeme zabývat pouze AAL 5.

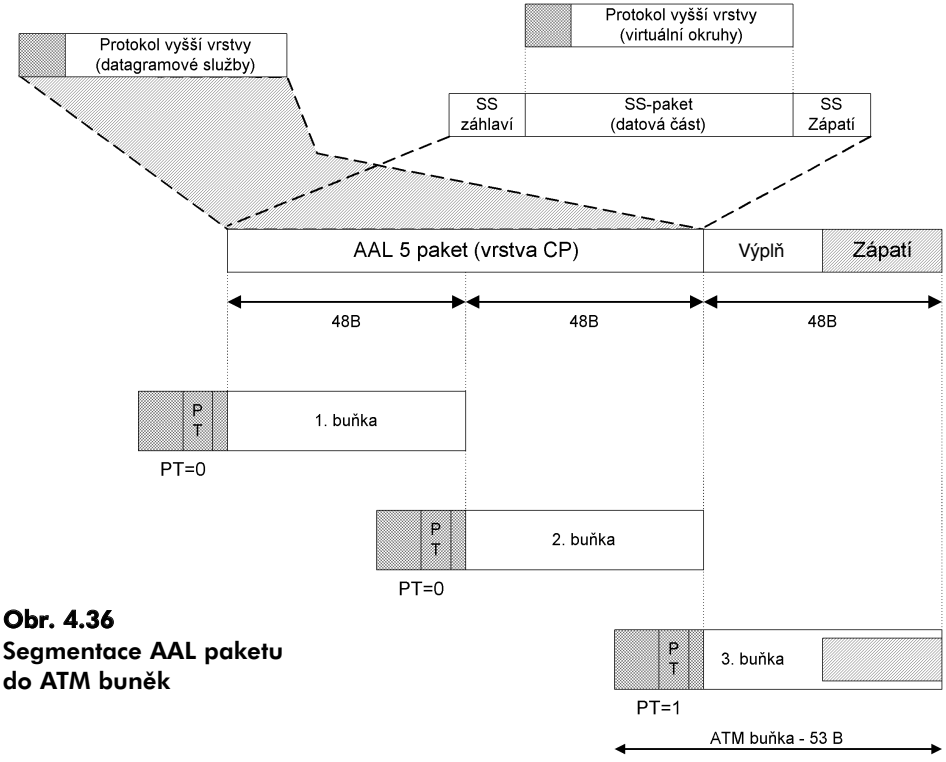
Obr. 4.35 Formát paketu AAL 5



Na obr. 4.35 je znázorněn formát paketu vrstvy AAL typu 5. Výplň slouží k dosažení délky paketu tak, aby byla násobkem 48. Pole UU (*User to User indication*) je určeno pro přenos uživatelské informace. Pole CPI (*Common Part Indicator*) je určeno pro řízení výkonu a monitorování (tč. se nevyužívá). Pole délka obsahuje délku AAL paketu a pole CRC obsahuje kontrolní součet.

Na obr. 4.36 je pak znázorněna segmentace (rozdělení) AAL paketu do buněk ATM. V záhlaví buněk ATM se využívá pole PT. Pole PT je nastaveno na nulu kromě poslední buňky, která je má nastaveno na jedničku. Vrstva AAL typu 5 předpokládá, že buňky ATM nesoucí konkrétní AAL paket jsou sítí ATM přenášeny za sebou, tj. nedochází k záměně pořadí buněk. Pomocí pole PT lze tak i buňku na druhém konci sestavit – datové části přicházejících buněk se skládají za sebe, dokud nedorazí buňka s PT=1, ta se přidá jako poslední a vznikne tak sestavený AAL paket.

Na obr. 4.36 je v AAL paketu uvedeno, že se jedná o datovou část vrstvy CP. Pokud se protokolem ATM mají přenášet datové pakety protokolů využívajících virtuální okruhy, pak je nutné navíc ještě použít vrstvu SS (viz obr. 4.34). Vrstva SS balí pakety protokolů vyšších vrstev do svých paketů majících SS záhlaví a SS zápatí. Až výsledný SS paket je pak balen do paketu CP, jak je znázorněno na obr. 4.36.



Obr. 4.36
Segmentace AAL paketu do ATM buněk

4.6.2 Signalizace

Dosud jsme se zabývali virtuálními okruhy (tj. virtuálními cestami a virtuálními kanálky), které už byly sestaveny a bez jakýchkoliv potíží pracovaly. Jakoby se jednalo o pevné okruhy. ATM však umožňuje virtuální okruhy i dynamicky vytvářet.

K vytváření okruhů slouží signalizace. Signalizace neslouží jen k vytváření okruhů, ale i k signalizaci chybových stavů, k monitorování sítě atd.

Se signalizací pro vytvoření virtuálního okruhu se denně setkáváme při telefonování. Při telefonování nejprve zvolíme na číselníku telefonní číslo a telefonní síť nám na základě volby čísla vytvoří telefonní okruh. Při volbě telefonního čísla sdělujeme číslo telefonní síti, aby nám okruh vytvořila. Tj. při vytváření telefonního okruhu je jedním účastníkem komunikace uživatel a druhým telefonní síť.

Při signalizaci ATM je tomu obdobně. Opět zůstaneme pouze u případu vytvoření virtuálního okruhu, což je jen jednou ze základních funkcí signalizace.

Uživatel pro vytvoření okruhu musí signalizací požádat síť o tuto službu (službu vytvoření virtuálního okruhu). Uživatel je na rozhraní uživatel – síť (UNI). Copak má uživatel se samotnou sítí spojení? To je opravdu problém, jak navázat spojení se sítí, aby uživatel mohl síti signalizovat své požadavky, či aby mu síť předávala nějaké informace.

Signalizace probíhá ve dvou krocích.

1. V prvním kroku se použije tzv. metasignalizace. Vzpomeňme si, že při popisování virtuálních cest a virtuálních kanálků jsme řekli, že cesty a kanálky o některých identifikacích slouží pro služební účely. Metasignalizace je právě takovým služebním účelem. Zpravidla virtuální kanálek

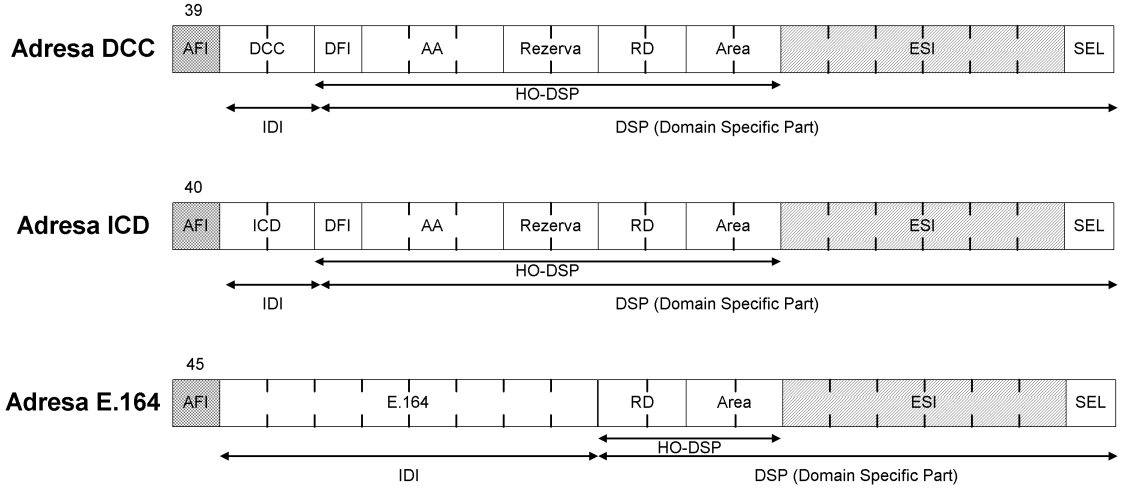
VPI=0/VCI=1 se využívá pro metasignalizaci. Metasignalizací pošle uživatel jednu ATM buňku (její formát je určen doporučením Q.2120), kterou žádá síť o navázání komunikace na vrstvě AAL (např. pro nás AAL typu 5). Síť pak opět jednou buňkou odpoví. Výsledkem je dohoda o spojení na nějakém virtuálním kanálku.

(Existují i jiné varianty signalizace, kdy se informace síti předají jen metasignalizací, pak se zpravidla využívá kanálek VPI=0/VCI=5.)

2. V druhém kroku již existuje komunikace na vrstvě AAL. Do paketu vrstvy AAL se vkládají pakety signalizace (např. dle doporučení Q.2931). Pakety signalizace obsahují jednotlivé zprávy, jako je např. zpráva SETUP nebo CONNECT. Jedná se o zcela analogickou situaci, kterou jsme popsali v kapitole 3.3.1 u signalizace ISDN (ISDN používá doporučení Q.931, jehož pakety jsou velice podobné paketům Q.2931).

Signalizaci můžeme také rozdělovat z pohledu rozhraní, tj. na signalizaci na rozhraní UNI a NNI.

I když se v sítích ATM signalizací ani směrováním dále již nebudeme podrobně zabývat, tak si uveďme alespoň tvar ATM adres.



Obr. 4.37

Na obr. 4.37 jsou nejčastěji používané ATM adresy (nejedná se o standardty ITU, ale ATM fóra). Adresy ATM jsou odvozeny od NSAP – adresy používané protokoly OSI. Adresy jsou dvacetibajtové a skládají se z polí:

- AFI (první bajt) specifikuje typ adresy.
- IDI (*Initial Domain Identifier*) obsahuje
  - pro adresu DCC pole DCC, což je identifikace země podle normy ISO 3166.
  - Pro adresu ICD pole ICD, což je mezinárodní kód podle normy ISO 6523.
  - Pro adresu E.164 pole E.164 obsahující číslo ISDN podle normy E.146.
- DSP (*Domain Specific Part*), které jednoznačně specifikuje adresu v rámci konkrétního IDI. Adresa síťového rozhraní je určena bez pole SEL. Pole SEL (*Selector*) je určeno pro bližší specifika-

ci protokolu vyšší vrstvy, které se mají data předávat. Např. mají-li se předávat LAN emulaci či přímo IP-vrstvě. Obecně se pole DSP dělí na:

- HO-DSP (*High-Order DSP*), což je adresa domény ("adresa sítě")
- ESI (*End Systém Identifier*), což je adresa síťového rozhraní, která je šestibajtová, takže se skutečně do ní např. vejde fyzická adresa protokolu Ethernet ("adresa počítače").

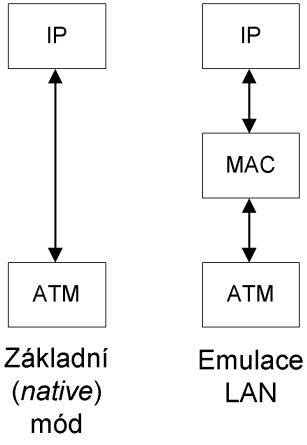
Pole HO-DSP se obecně dělí na pole DFI (*Domain Specific Part Identifier*), AA (*Administrative Authority*) RD (*Routing Domain*) a Area. Toto dělení se však v sítích ATM už příliš nevyužívá.

4.6.3 Emulace LAN

Otázkou je, jak využít síť ATM. V praxi se setkáváme se dvěma typy použití:

1. Základním (*native*) módem, kdy pakety protokolů vyšších vrstev (na obr. 4.38 IP-protokolu) jsou vkládány přímo do ATM, tj. do paketů AAL typu 5 (resp. paketů podvrstvy CP).
2. Druhou možností je emulovat protokoly LAN (např. Ethernet, Token Ring apod.) nad sítí ATM. Pak se IP-datagramy vkládají např. do rámců protokolu Ethernet, který se vkládá do paketů AAL typ 5 (resp. do paketů podvrstvy SS, které se vkládají do paketů podvrstvy CP).

Obr. 4.38 Základní mód a emulace LAN



Emulace lokálních sítí má jistou nevýhodu ve zvýšení režie, protože do dat se vkládají další hlavičky, ale má i své nesporné výhody. Výhodou je, že lze integrovat stávající LAN se sítěmi ATM. Vznikají tak virtuální lokální sítě, které se mohou skládat z počítačů ležících na segmentu klasické LAN plus počítačů připojených v síti ATM kdekoliv na Zemi. Navíc přenosové rychlosti v sítích ATM bývají někdy vyšší než v klasických LAN.

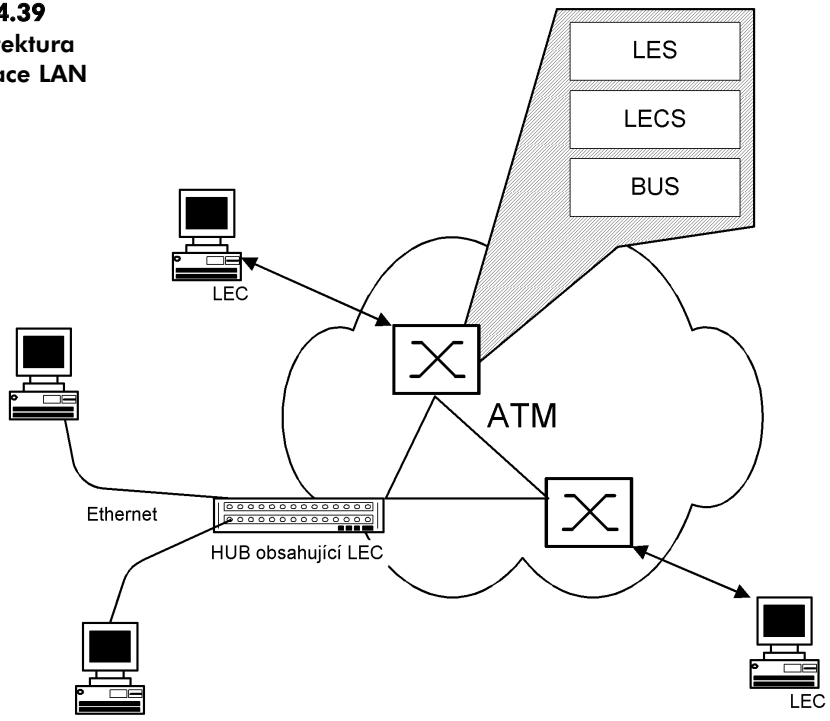
Architektura emulace LAN je znázorněna na obr. 4.39. Architektura se skládá z následujícího softwaru:

- Klienta (*LAN Emulation Client* – LEC), který zajišťuje veškerou funkčnost přenosu dat přes ATM. Musí být v každé koncové stanici emulované LAN přes ATM. Pakliže je stanice na klasickém segmentu LAN, pak LEC je součástí aktivního prvku (HUB). LEC má jednu nebo více jednoznačných adres ATM.

  LEC může být realizován i na síťových kartách.

- Serveru (*LAN Emulation Server* – LES), který řídí konkrétní emulovanou LAN (každá emulovaná LAN musí mít svůj LES). Má jednoznačnou ATM adresu. LES bývá součástí softwaru ATM přepínače.

Obr. 4.39
Architektura emulace LAN



- Konfiguračního serveru (*LAN Emulation Configuration Server* – LECS), který umožňuje konfiguraci a správu emulované LAN. Zde se udržuje databáze klientů, kteří patří konkrétní emulované LAN.
- Serveru pro oběžníky a neznámý provoz (*Broadcast and Unknown Server* – BUS), který zajišťuje oběžníky na LAN a ošetřuje provoz neznámých stanic.

4.6.4 Závěr k ATM

ATM je určeno pro sítě, kde je nutno přenášet různé datové protokoly, video, audio atp. Univerzálnost přináší zvyšování režie. Uvědomme si, že Ethernetový rámec je vložen do paketu emulace LAN, ten je vložen do paketu AAL, který je rozdělen do buňky ATM. Všechny tyto pakety mají jisté záhlaví či zápatí, které zvyšuje režii.

Pokud bychom chtěli např. využít ATM pro dvoubodový spoj o kterém víme, že skrze něj budeme přenášet pouze IP, pak režie může stoupnout až na 30 %, kdežto režie u protokolu HDLC bývá okolo 4 %. A kromě toho pro digitální linky se jeví do budoucna rozumné u takovýchto dvoubodových spojů vkládat IP-datagramy přímo do fyzické vrstvy.

Emulace LAN na dvoubodovém spoji je extrémní případ, ale je dobré si uvědomit, že tento extrémní případ zvýší režii na 30 %. A 30 % na lince 1 Gb/s je opravdu velké číslo.

Naopak v komplikovaných sítích, kde se současně s daty přenáší i audio či video, oceníme robustnost ATM.

4.7 Lokální sítě (LAN)

Lokální sítě se svou přenosovou rychlostí 10 Mb/s až Gb/s se řadí mezi středně rychlé sítě. Cílem LAN je propojit mezi sebou počítače (a jiná komunikační zařízení jako např. směrovače) v rámci jedné nebo několika budov tak, aby mohly vzájemně mezi sebou komunikovat. Při použití optických rozvodů může LAN pokrývat území i několika kilometrů.

V uplynulých deseti letech byla vyvinuta celá řada systémů LAN. Masového rozšíření se však dočkaly jen dva: Ethernet a v menším rozsahu FDDI. (Někdy se ještě setkáváme se systémem Token Ring firmy IBM, ale to spíše v případech, že uživatel je kompletně vybaven systémy firmy IBM.)

Pro připojení stanice na LAN je nutné do stanice vložit příslušnou síťovou kartu. Linkové protokoly LAN jsou realizovány z části přímo v síťové kartě.

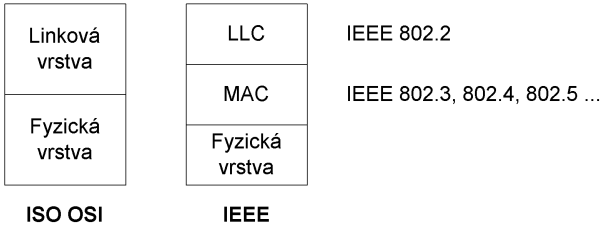
Problematika LAN se vždy skládá z:

- Problematiky kabeláže, která patří do fyzické vrstvy.
- Problematiky síťových karet, které se vkládají do počítačů a ostatních zařízení. To je součást jak fyzické vrstvy, tak i linkové vrstvy, protože část softwaru pro obsluhu linkové vrstvy je realizována přímo na síťové kartě.
- Problematiky samotného linkového protokolu (včetně obsahu linkových rámců) a jeho realizace programy v počítači (ovladači).

Instituce IEEE před dvaceti lety předložila projekt, jehož cílem bylo vypracovat normy pro jednotlivé typy LAN (např. Ethernet, Arcnet, Token Ring atd.). Tyto normy popisovaly pro každý typ LAN vrstvu MAC. Vznikla tak norma IEEE 802.3 pro Ethernet, IEEE 802.4 pro Token Bus, IEEE 802.5 pro Token Ring atd.

Pro všechny systémy pak byla vypracována společná norma pro vrstvu LLC pod označením IEEE 802.2, což schématicky vyjadřuje obr. 4.40.

Obr 4.40



Problematika linkové vrstvy pro LAN tak byla rozdělena do dvou podvrstev. Spodní vrstva Medium Access Control (MAC) částečně zasahující do fyzické vrstvy se zabývá přístupem na přenosové médium. Horní vrstva Logical Link Control (LLC) umožňuje navazovat, spravovat a ukončovat logická spojení mezi jednotlivými stanicemi LAN.

Uvedené normy IEEE byly převzaty později ISO. Z normy IEEE 802.2 tak vznikla norma ISO 8802-3, z normy IEEE 802.3 vznikla norma ISO 8802-3 atd.

4.7.1 Ethernet

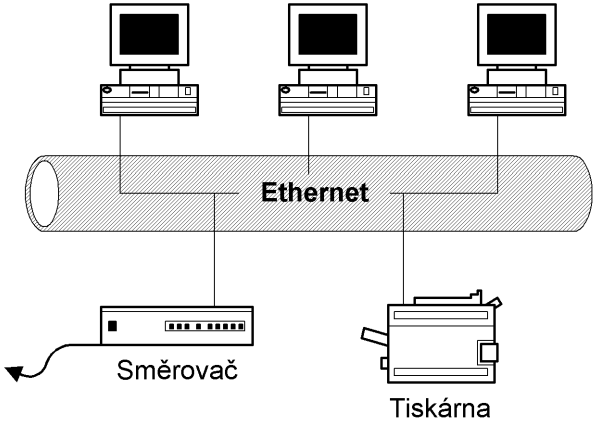
Protokol Ethernet byl původně vyvinut firmami DEC, Intel a Xerox. Jeho varianta 10 MHz se označuje jako Ethernet II. Později byl Ethernet normalizován institutem IEEE jako norma 802.3. Tato norma byla převzata ISO a publikována jako ISO 8802-3. Formát rámců podle normy Ethernet II se mírně odli-

šuje od formátu ISO 8802-3. Postupem času vznikla norma IEEE 802.3u pro Ethernet na frekvenci 100 MHz (Fast Ethernet) a norma IEEE 802.3z pro frekvenci 1 GHz (gigabitový Ethernet).

Původní rozvod Ethernetu by prováděn tzv. tlustým koaxiálním kabelem označovaným 10BASE5. Koaxiální kabel, který mohl být dlouhý maximálně 500 metrů tvořil jeden segment lokální sítě. Segment tlustého Ethernetu (jak se tomuto rozvodu často říkalo) byl většinou tvořen jedním kusem koaxiálního kabelu. Na koaxiální kabel byly napichovány transceivery, které se propojovaly kabelem na AUI-port ethernetové přídavné karty v počítači. AUI-port zpravidla používá konektor CANNON-15.

Označení 10BASE5 vyjadřuje, že se jedná o síť používající přenosovou frekvenci 10 MHz (ta je v případě Ethernetu rovná i teoretické přenosové rychlosti sítě).

Obr. 4.41
Segment Ethernetu tvořený koaxiálním kabelem



Masově se Ethernet rozšířil na tzv. tenkém koaxiálním kabelu. Tenký koaxiální kabel je u každé stanice přerušen a na oba konce přerušení je buď napájen nebo speciálními kleštěmi namáčknut BNC-konektor. Mezi dva BNC-konektory se vloží BNC-T-konektor – "odbočka k počítači". Třetí vývod BNC-konektoru se nasadí přímo na ethernetovou síťovou kartu v počítači (na její BNC-konektor). Existují však i transceivery pro tenký Ethernet, pak se BNC-T-konektor připojí na transceiver pro tenký Ethernet a kabel z transceiveru se připojí na AUI-port počítače.

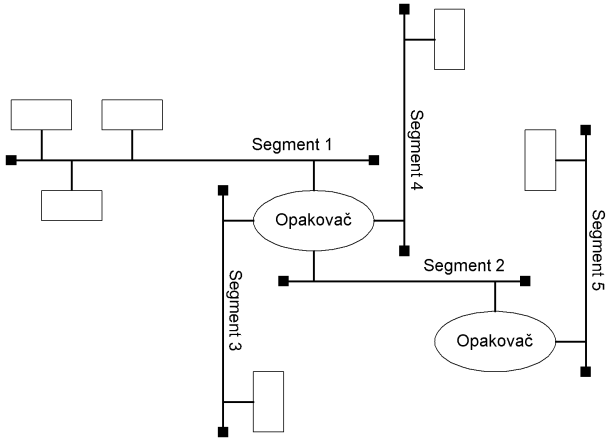
Tenký Ethernet, označovaný jako 10BASE2 může být tvořen segmentem o maximální délce 185 metrů. Použijí-li se na segmentu stejné síťové přídavné karty, pak v případě některých karet je možné segment zvětšit až na 300-400 metrů.

Délka segmentu LAN je tedy 500 (resp. 185 – 300) metrů. Rozsah LAN je možné zvětšit tím, že použijeme více segmentů, které mezi sebou propojíme tzv. **opakovači**. Opakovač je tvořen dvěma nebo více síťovými kartami, které jsou vzájemně propojeny. Objeví-li se nějaký datový rámec na jednom rozhraní, pak je automaticky zopakován na všechny ostatní. Opakovač může být osazen AUI i BNC porty, takže některé segmenty mohou používat tlustý a jiné tenký Ethernet.

Mezi dvěma opakovači může být použita i dvojice optických kabelů, tento typ Ethernetu se někdy označuje jako 10BASE-F. Délka optického propojení dvou opakovačů může být 1 km.

Nyní si řekneme, že opakovač může být osazen i porty pro kroucenou dvojlinku. V případě kroucené dvojlinky je situace trochu odlišná. Kroucená dvojlinka (přesněji řečeno dva páry vodičů) je rozhraní

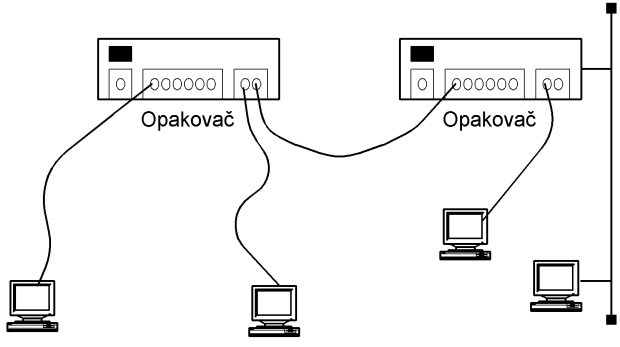
Obr. 4.42
LAN tvořená jednotlivými segmenty



mezi opakovačem a počítačem. Spíše toto rozhraní připomíná rozhraní mezi transceiverem a AUI-konektorem (neobsahuje však napájení).

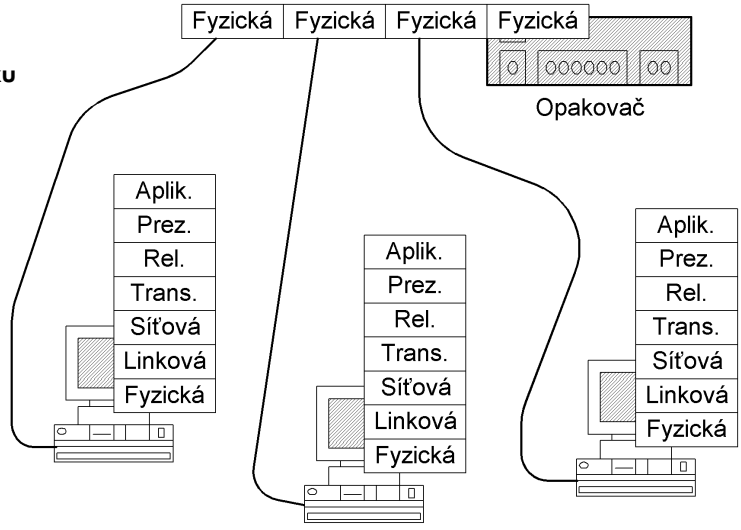
V případě kroucené dvojlinky je jádrem sítě opakovač (na rozdíl od koaxiálního kabelu). Z opakovače se hvězdicovitě rozbíhají kroucené dvojlinky k jednotlivým počítačům. Opakovač pro kroucenou dvojlinku se označuje jako HUB (označení HUB se používalo pro aktivní prvek u sítí s hvězdicovou topologií). HUB může mít pochopitelně i BNC nebo AUI-porty.

Obr. 4.43
Opakovač pro kroucenou dvojlinku (HUB)



Spoj mezi opakovačem a počítačem je tvořen dvěma páry kroucené dvoulinky (4 vodiče). Jedná se o duplexní spoj, kde pro každý kanál je určen jeden pár. Z hlediska počítače je tedy jeden pár "vysílání" a druhý pár "příjem". HUBy pro kroucenou dvojlinku je možné mezi sebou vzájemně propojovat. Ale pozor, co je pro jeden "vysílání", je pro druhý "příjem", takže v propojovací šňůře musí být páry překřížené (jako např. v případě nulových modemů). Většinou se však dodávají HUBy, kde jeden port je osazen přepínačem, který právě způsobí překřížení párů, takže stačí použít „normální" propojovací šňůru a připojit ji do portu s přepínačem a ten přepnout do vhodné polohy.

Obr. 4.44
Opakovač pro kroucenou dvojlinku



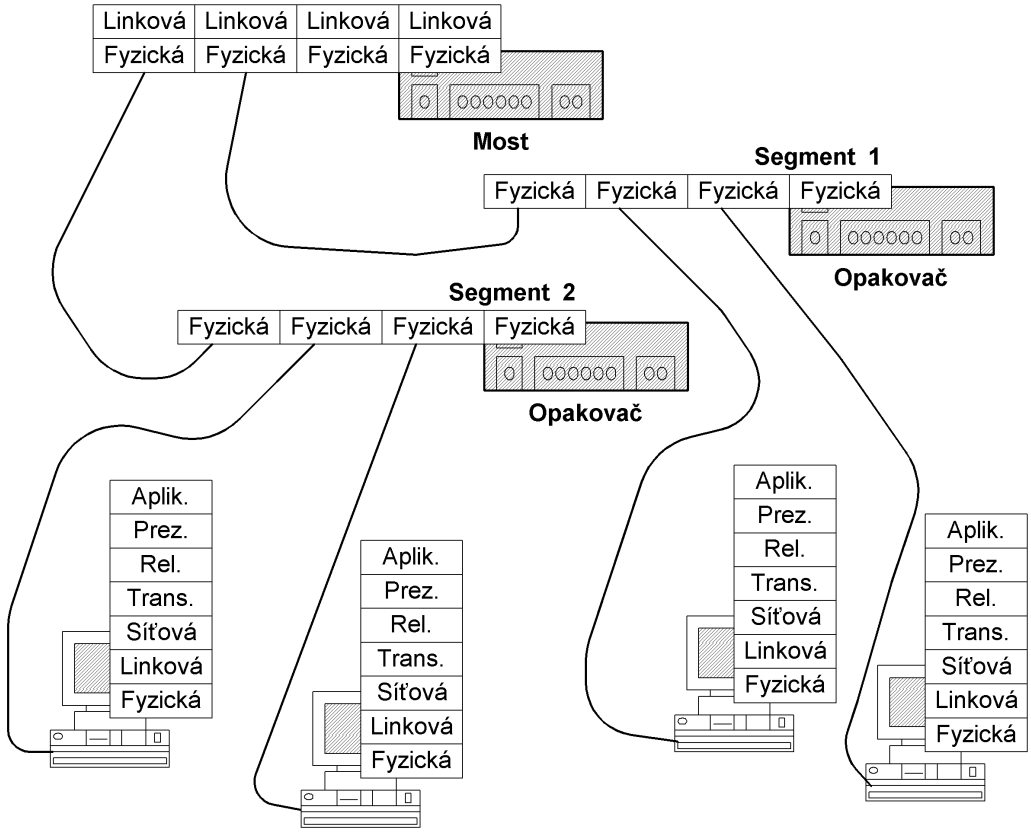
Ethernet na kroucené dvojlince se označuje jako 10BASE-T. Existuje i verze desetkrát rychlejšího Ethernetu označovaná 100BASE-TX a gigabitový Ethernet označovaný 1000BASE-CX. (Pomocí opakovačů nelze kombinovat 10BASE-T, 100BASE-TX a 1000BASE-CX – propojit je lze až pomocí přepínače). Délka dvojlinky mezi opakovačem a stanicí je standardně do 100 metrů.

Z hlediska síťového modelu pracuje opakovač (HUB) na fyzické úrovni. Komunikace mezi počítači je v LAN osazené opakovači transparentní (průhledná), tj. počítače na LAN spolu komunikují, aniž by o opakovači věděly.

Oproti opakovači **most** také spojuje mezi sebou jednotlivé segmenty LAN, ale neopakuje mechanicky všechny rámce, které se na nějakém z jeho portů objeví. Most je realizován specializovaným počítačem, který má předávací tabulku. V tabulce je seznam všech linkových adres všech síťových rozhraní LAN. U každé adresy má poznamenáno, za kterým síťovým rozhraním mostu se nachází. Objeví-li se datový rámec na nějakém síťovém rozhraní mostu, pak se most podívá do datového rámce na adresu příjemce a z předávací tabulky zjistí, za jakým rozhraním se adresát nachází. Rámec pak zopakuje pouze do rozhraní, za kterým je adresát. V případě, že se adresát nachází za stejným rozhraním, pak jej neopakuje vůbec. Oběžníky se pochopitelně opakují do všech rozhraní.

Důležitým parametrem mostu je, jak velkou může mít předávací tabulku, tj. kolik na ní má paměti. Avšak kardinální otázkou je, jak takovou tabulku naplnit správnými údaji. Naskýtá se odpověď, že data do ní může pořídit správce LAN ručně. Možná, že vám to připadá jako směšné řešení, ale toto řešení je oblíbené v případě sítí, kde se klade velký důraz na bezpečnost. Pak správce LAN takovou tabulku přesně nastaví. Dnes se mosty doplňují i o další tabulku, která je obdobou předávací tabulky a která vyjadřuje, kdo kam nemůže.

Jak se ale předávací tabulka naplní automaticky? Algoritmus je velice jednoduchý. Most pracuje po zapnutí v podstatě jako opakovač, tj. opakuje vše na všechna rozhraní. Avšak každému příchozímu rámci se podívá na adresu odesílatele. Most ví z jakého rozhraní rámec přišel, takže si může jako novou položku do předávací tabulky uložit adresu odesílatele a příslušné rozhraní.



Obr. 4.45 *Most*

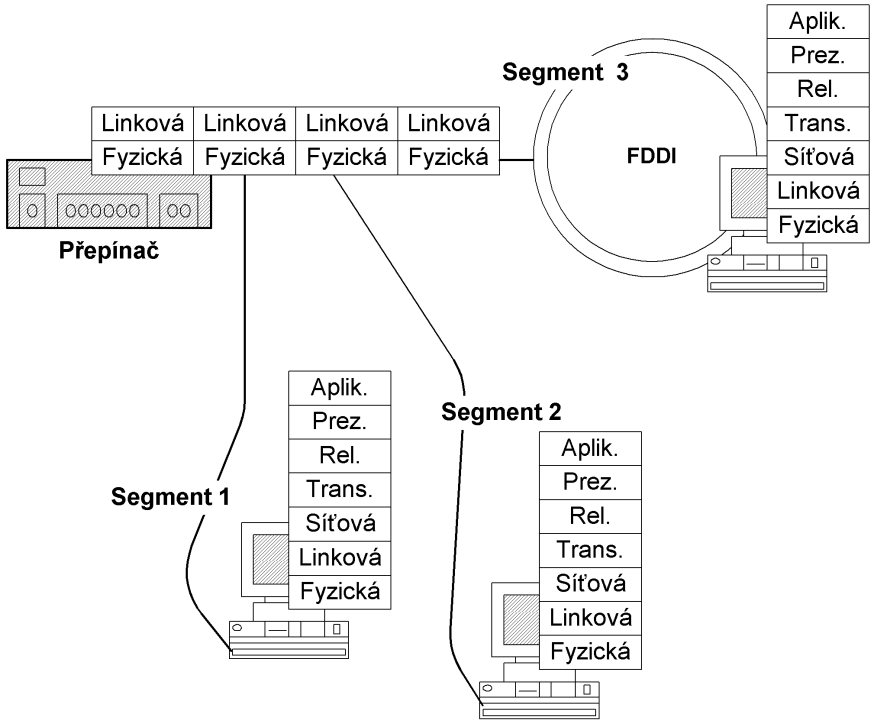
V lokální síti můžeme mít i více mostů. Předávání rámců mezi jednotlivými rozhraními mostu nemusí být tak rychlé jako u opakovače (může být delší doba odezvy). To otevírá cestu k tomu, aby dva mosty sítě byly propojeny např. sériovou linkou s modemy nebo radioreleovým spojem.

Jádrem jednotlivých segmentů LAN je opakovač. Jednotlivé segmenty jsou propojeny pomocí mostu. Na segment se pak umísťují počítače, které spolu více komunikují. Např. počítače jednoho oddělení. Na port mostu je užitečné připojit např. směrovač směřující do Internetu nebo na centrální server atp. Pomocí mostu lze tedy oddělit provoz mezi segmenty.

Jiným řešením je použít most s velkým počtem portů a nepoužít již opakovače pro jednotlivé segmenty sítě. Takovéto řešení se někdy nazývá přepínaný Ethernet. Jádrem přepínaného Ethernetu je inteligentní most, který v okamžiku, kdy zjistí, na které rozhraní má rámec opakovat, paralelně již začíná zpracovávat další rámec. Takovýto most se již označuje jako **přepínač**.

Přepínačem se označují výkonnější mosty, které umí opakovat rámce nejen mezi jednotlivými segmenty Ethernetu, ale i např. mezi Ethernetem a Fast Ethernetem, mezi Ethernetem a FDDI atd. Přepínač musí umět nejenom změnit tvar rámce např. z Ethernetu na FDDI, ale i pokusit se překlenout rozdíl mezi přenosovými rychlostmi. Problém je totiž při přenosu dat mezi rychlým segmentem (FDDI) a např. Ether-

Obr. 4.46
Přepínač



netem, kdy z FDDI může směrovat na Ethernet takové množství dat, že je Ethernet nedokáže odebírat. Rámce se musí ukládat do vyrovnávací paměti přepínače atd.

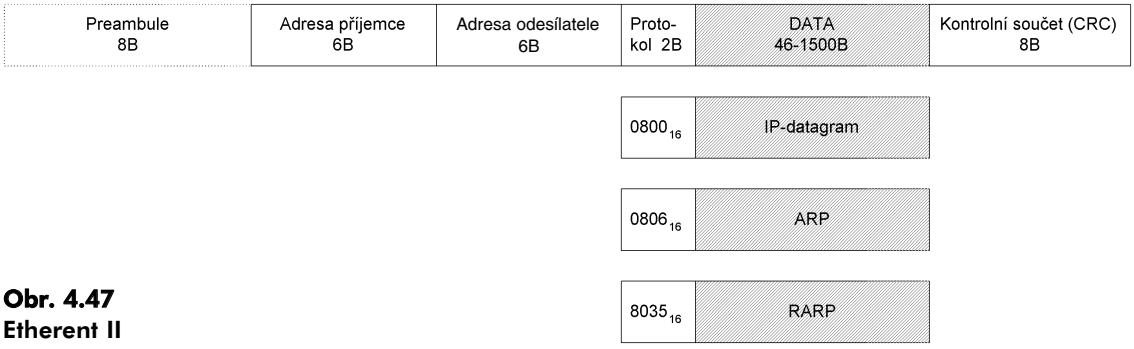
Pro výměnu rámců mezi stanicemi se používá protokol CSMA/CD. V tomto protokolu jsou si všechny stanice na LAN rovny. Potřebuje-li nějaká stanice vysílat, pak si poslechne zdali jiná stanice právě nevysílá. V případě, že médium není používáno (jiná stanice nevysílá), pak může stanice začít vysílat. Jenže v přibližně stejném okamžiku to mohlo napadnout dvě stanice najednou. Takže kromě toho, že stanice vysílá data, tak ještě připoslouchává, jestli nezačal vysílat současně někdo jiný. V případě, že současně začala vysílat jiná stanice, dochází ke kolizi. Při kolizi nemohou obě stanice okamžitě přestat vysílat (aby kolize byla i ostatními detekovatelná), tak ještě nějakou dobu vysílají bezvýznamné znaky a pak se na náhodně zvolený časový interval odmlčí.

Čím je na Ethernetu větší provoz, tím je větší prvaděpodobnost vzniku kolizí. Rozumnou zátěží je využití sítě asi na 20 %. Takže u varianty Ethernetu s frekvencí 10 MHz kalkulujeme propustnost sítě asi na 2 Mb/s (tj. 256 KB/s. Pro ilustraci u FDDI (100 MHz) je výtěžnost 80-90 %, takže lze kalkulovat 90 Mb/s, tj. asi 11 MB/s.

Pokud ale máme segment, kde jsou pouze dvě stanice, tak na koaxiálním kabelu může dojít na takovémto segmentu také ke kolizi. Jiná je situace v případě, že segment o dvou stanicích je na kroucené dvojlince, která má samostatný pár pro vysílání a samostatný pár pro příjem. Síťové karty se pak na takovýchto segmentech přepnou do plně duplexního provozu, ve kterém může stanice současně přijímat i vysílat data. Takovýto segment se nazývá bezkolizním segmentem. Na bezkolizním segmentu můžeme dosahovat praktických přenosových rychlostí blížících se až k teoretickému maximu. Pokud jádrem

LAN není opakovač, ale přepínač a jednotlivé stanice jsou připojeny bezkolizním segmentem, pak hovoříme o přepínaném Ethernetu. Bezkolizní segment je tvořen z jedné strany počítačem a z druhé strany rozhraním přepínače.

Struktrura rámce protokolu Ethernet závisí na použité normě. Struktura rámce protokolu Ethernet II je znázorněna na obr. 4.47.



Obr. 4.47
Etherent II

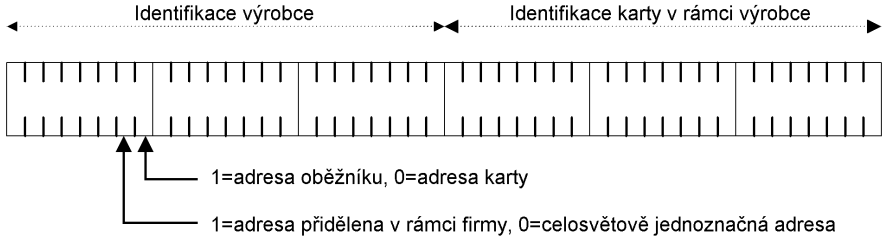
Ethernet II má na počátku synchronizační preambuli (součást fyzické vrstvy), při které se synchronizují všechny stanice přijímající rámec. Na konci rámce je kontrolní součet, ze kterého lze zjistit, nebyl-li rámec přenosem poškozen. Dále obsahuje šestibajtovou linkovou adresu příjemce a odesílatele, pole specifikující protokol vyšší vrstvy (tj. síťové vrstvy) a vlastní přenášená data (specifikace protokolů: IP verze 4, ARP a RARP je patrná z obrázku 4.47). Datové pole musí být minimálně 46 bajtů dlouhé, takže v případě, že je potřeba přenášet méně dat, tak se datové pole zprava doplní bezvýznamnou výplní.

Fyzická adresa je šestibajtová. První tři bajty specifikují výrobce síťové karty a zbylé tři bajty kartu v rámci výrobce, takže adresy jsou celosvětově unikátní. Toto platí pouze pro tzv. globální adresy, které jsou celosvětově jednoznačné. Tyto adresy jsou uloženy v permanentní paměti síťové karty. Při inicializaci karty ovladačem lze kartě sdělit, aby nepoužívala tuto adresu, ale adresu jinou. V rámci firmy tak lze používat vlastní systém linkových adres. Tento mechanismus využíval např. protokol DECnet fáze IV.

Síťová karta může používat globálně jednoznačnou adresu nebo jednoznačnou adresu v rámci firmy. Kromě těchto jednoznačných adres existují ještě oběžníky. Všeobecný oběžník (adresa se skládá z 48 jedniček) je určen pro všechny stanice na LAN. Adresný oběžník (má nastaven nejnižší bit prvního bajtu na jedničku) je určen pouze některým stanicím na LAN, stanicím, které akceptují uvedenou adresu.

Nultý a první bit prvního bajtu linkové adresy mají specifický význam (viz obr. 4.48):

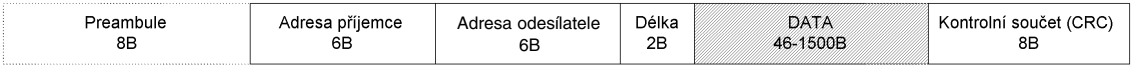
Obr. 4.48
Linková adresa příjemce



- Nultý bit specifikuje, zdali se jedná o jednoznačnou adresu nebo adresu oběžníku.
- První bit specifikuje, zdali se jedná o globálně jednoznačnou adresu.

Uveďme si příklad výpisu rámce protokolu Ethernet II z MS Network Monitoru:

```
+ FRAME: Base frame properties
  ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
      ETHERNET: Destination address : 00000C31D211
          ETHERNET: .......0 = Individual address
          ETHERNET: ......0. = Universally administered address
      ETHERNET: Source address : 0010A4F18B3E
          ETHERNET: .......0 = No routing information present
          ETHERNET: ......0. = Universally administered address
      ETHERNET: Frame Length : 74 (0x004A)
      ETHERNET: Ethernet Type : 0x0800 (IP:  DOD Internet Protocol)
      ETHERNET: Ethernet Data: Number of data bytes remaining = 60 (0x003C)
+ IP: ID = 0xAB06; Proto = ICMP; Len: 60
+ ICMP: Echo,     From 195.47.37.200 To   194.149.105.18

00000:  00 00 0C 31 D2 11 00 10 A4 F1 8B 3E 08 00 45 00   ...1.......>..E.
00010:  00 3C AB 06 00 00 20 01 DB 1B C3 2F 25 C8 C2 95   .<.... ..../%...
00020:  69 12 08 00 42 5C 01 00 0A 00 61 62 63 64 65 66   i...B\ ....abcdef
00030:  67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76   ghijklmnopqrstuv
00040:  77 61 62 63 64 65 66 67 68 69                     wabcdefghi
```



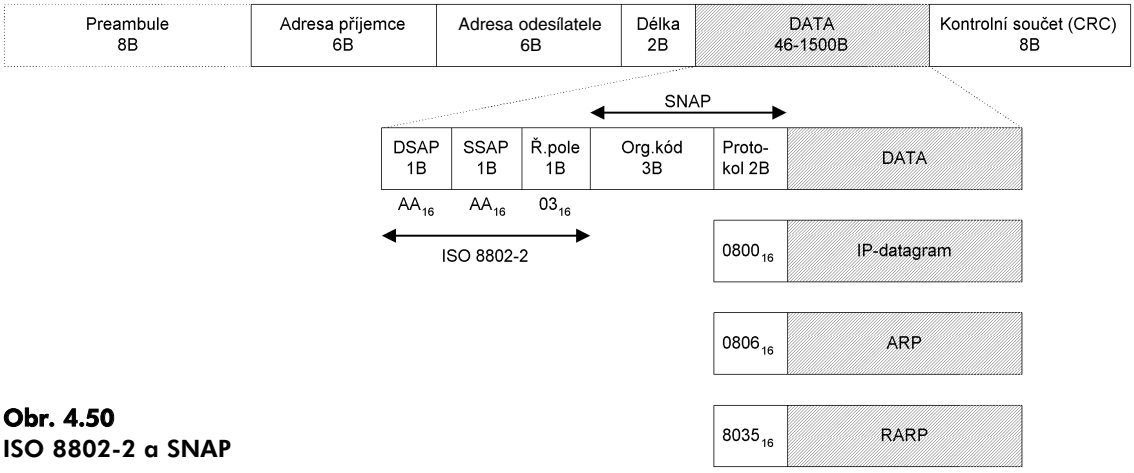
Obr. 4.49

Situace u protokolu ISO 8802-3 je poněkud složitější. Datový rámec protokolu ISO 8802-3 se liší pouze v jednom poli proti protokolu Ethernet II viz obr. 4.49.

Avšak datové pole (viz obr. 4.50) může v sobě nést nikoliv přímo data, ale paket protokolu ISO 8802-2, jehož záhlaví může být rozšířeno ještě o další dvě pole tvořící tzv. SNAP. Jinými slovy stanice mohou spolu komunikovat:

- Surovými rámci protokolu ISO 8802-3 (bez ISO 8802-2 a bez SNAP).
- Rámci protokolu ISO 8802-3, ve kterých je zabalen protokol ISO 8802-2 bez SNAP. Hovorově Ethernet ISO 8802-2.
- Rámci protokolu ISO 8802-3, ve kterých je zabalen protokol ISO 8802-2 se SNAP. Hovorově Ethernet SNAP.

Pole délka vyjadřuje délku přenášených dat. Je to pole, kterým se právě obě normy liší. V provozu sítě však nemůže dojít k záměně typů rámců jednotlivých protokolů, protože délka dat je nejvýše 1500 B a specifikace protokolů pro normu Ethernet II jsou vyjadřovány vyššími čísly než 1500 B.



Obr. 4.50
ISO 8802-2 a SNAP

Nyní uvádíme příklad výpisu rámce Ethernet SNAP.

```
+ FRAME: Base frame properties
  ETHERNET: 802.3 Length = 60
      ETHERNET: Destination address : 010081000100
          ETHERNET: .......1 = Group address
          ETHERNET: ......0. = Universally administered address
      ETHERNET: Source address : 0000810C3D50
          ETHERNET: .......0 = No routing information present
          ETHERNET: ......0. = Universally administered address
      ETHERNET: Frame Length : 60 (0x003C)
      ETHERNET: Data Length : 0x0013 (19)
      ETHERNET: Ethernet Data: Number of data bytes remaining = 46 (0x002E)
  LLC: UI DSAP=0xAA SSAP=0xAA C
      LLC: DSAP = 0xAA : INDIVIDUAL : Sub-Network Access Protocol (SNAP)
      LLC: SSAP = 0xAA: COMMAND : Sub-Network Access Protocol (SNAP)
      LLC: Frame Category: Unnumbered Frame
      LLC: Command = UI
      LLC: LLC Data: Number of data bytes remaining = 43 (0x002B)
  SNAP: ETYPE = 0x01A2
      SNAP: Snap Organization code = 00 00 81
      SNAP: Snap etype : 0x01A2
      SNAP: Snap Data: Number of data bytes remaining = 38 (0x0026)

00000:  01 00 81 00 01 00 00 00 81 0C 3D 50 00 13 AA AA   ..........=P....
00010:  03 00 00 81 01 A2 7F 00 00 02 00 01 01 2C 01 02   ...... .......,..
00020:  00 00 80 00 00 00 81 0C 3D 50 80 04 00 00 14 00   ........=P......
00030:  02 00 0F 00 00 00 00 00 00 00 00 00               ............
```

Zvolený rámec nenese IP-datagram, jak jste asi očekávali. V Internetu je předepsáno, že každá stanice musí podporovat protokol Ethernet II. Pouze stanice, které se nějak dohodnou na použití protokolu Ethernet ISO 8802-3, jej mohou používat. Proto se v naprosté většině případů v Internetu setkáváme s protokolem Ethernet II.

Vraťme se k popisu polí. Destination Service Access Point (DSAP)/Source Service Access point (SSAP) specifikují aplikaci cílovou/zdrojovou aplikaci, která rámec odesílá/přijímá. Např. pro IP-protokol se používá DSAP=SSAP=AA16 a pro NetBIOS se používá DSAP=SSAP=F016. Při použití protokolu ISO 8802-2 je možné doručovat data až jednotlivým aplikacím běžícím na stanici. Existují i síťové protokoly, které pro komunikaci na LAN používají pouze tuto adresaci (nepoužívají síťovou vrstvu). Použití takových protokolů je sice efektní (o jednu vrstvu jsou rychlejší), ale jsou nesměrovatelné, tj. jsou určeny pouze pro LAN, nikoliv pro WAN. Příkladem takovéhoto exotického protokolu je protokol NetBEUI.

Řídící pole je naprosto analogické řídícímu poli protokolu HDLC. Opět mezi stanicemi se může komunikovat pomocí U, I a S-rámců. Rámce mohou být číslovány, v případě ztráty nebo chyby v rámci může být vyžádána retransmise atd. Pro potřeby protokolu IP se používají pouze U-rámce a P/F bit je nastaven na nulu, tj. řídící pole má hodnotu 0316 (obdobně jako v případě protokolu PPP).

Pomocí záhlaví SNAP (Sub-network Access Protocol) je možné specifikovat protokol vyšší vrstvy, jedná se tedy o obdobu pole protokol v Ethernetu II. Dokonce pro specifikaci protokolu vyšší vrstvy se používají stejné hodnoty. Jinými slovy co chybělo protokolu ISO 8802-3 oproti protokolu Ethernet II (pole protokol), se krkolomně řeší pomocí záhlaví SNAP.

4.7.2 FDDI

FDDI je normalizováno normou ISO 9314.

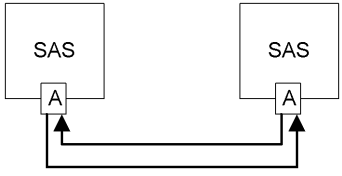
FDDI je lokální síť tvořící kruh. Jednotlivé stanice jsou propojeny do kruhu. K propojení stanic se používá optické vlákno. Lidovější variantou FDDI je varianta používající místo optického vlákna kroucenou dvojlinku (CDDI). Obě varianty se liší pouze použitým přenosovým médiem a modulem propojujícím přenosové médium se síťovou kartou. Tyto moduly jsou často výměnné, takže je možné snadno přejít od optického vlákna ke dvojlince a naopak.

Propojovací kruh je zpravidla zdvojen. Obsahuje primární a sekundární okruh. To umožňuje snadné zálohování situace, kdy dojde k přerušení kruhu. Přídavné síťové karty pro FDDI se vyrábějí ve dvou variantách:

- SAS (Single-Attachment Station) – s jednou dvojicí konektorů, tj. pro použití pouze na jednoduchém kruhu.
- DAS (Dual-Attachment Station) – s dvěma dvojicemi konektorů, tj. pro použití na zdvojeném kruhu (backbone).

Nejjednodušší zapojení dvou stanic SAS, které lze považovat spíše za nouzové řešení, je znázorněno na obr. 4.51.

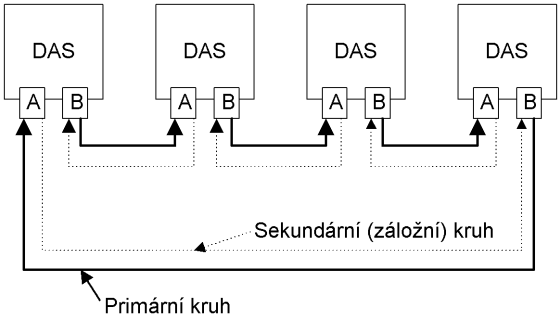
Obr. 4.51 Nejjednodušší propojení FDDI



Klasické zapojení FDDI spočívá v použití síťových karet DAS. Páteř LAN na bázi FDDI spočívá právě ve vytvoření páteřního kruhu ze stanic DAS. Právě karty DAS umožňují zdvojený páteřní kruh (primár-

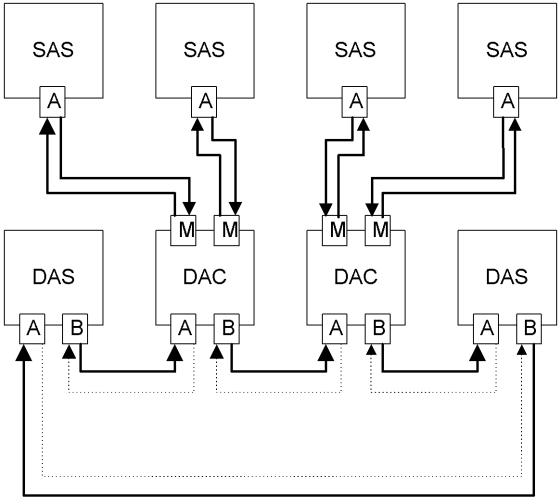
ní a sekundární). Síťové karty DAS se používají u koncentrátorů, přepínačů a případně centrálních počítačů.

Obr. 4.52
Primární a sekundární kruh FDDI



Koncentrátor (*Dual-Attachment Concentrator* – DAC) je aktivní prvek LAN, který kromě karty DAS obsahuje 2, 4, 8, 16, 32 atd. SAS rozhraní (viz obr. 4.53). Tato SAS rozhraní označená jako M (*Master)* slouží pro připojení jednotlivých počítačů s kartou SAS typu *Slave* k FDDI kruhu. V případě zapnutí stanice koncetrátor zařadí stanici do logického kruhu FDDI a stanice může komunikovat s ostatními stanicemi na LAN. Použití SAS rozhraní pro jednotlivé počítače přináší úsporu nejen v ceně jednodušší síťové karty, ale i jednoduššího rozvodu.

Obr. 4.53
DAC



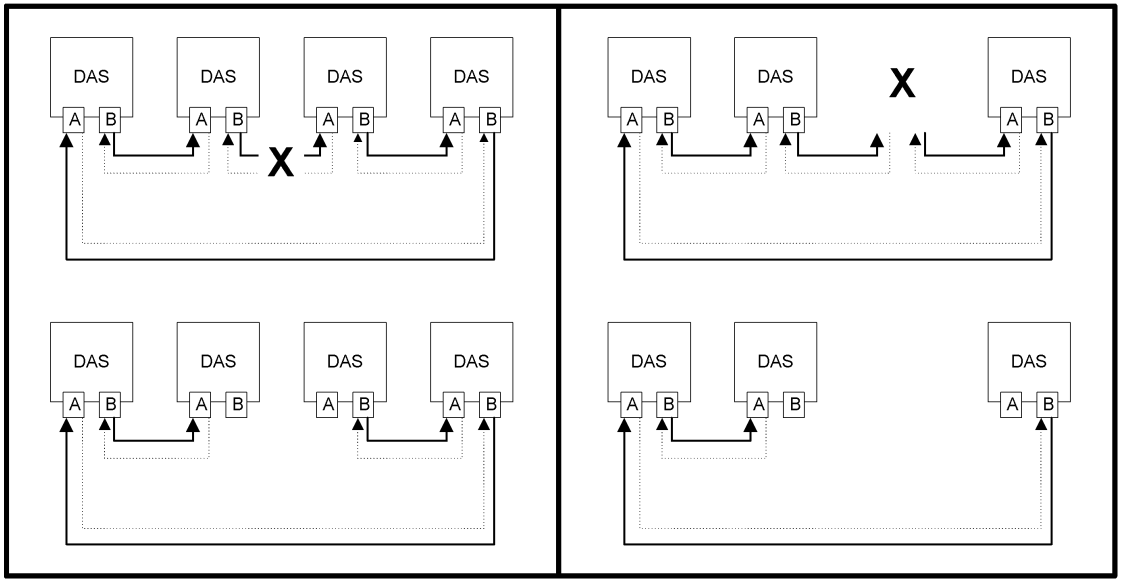
Vypnutí nebo poruchu stanice SAS připojené ke koncetrátoru zjistí koncetrátor a stanici vyjme z logického kruhu FDDI bez toho, aby byla ohrožena komunikace na kruhu.

Funkce sekundárního kruhu FDDI spočívá v jeho využití při poruše primárního kruhu. Přerušení páteřního kruhu tvořeného primárním a sekundárním okruhem v jednom místě nebo porucha jedné stanice neohrozí celou síť. Obě situace a jejich řešení jsou znázorněny na následujícím obrázku 4.54.

Důležité je, že k přerušení kruhu může dojít pouze na jednom místě. Přerušení na více místech je pro FDDI osudové. Z tohoto důvodu se páteřní kruh zpravidla nerozvádí po budově, ale jen na nezbytně nutné vzdálenosti. K přerušení kruhu dojde i prostým vypnutím stanice, proto se na páteřní kruh neumísťují stanice, které neběží nepřetržitě.

FDDI se používá jako páteř lokální sítě. Pomocí přepínačů se propojuje s ostatními segmenty LAN, pro které je použit např. Ethernet, viz obr. 4.55.

Je třeba si uvědomit funkci přepínače. Posílá-li totiž stanice na FDDI datové rámce stanici na Ethernetu, pak rámce mohou přicházet rychleji, než je může přepínač předávat na Ethernet. Musí mít vyrovnávací paměť, do které si může určité množství dat uložit. Dalším problémem přepínače je, že rámce na FDDI a na Ethernetu mají odlišnou strukturu, proto musí umět provést konverzi rámců mezi FDDI a Ethernetem.



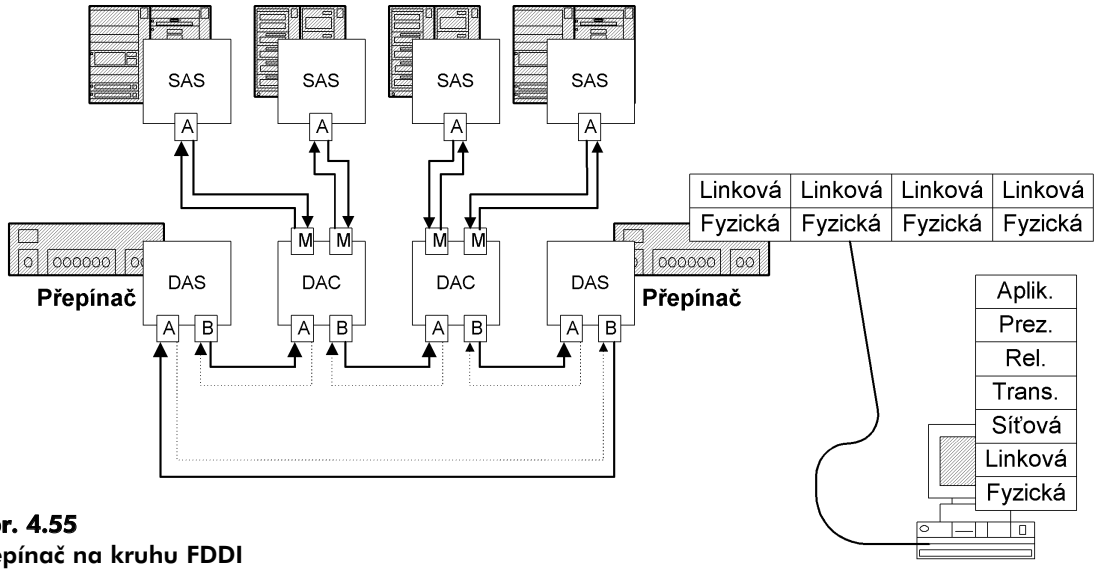
Obr. 4.54 Porucha na páteřním kruhu FDDI

Existují i přepínače, které přepínají mezi dvěma či více kruhy FDDI, mezi FDDI a Fast Ethernetem apod.

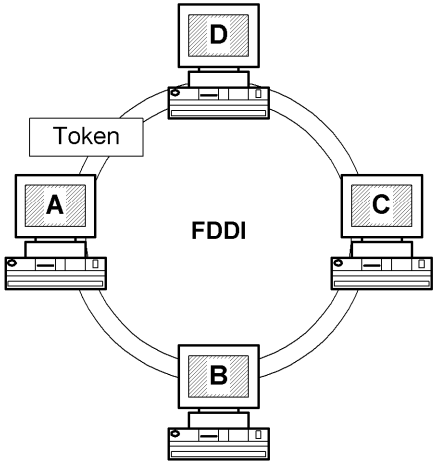
FDDI používá protokol token-passing pro výměnu rámců mezi stanicemi na logickém kruhu FDDI. Základem je, že stanice smí vysílat jen tehdy, když obdrží speciální rámec nazývaný token. Princip si objasníme na příkladu, kdy FDDI je tvořeno čtyřmi stanicemi. Stanice A chce poslat rámec 1 stanici C a stanice B chce poslat rámec 2 stanici D (viz obr. 4.56 a 4.61).

Je zřejmé, že řízení toku dat je u FDDI poměrně komplikované. Neřešili jsme otázky, kdo vloží první token do kruhu, jak se má zachovat stanice, když v určitém intervalu vůbec neobdrží token atd.

Vedle fyzické vrstvy, vrstvy MAC a vrstvy LLC je zde proto poměrně významná instance správy stanice (*Station Management*), viz obr. 4.62.

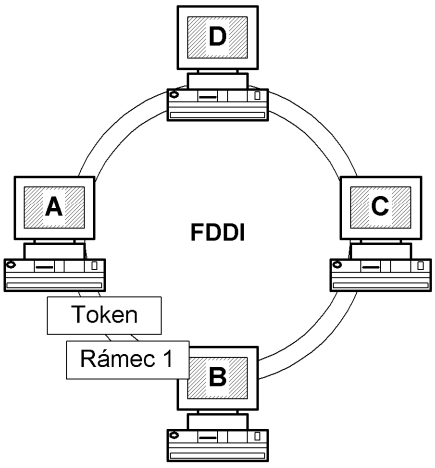


Obr. 4.55
Přepínač na kruhu FDDI

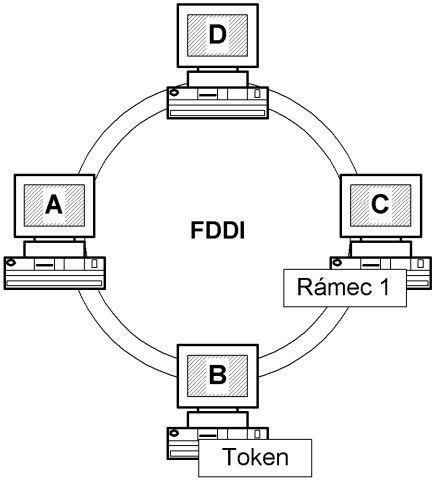


Obr. 4.56
První fází je, že stanice A chce odeslat rámec 1, takže si musí počkat, až dostane token.

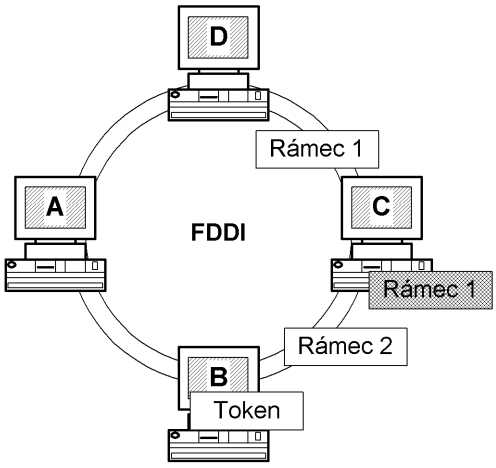
Datová část rámce FDDI může obsahovat paket vrstvy LLC (podobně, jak jsem se o tom již dříve zmínil u Ethernetu), čehož se využívá i v případě IP-protokolu přes FDDI. IP-datagram je předcházen záhlavím SNAP, které je předcházeno záhlavím ISO 8802.2 a celé je to vloženo do rámce FDDI. Struktura rámce FDDI a tokenu je znázorněna na následujícím obrázku 4.63.



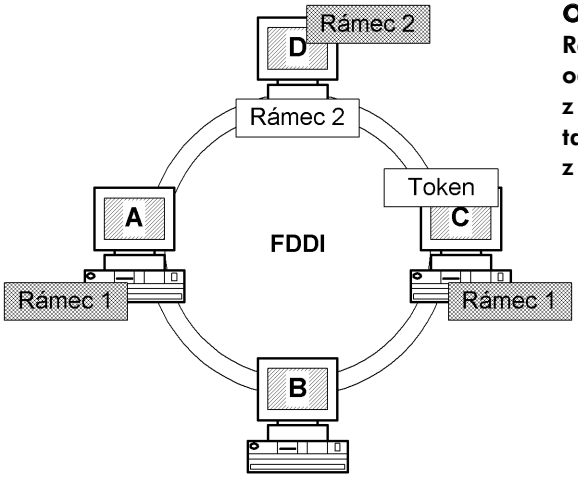
Obr. 4.57
Stanice A obdrží token, který pozdrží a vyšle rámec 1 adresovaný stanici C. Poté stanice A vyšle dále token, aby další stanice také mohly vysílat.



Obr. 4.58
Stanice B propustí rámec 1, protože jí není určen. Avšak stanice B obdržela také token, takže jej pozdrží, aby sama mohla vysílat. Ke stanici C dorazil rámec 1, který je pro ni určen, takže stanice C si jej zkopíruje z logického kruhu FDDI, aby jej mohly využít její aplikace.



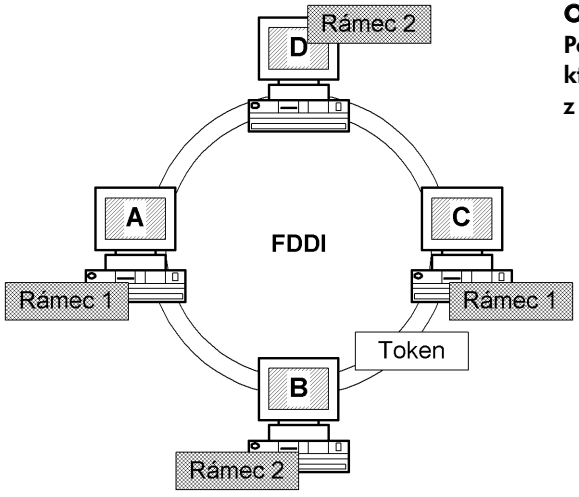
Obr. 4.59
Stanice C po zkopírování propouští rámec 1. Stanice B vysílá rámec 2.



Obr. 4.60
Rámec 1 dorazil ke stanici A, která je odesílatelem, takže stanice A jej odstraní z logického kruhu. Ke stanici D dorazil rámec 2, takže stanice D si jej může zkopírovat z logického kruhu pro využití jejími aplikacemi.

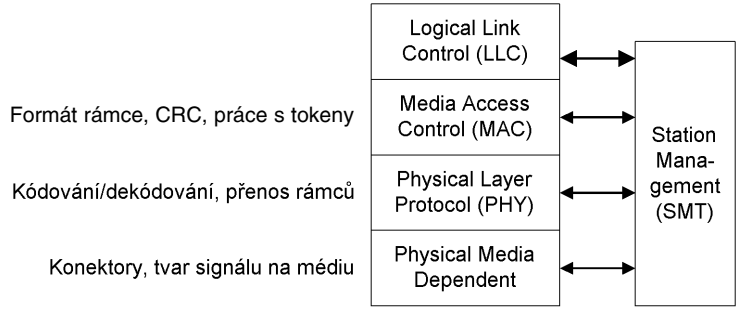
Formát zdrojové a cílové adresy se u protokolu FDDI používá stejný jako pro Ethernet. Zajímavé je řídící pole, které obsahuje:

- Synchronní/asynchronní přenos, toto políčko je určeno pro signalizaci, zdali se jedná o synchronní přenos, tj. přenos kdy se garantuje šíře pásma (např. pro audio).
- 16/48 bitová adresa specifikuje délku linkové adresy. Zásadně se používá adresa dlouhá 48 bitů.
- Typ rámce souvisí s poslední částí, která je určena pro potřeby vrstvy MAC.



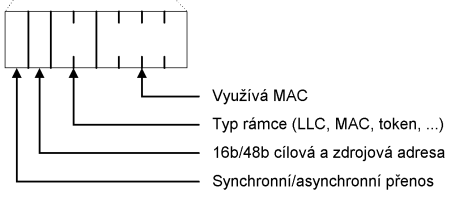
Obr. 4.61
Poslední fází je, že rámec 2 dorazil ke stanici B, která jej odeslala. Stanice B odstraňuje rámec 2 z kruhu.

Obr. 4.62



Rámec FDDI

Preambule	Křídlová značka	Řídící pole	Cílová adresa	Zdrojová adresa	DATA	Kontrolní součet	Křídlová značka	Status
8B	1B	1B	6B	6B	0-4478B	4B	0,5B	1,5B



Obr. 4.63 Rámec FDDI

Token FDDI

Preambule	Křídlová značka	Řídící pole	Křídlová značka
8B	1B	1B	1B

5

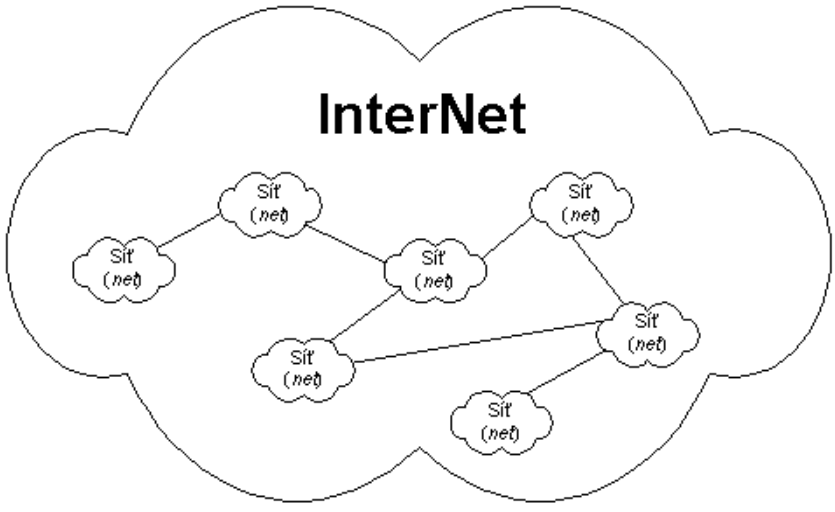
IP protokol *(Internet Protocol)*

Některé linkové protokoly jsou určeny pro dopravu dat v rámci lokální sítě, jiné linkové protokoly dopravují data mezi sousedními směrovači rozsáhlé sítě. IP-protokol na rozdíl od linkových protokolů dopravuje data mezi dvěma libovolnými počítači v Internetu, tj. i přes mnohé LAN.

Data jsou od odesílatele k příjemci dopravována (směrována) přes směrovače (*router*). Na cestě od odesílatele k příjemci se může vyskytnout cela řada směrovačů. Každý směrovač řeší samostatně směrování k následujícímu směrovači. Data jsou tak předávána od směrovače k směrovači. Z angličtiny se počeštil v tomto kontextu termín následující hop (*next hop*), jako následující uzel kam se data předávají. Hopem se rozumí buď následující směrovač nebo cílový stroj.

IP-protokol je protokol, umožňující spojit jednotlivé lokální sítě do celosvětového Internetu. Od protokolu IP dostal také Internet své jméno. Zkratka IP totiž znamená InterNet Protocol, tj. protokol spojující jednotlivé sítě. Později se místo InterNet začalo psát Internet a Internet byl na světě.

Obr. 5.1
InterNet

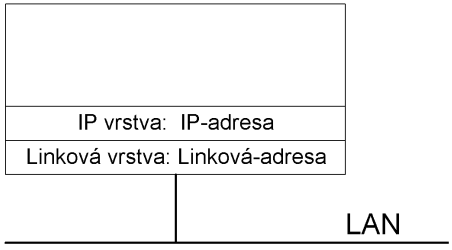


IP-protokol je tvořen několika dílčími protokoly:

- Vlastním protokolem IP.
- Služebním protokolem ICMP sloužícím zejména k signalizaci mimořádných stavů.
- Služebním protokolem IGMP sloužícím pro dopravu adresných oběžníků.
- Služebními protokoly ARP a RARP, které jsou často vyčleňovány jako samostatné, na IP nezávislé protokoly, protože jejich rámce nejsou předcházeny IP-záhlavím.

Zatímco v linkovém protokolu mělo každé síťové rozhraní (*network interface*) svou fyzickou (tj. linkovou) adresu, která je v případě LAN zpravidla šestibajtová, tak v IP-protokolu má každé síťové rozhraní alespoň jednu IP-adresu, která je v případě IP-protokolu verze 4 čtyřbajtová, a v případě IP-protokolu verze 6 šestnáctibajtová.

Obr. 5.2
Linková adresa a IP-adresa



Základním stavebním prvkem WAN je směrovač (anglicky *router*), kterým se vzájemně propojují jednotlivé LAN do rozsáhlé sítě. Jako směrovač může sloužit běžný počítač s více síťovými rozhraními a běžným operačním systémem nebo specializovaná skříňka (*box*), do které nebývá běžně zapojen ani monitor ani klávesnice. Tyto specializované skříňky se u nás v Česku mezi odbornou veřejností nazývají routery a v tiskovinách směrovače. Slovo směrovač má tedy dva významy. V prvním obecném významu se směrovačem míní funkce počítače (ať klasického počítače nebo specializované skříňky) předávat datové pakety mezi dvěma síťovými rozhraními a v druhém a to praktickém smyslu se jím označuje specializovaná skříňka pracující jako směrovač.

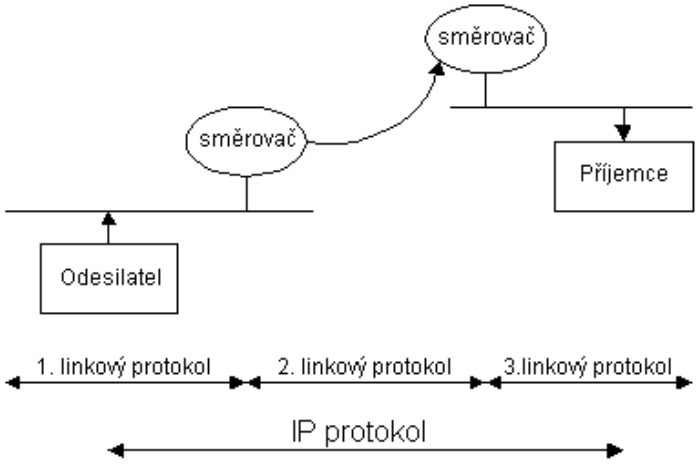
Schopnost předávat datové pakety mezi síťovými rozhraními směrovače se nazývá jako předávání (*forwarding*). Zatímco u směrovačů je tato funkce požadována, tak u počítačů s klasickým operačním systémem (UNIX, OpenVMS, NT apod.) je někdy dotazováno, jak přinutit jádro operačního systému předávání zakázat.

Základní otázkou je: „Proč jsou třeba dva protokoly: linkový protokol a protokol IP? Proč nestačí pouze linkový protokol?“. Linkový protokol slouží pouze k dopravě dat v rámci LAN. Tj. k dopravě dat k nejbližšímu směrovači, ten z linkového rámce data „vybalí“ a „přebalí“ je do jiného linkového rámce. Na každém rozhraní směrovače může být použit jiný linkový protokol. A nenechte se mýlit případem, kdy směrovač na svých rozhraních používá stejný linkový protokol – např. Ethernet. I v tomto případě dochází k „přebalování“ – stačí si uvědomit, že ethernetový rámec používá před přebalením jiné fyzické adresy než po přebalení.

Avšak pádným argumentem na otázku „proč dva protokoly“ jsou až vlastnosti protokolů, které používají k dopravě dat pouze linkovou vrstvu, tj. jednotliví účastníci komunikace mají pouze linkové (šestibajtové) adresy. Takovými protokoly jsou např. NetBEUI (Microsoft) či LAT (Digital). Tyto protokoly

jsou jednoduché a opravdu asi rychlejší při tvorbě a zpracování svých paketů. Avšak díky tomu, že lze příjemce adresovat pouze v rámci LAN, tak nelze odeslat data příjemci za směrovačem – tj. ve WAN. Proto se tyto protokoly označují jako nesměrovatelné. Jsou použitelné pouze v rámci lokální sítě, nikoliv mimo ni.

Obr. 5.3
Linkové protokoly a IP protokol



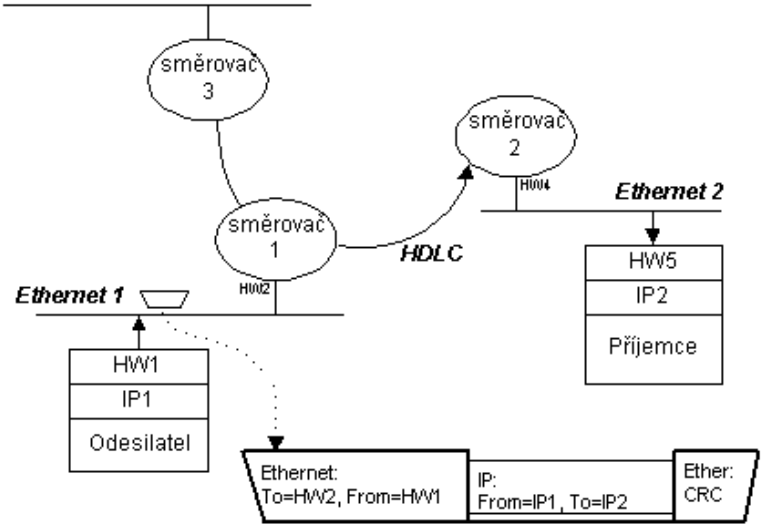
Obrázek 5.3 znázorňuje, že linkový protokol dopravuje datové rámce pouze k následujícímu směrovači, kdežto IP-protokol dopravuje data mezi dvěma vzdálenými počítači rozsáhlé sítě (WAN). Zatímco obálka, kterou jsou na linkové vrstvě data obalena je na každém směrovači vždy zahozena a vytvořena nová, tak IP-datagram není směrovačem změněn. Směrovač nesmí změnit obsah IP-datagramu. Výjimkou je pouze položka TTL ze záhlaví IP-datagramu, kterou je každý směrovač povinen zmenšit alespoň o jedničku a v případě změny na nulu se IP-datagram zahazuje. Tímto mechanismem se Internet snaží zabránit nekonečnému toulání paketů Internetem. Existují i další výjimky, ke kterým se také později dostaneme (např. fragmentace).

Zatímco u linkových protokolů jsme základní přenášené kvantum dat označovali jako linkový rámec, tak u IP-protokolu je základní jednotkou přenášených dat IP-datagram.

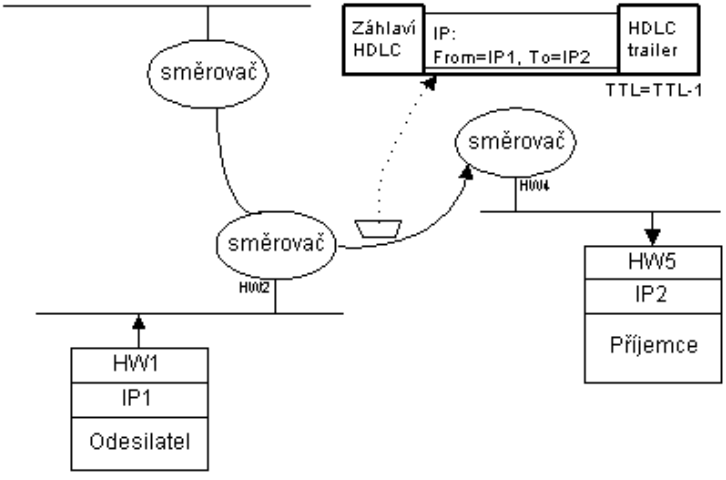
Proberme si případ z obr. 5.4, kdy odesílatel z lokální sítě **Ethernet 1** odesílá IP-datagram příjemci na síti **Ethernet 2**. IP-adresu odesílatele a příjemce jsme na obrázku 5.4 pro jednoduchost označili slovy **From** a **To** jak je zvykem u elektronické pošty. Obdobně jsme označili i linkové adresy. Např. odesílatel má na obrázku linkovou adresu HW1.

Odesílatel chce odeslat IP-datagram příjemci o IP-adrese IP2. Vytvoří IP-datagram, ale aby jej mohl vložit do lokální sítě, tak jej musí vložit do linkového rámce (v našem případě Ethernet). Docela výstižné je přirovnání, že „IP-datagram byl naložen na loď Ethernet 1". Linkovým protokolem však mohou tato data putovat jen na **směrovač 1**, který IP-datagram vybalí z ethernetového rámce a podívá se na IP-adresu příjemce. Podle IP-adresy příjmce se rozhodne kterým svým rozhraním pošle IP-datagram dále – tj. „na jaký linkový protokol se provede překládka IP-datagramu".

Rozhodování to však není jednoduché, směrovač se rozhoduje na základě svých směrovacích tabulek (*routing table*), kterým se budeme také podrobně věnovat. Předpokládejme, že směrovač se rozhodl pro linku **HDLC**.



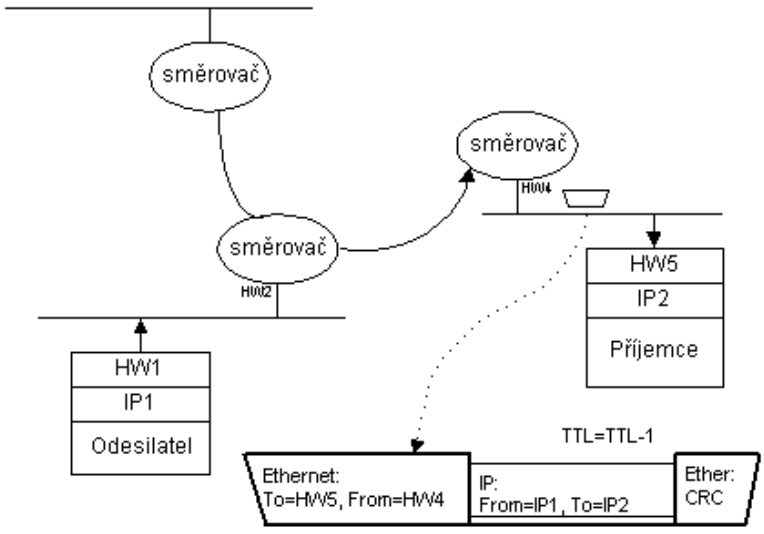
Obr. 5.4
Odesílatel odesílá IP-datagram v rámci protokolu Ethernet



Obr. 5.5
IP-datagram byl vložen do rámce protokolu HDLC

Směrovač sníží hodnotu položky TTL alespoň o jedničku a vloží náš IP-datagram do jiného linkového protokolu, kterým je v tomto případě protokol HDLC – viz obr. 5.5. Přirovnáme-li protokol HDLC ke kontejnerové dopravě, pak „náš IP-datagram byl přeložen z lodi Ethernet 1 do kontejneru společnosti HDLC“.

Protokolem HDLC je náš IP-datagram dopraven na následující směrovač, který opět IP-datagram vybalí z HDLC-obálky, sníží hodnotu položky TTL a po obalení ethernetovou obálkou jej vloží do cílové LAN.



Obr. 5.6
IP-datagram je opět vložen do rámce protokolu Ethernet

Záměrně jsem si na obou LAN vybral stejný linkový protokol (Ethernet). Aby bylo vidět, že pokaždé se jedná o zcela jiný linkový rámec. Na LAN odesílatele má ethernetový rámec adresu příjemce HW2 a odesílatel HW1, kdežto na LAN příjemce se sice také jedná o Ethernet, ale linková adresa příjemce je HW5 a odesílatele HW4.

5.1 IP-datagram

Při výkladu protokolů TCP/IP je zvykem vše znázorňovat v tabulce jejíž řádek má 4 bajty, tj. bity 0 až 31. I my budeme často používat toto znázornění.

IP-datagram se skládá ze záhlaví a přenášených dat. Záhlaví má zpravidla 20 bajtů. Záhlaví však může obsahovat i volitelné položky a v takovém případě je záhlaví o ně delší.

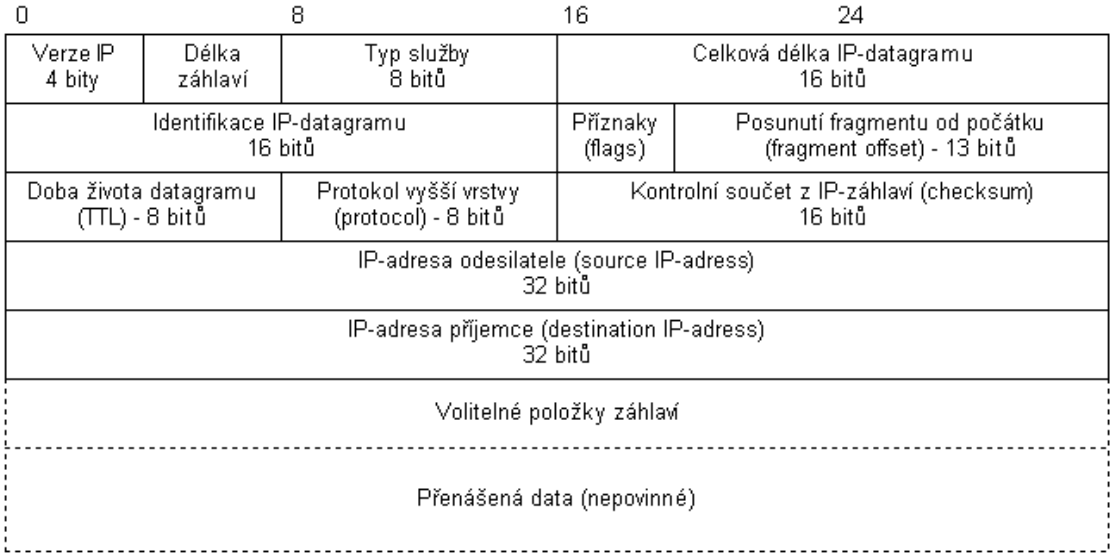
Struktura IP-datagramu je na obrázku 5.7.

Ještě než začneme popisovat jednotlivé položky záhlaví, tak si nějaký IP-datagram odchytneme pomocí MS Network Monitoru (viz obr. 5.8).

A tak bude okamžitě vidět, zdali sítí chodí opravdu to, co popisujeme. Nyní již můžeme začít s popisováním významu jednotlivých položek záhlaví IP-datagramu.

Verze (*version*) je první položkou záhlaví IP-datagramu. Tato položka dlouhá 4 bity (půl bajtu) obsahuje verzi IP-protokolu. V této kapitole hovoříme o IP-protokolu verze 4, tudíž tato položka je v našem případě rovná hodnotě 4.

Délka záhlaví (*header length*) obsahuje délku záhlaví IP-datagramu. V případě odchyceného IP-datagramu na obr. 5.8 je délka záhlaví 20, ale jak je vidět z hexadecimálního výpisu z MS Network Monitoru, tak položka délka záhlaví nabývá hodnoty 5 (nikoliv 20). Vysvětlení je prosté. Délka není uváděna v bajtech, ale v čtyřbajtech a 5x4=20. Délka záhlaví musí tak být i v případě použití volitelných položek násobkem čtyř. V případě, že by záhlaví nevyšlo na násobek čtyř, pak se na násobek čtyř doplní nevýznamnou výplní.



Obr. 5.7 IP-datagram

```
+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
  IP: ID = 0x5814; Proto = ICMP; Len: 60
      IP: Version = 4 (0x4)
      IP: Header Length = 20 (0x14)
      IP: Service Type = 0 (0x0)
          IP: Precedence = Routine
          IP: ...0.... = Normal Delay
5.1.1             IP: ....0... = Normal Throughput
          IP: .....0.. = Normal Reliability
      IP: Total Length = 60 (0x3C)
      IP: Identification = 22548 (0x5814)
      IP: Flags Summary = 0 (0x0)
5.1.2             IP: .......0 = Last fragment in datagram
5.1.2.1                  IP: ......0. = May fragment datagram if necessary
      IP: Fragment Offset = 0 (0x0) bytes
      IP: Time to Live = 32 (0x20)
5.1.3         IP: Protocol = ICMP - Internet Control Message
      IP: Checksum = 0xEBF0
      IP: Source Address = 194.149.104.198
      IP: Destination Address = 194.149.104.203
      IP: Data: Number of data bytes remaining = 40 (0x0028)
+ ICMP: Echo,     From 194.149.104.198 To   194.149.104.203

00000:  00 00 F8 21 71 A4 00 20 AF FA 25 89 08 00 45 00   ...!q.. ..%...E.
00010:  00 3C 58 14 00 00 20 01 EB F0 C2 95 68 C6 C2 95   .<X... .....h...
00020:  68 CB 08 00 46 5C 01 00 06 00 61 62 63 64 65 66   h...F\....abcdef
00030:  67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76   ghijklmnopqrstuv
00040:  77 61 62 63 64 65 66 67 68 69                     wabcdefghi
```

Obr. 5.8 IP-datagram odchycený pomocí MS Network Monitoru

Maximální délka záhlaví IP-datagramu je tedy omezena tím, že položka délka záhlaví má k dispozici pouze 4 bity ($1111_2=F_{16}=15_{10}$). Délka záhlaví IP-datagramu je tedy maximálně 60 B (=15x4). Jelikož povinné položky mají 20 B, tak na volitelné položky zbývá maximálně 40 B.

Typ služby (*type of service – TOS*) je položka, která v praxi nenašla svého naplnění. V normách RFC-791 a RFC-1349 lze nalézt konkrétní návrhy využití. Záměr spočíval v jistém nedostatku IP-protokolu jehož podstatou je skutečnost, že v Internetu není zaručena šíře přenosového pásma mezi účastníky. Jistého vylepšení se mělo dosáhnout právě touto položkou, pomocí které je možné označit některé IP-datagramy tak, aby byly dopravovány přednostně či aby byla zaručena rychlá odezva atp.

Celková délka IP-datagramu (*total length*) obsahuje celkovou délku IP-datagramu v bajtech. Jelikož je tato položka pouze dvojbajtová, tak maximální délka IP-datagramu je 65535 bajtů.

Identifikace IP-datagramu (*identification*) obsahuje identifikaci IP-datagramu, kterou do IP-datagramu vkládá operační systém odesílatele. Tato položka se společně s položkami **příznaky** (*flags*) a **posunutí fragmentu** (*fragment offset*) využívá mechanismem fragmentace datagramu.

Obr. 5.9
Příznaky

Do češtiny se názvy bitů pole příznaky překládají v negaci (viz obr. 5.9). Je-li DF bit nastaven na 1, pak je fragmentace zakázána. Nastavení na 0 naopak znamená, že fragmentace je možná. Je-li nastaven bit MF na jedničku, pak vyjadřuje, že není posledním fragmentem.

Doba života datagramu (*time to live – TTL*) slouží k zamezení nekonečného toulání IP-datagramu Internetem. Každý směrovač kladnou položku TTL snižuje alespoň o jedničku. Není-li už možné hodnotu snížit, IP-datagram se zahazuje a odesílateli IP-datagramu je tato situace signalizována protokolem ICMP.

Jak se hodnota položky TTL nastavuje? U příkazů ping a traceroute je možné ji explicitně nastavit. Obecně se však jedná o parametr jádra operačního systému, pokud ji tvůrci programu nenastaví explicitně).

Protokol vyšší vrstvy (*protocol*) obsahuje číselnou identifikaci protokolu vyšší vrstvy, který využívá IP-datagram ke svému transportu. V praxi se nesetkáváme s případem, že by se komunikovalo přímo IP-protokolem. Vždy je použit protokol vyšší vrstvy (TCP nebo UDP) nebo jeden ze služebních protokolů ICMP či IGMP. Protokoly ICMP a IGMP jsou sice formálně součástí protokolu IP, avšak chovají se jako protokoly vyšší vrstvy, tj. v přenášeném paketu je záhlaví IP-protokolu následováno záhlavím protokolu ICMP (resp. IGMP).

Čísla protokolů vyšších vrstev přiřazuje tvůrcům protokolů vyšších vrstev organizace IANA. Přiřazená čísla lze najít na http://www.iana.org. Pro zajímavost jsou některá čísla protokolů popisovaných v této publikaci uvedena v tab. 5.1.

Číslo protokolu vyšší vrstvy	Protokol
1	ICMP
2	IGMP
6	TCP
$17_{10}=11_{16}$	UDP

Tab. 5.1

Jako protokol vyšší vrstvy může být i protokol, který je tunelován přes Internet (*encapsulation*). Tunelovány mohou být např. protokoly, které Internet nepodporuje, jako je např. protokol IPX. Nebo může být tunelován sám protokol IP (*IP over IP*). Tunelování IP přes IP se může na první pohled jevit jako nesmyslné plýtvání. Avšak v případě, že přes Internet chceme přenášet data mezi dvěma částmi privátní sítě o adrese 10.0.0.0, pak je takový tunel nezbytností. Navíc je možné vnitřní IP-datagramy zabezpečit šifrováním a vznikne nám tak jednoduchá virtuální privátní síť (*VPN*).

Číslo protokolu vyšší vrstvy (desítkově)	Protokol
4	IP over IP
97	Ethernet within IP
111	IPX in IP

Tab. 5.2

Pokud je třeba transportovat datagramy protokolu IP verze 6 přes síť podporující pouze IP-protokol verze 4, pak také nezbývá opět nic jiného než tunelování. V tab. 5.2 jsou uvedena některá čísla pro tunelované protokoly.

Kontrolní součet z IP-záhlaví (*header checksum*) obsahuje kontrolní součet, avšak pouze ze záhlaví IP-datagramu a nikoliv z datagramu celého. Jeho význam je tedy omezený. Bližší informace o výpočtu kontrolního součtu lze nalézt v normách RFC-1071 a RFC-1141.

Problém s kontrolním součtem spočívá v tom, že když směrovač změní nějakou položku v záhlaví IP-datagramu (např. TTL změnit musí), tak musí změnit i hodnotu kontrolního součtu, což vyžaduje jistou režii směrovače.

IP-adresa odesílatele a **IP-adresa příjemce** (*source* and *destination adress*) obsahuje čtyřbajtovou IP-adresu odesílatele a příjemce IP-datagramu.

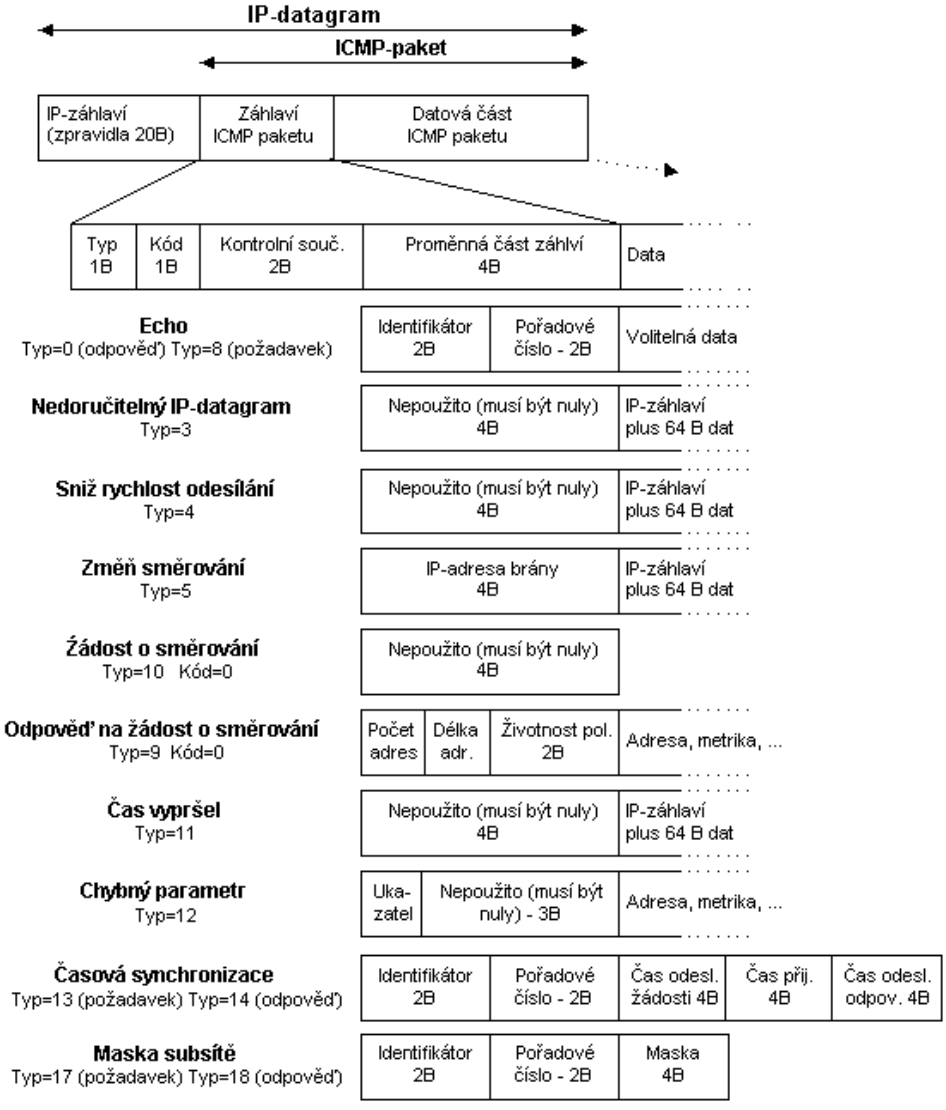
Volitelné položky jsou využívány ojediněle a zpravidla směrovače bývají nakonfigurovány tak, aby IP-datagramy s použitými volitelnými položkami byly bez okolků zahozeny.

5.2 Protokol ICMP

Protokol ICMP je služební protokol, který je součástí IP-protokolu. Protokol ICMP slouží k signalizaci mimořádných událostí v sítích postavených na IP-protokolu. Protokol ICMP svoje datové pakety balí do IP-protokolu, tj. pokud budeme prohlížet přenášené datagramy, pak v nich najdeme za linkovým záhlavím záhlaví IP-protokolu následované záhlavím ICMP paketu.

Protokolem ICMP je možné signalizovat nejrůznější situace, skutečnost je však taková, že konkrétní implementace TCP/IP podporují vždy jen jistou část těchto signalizací a navíc z bezpečnostních důvodů mohou být na směrovačích mnohé ICMP signalizace zahazovány.

Obr. 5.10
ICMP-paket



Záhlaví ICMP-paketu je vždy osm bajtů dlouhé (viz obr. 5.10). První čtyři bajty jsou vždy stejné a obsah zbylých čtyř závisí na typu ICMP-paketu.

První čtyři bajty záhlaví obsahují vždy typ zprávy, kód zprávy a šestnáctibitový kontrolní součet. Formát zprávy závisí na hodnotě pole Typ. Pole Typ je hrubým dělení ICMP-paketů. Pole Kód pak specifikuje konkrétní problém (jemné dělení), který je signalizován ICMP-protokolem.

Přehled jednotlivých typů a kódů uvádí tabulka 5.3.

Typ	Kód	Popis	Co signalizuje	Kdo zpracovává
0	0	Echo	Odpověď už. aplikaci	Uživ.aplikace
3		**Nedoručitelný IP-datagram (Destination unreacheble)**	Chyba	Uživ.aplikace
	0	Nedosažitelná síť *(Network unreachable)*		
	1	Nedosažitelný uzel *(Host unreachable)*		
	2	Nedosažitelný protokol *(Protocol unreachable)*		
	3	Nedosažitelný port protokolu UDP *(Port unreachable)*		
	4	Fragmentace zakázána, avšak pro další přenos by byla nutná *(Fragmentation needed but don't fragment bit set)*		
	5	Explicitní směrování selhalo *(Source route failed)*		
	6	Adresátova síť je neznámá *(Destination network unknown)*		
	7	Adresátův uzel je neznámý *(Destination host unknown)*		
	9	Adresátova síť je administrativně uzavřena *(Destination network administratively prohibited)*		
	10	Adresátův uzel je administrativně uzavřen *(Destination host administratively prohibited)*		
	11	Nedosažitelná síť pro uvedený typ služby *(Network unreachable for TOS)*		
	12	Nedosažitelný uzel pro uvedený typ služby *(Host unreachable for TOS)*		
	13	Komunikace administrativně uzavřena filtrací *(Communication administratively prohibited by filtering)*		
4	0	**Sniž rychlost odesílání *(Source quench)***	Chyba	**1.** Jádro OS pro TCP **2.** Zahazuje se pro UDP

Tab. 5.3 Přehled zpráv protokolu ICMP

5		**Změň směrování** ***(Redirect)***	Chyba	Jádro OS
	0	Změň směrování pro síť *(Redirect for network)*		
	1	Změň směrování pro uzel *(Redirect for host)*		
	2	Změň směrování pro síť pro daný typ služby *(Redirect for TOS and network)*		
	3	Změň směrování pro uzel pro daný typ služby *(Redirect for TOS and host)*		
8	0	**Žádost o echo** ***(Echo rquest)***	Dotaz už. aplikace	Jádro OS
9	0	**Odpověď na žádost o směrování** ***(router advertisement)***	Odpověď už. aplikaci	Uživ.proces
10	0	**Žádost o směrování** ***(router solicitation)***	Dotaz už.aplikace	Uživ.proces
11		**Čas vypršel** ***(time exceeded)***	Chyba	Uživ.proces
	0	Čas vypršel během transportu *(TTL equals 0 during transit)*		
	1	Vypršel čas na sestavení IP-datagramu z jeho fragmentů *(time to live equals 0 during reasembly)*		
12		**Chybný parametr** ***(parameter problem)***	Chyba	Uživ.proces
	0	Chybné IP-záhlaví *(IP header bad)*		
	1	Schází požadovaný volitelný parametr *(required option missing)*		
13	0	**Požadavek na časovou synchronizaci** ***(timestamp request)***	Dotaz už. aplikace	Uživ.proces
14	0	**Odpověď na časovou synchronizaci** ***(timestamp reply)***	Odpověď už. aplikaci	Jádro OS
17	0	**Žádost o masku subsítě** ***(address mask request)***	Dotaz už. aplikace	Uživ. proces
18	0	**Odpověď na žádost o masku** ***(address mask reply)***	Odpověď už. aplikace	Jádro OS

OS=Operační systém

Tab. 5.3 Přehled zpráv protokolu ICMP *(pokračování)*

Nyní se zastavme u jednotlivých typů zpráv.

5.2.3 Echo

Je jednoduchý nástroj protokolu ICMP, kterým můžeme testovat dosažitelnost jednotlivých uzlů v Internetu. Žadatel vysílá ICMP-paket „Žádost o echo“ a cílový uzel je povinen odpovědět ICMP-paketem „*Echo*“.

Všechny operační systémy podporující protokol TCP/IP obsahují program ping, kterým uživatel může na cílový uzel odeslat žádost o echo. Program ping pak zobrazuje odpověď.

Význam pole identifikátor v záhlaví ICMP-paketu spočívá ve spárování žádosti s odpovědí (aby se dalo zjistit, ke které žádosti paří příslušná odpověď).

Např. ve Windows NT chceme zjistit dostupnost uzlu 194.149.105.18:

```
D:\>ping 194.149.105.18

Pinging 194.149.105.18 with 32 bytes of data:

Reply from 194.149.105.18: bytes=32 time<10ms TTL=63
Reply from 194.149.105.18: bytes=32 time<10ms TTL=63
Reply from 194.149.105.18: bytes=32 time<10ms TTL=63
Reply from 194.149.105.18: bytes=32 time<10ms TTL=63
```

Systém odeslal čtyřikrát žádost o echo. Odpověď měla 32 bajtů dlouhou datovou část a získal ji do 10 ms. V odpovědi měla položka TTL hodnotu 63.

5.2.4 Nedoručitelný IP-datagram

Nemůže-li být IP-datagram předán dále směrem k adresátovi, pak je zahozen a odesílatel je protokolem ICMP o tom uvědomen zprávou „Nedoručitelný IP-datagram“. Jednotlivé důvody jsou uvedeny v tabulce 5.3.

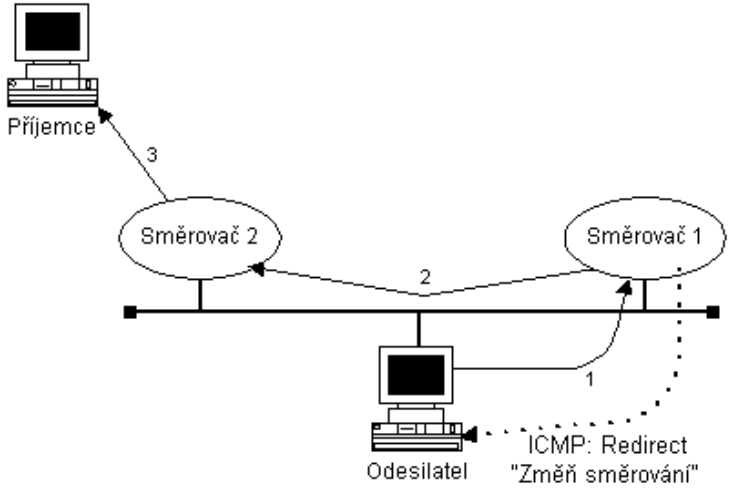
5.2.5 Sniž rychlost odesílání

Jestliže je síť mezi odesílatelem a příjemcem v některém místě přetížena, pak směrovač, který není schopen předávat dále všechny IP-datagramy signalizuje odesílateli „Sniž rychlost odesílání“. Odesílatel pak v případě, že používá protokol TCP snižuje rychlost odesílání TCP segmentů. V případě protokolu UDP se zprávy „Sniž rychlost odesílání“ ignorují.

5.2.6 Změň směrování *(Redirect)*

Pomocí tohoto ICMP-paketu se provádí dynamické změny ve směrovací tabulce.

Obr. 5.11
Redirect



Na obr. 5.11 směrovač 1 má předávat IP-datagram na stejné síťové rozhraní (*interface*), kterým IP-datagram přišel, pak jej sice předá, avšak upozorní adresáta ICMP-paketem, aby si změnil vlastní směrovací tabulku a takovou podivnou službu už více nežádal.

Tato situace nejčastěji nastává tehdy, když máme na lokální síti více směrovačů, ale jednotlivé počítače na LAN mají po svém startu pouze položku *default* ukazující na jeden ze směrovačů.

5.2.7 Žádost o směrování

Jedná se o poměrně novou záležitost, pomocí které nemusíme do směrovací tabulky počítačů na LAN ručně konfigurovat vůbec žádnou položku *default*. Počítač po svém startu vyšle oběžníkem „Žádost o směrování" a směrovač mu odpoví ICMP-paketem „Odpověď na žádost o směrování", která obsahuje: počet adres směrovače, délku adresy a pak dvojice IP-adresa a preference. Z odpovědi může počítač automaticky vygenerovat položku *default*.

Čím má preference vyšší hodnotu, tím je IP-adresa více preferována. Hodnota preference 80000000_{16} signalizuje, že tato adresa se má ze směrovací tabulky vypustit.

Směrovače odpovídají na žádost o směrování, avšak v náhodném intervalu mezi 450 a 600 vteřinami by měly oběžníkem samy do lokální sítě generovat ICMP-pakety „odpověď na žádost o směrování".

Položka „doba života" udává čas, po který je informace platná, tj. po který má být položka ve směrovací tabulce udržována.

5.2.8 Čas vypršel *(time exceeded)*

Tento typ zahrnuje dva velmi odlišné případy.

Pro kód=0 signalizuje, že položka TTL by byla na směrovači snížena na nulu, tj. že je podezření, že IP-datagram v Internetu zabloudil, proto bude zlikvidován.

Pro kód=1 signalizuje, že počítač adresáta není schopen v daném čase sestavit z fragmentů celý IP-datagram.

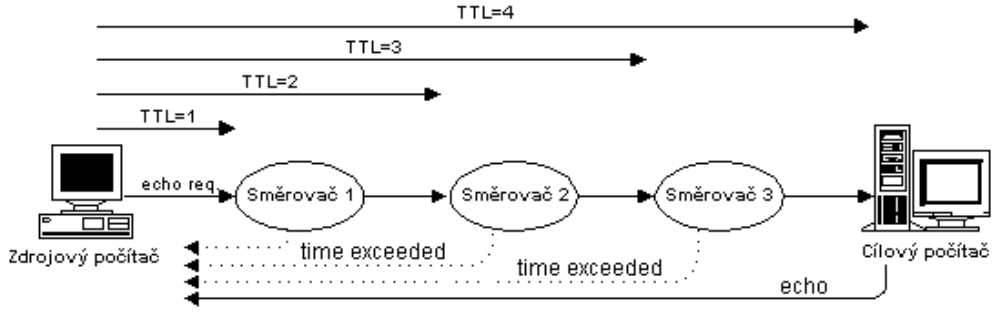
ICMP-paket čas vypršel kód=0 využívá ke své činnosti program traceroute (UNIX) i program tracert (Microsoft).

Program tracert je jednodušší. Tento program odesílá ze zdrojového počítače na cílový uzel ICMP-pakety „Žádost o echo", avšak v prvním paketu nastaví položku TTL na jedničku. První směrovač na cestě paket zahodí a vrátí ICMP-paket „Čas vypršel", protože musí zmenšit TTL alespoň o jedničku, ale tímto zmenšením už dostane nulu.

Zdrojový počítač tak od prvního směrovače na cestě dostane v IP-datagramu ICMP-paket „čas vypršel". Z položky adresa odesílatele v IP-záhlaví lze zjistit adresu prvního směrovače na cestě. Změří se časový interval od odeslání po příjem paketu a zjistí se tak čas procházky paketu od odesílatele k příjemci a zpět. Toto se opakuje třikrát a všechny tři časy se zobrazí. Na konec řádku ještě zobrazí jméno směrovače a v hranatých závorkách jeho IP-adresu. Jméno získá z reverzního překladu v DNS.

Nezíská-li v časovém limitu odpověď, zobrazí místo času hvězdičku (*).

Poté vše opakuje s hodnotou TTL=2 atd. Svou činnost ukončí, když od cílového uzlu obdrží ICMP-zprávu „Echo". K ukončení může pochopitelně také dojít, když nějaký směrovač nezná cestu k cílovému počítači, pak zdrojovému počítači zašle zprávu „nedoručitelný IP-datagram".



Obr. 5.12 Příkaz tracert

```
D:\>tracert kula.usp.ac.fj
Tracing route to kula.usp.ac.fj [144.120.8.11]
over a maximum of 30 hops:

  1   <10 ms    10 ms   <10 ms  cbuN002e00.pvt.net [194.149.104.193]
  2    10 ms    10 ms    10 ms  phucbu.pvt.net [194.149.96.13]
  3   601 ms   561 ms   641 ms  951.Hssi5-0.GW1.NYC2.ALTER.NET [157.130.0.117]
  4   591 ms   571 ms   571 ms  143.ATM2-0.XR1.EWR1.ALTER.NET [146.188.177.50]
  5   591 ms   581 ms   571 ms  193.ATM1-0-0.BR1.EWR1.ALTER.NET [146.188.176.49]

  6   400 ms   381 ms   360 ms  sl-pen-11-h3.sprintlink.net [137.39.44.130]
  7   811 ms   591 ms   661 ms  sl-bb10-pen-0-1.sprintlink.net [144.232.5.5]
  8   500 ms   651 ms   731 ms  sl-bb22-stk-6-0.sprintlink.net [144.232.8.178]
  9   871 ms   831 ms   932 ms  sl-bb23-stk-8-0.sprintlink.net [144.232.4.110]
 10   691 ms   650 ms   611 ms  sl-bb10-sj-6-0.sprintlink.net [144.232.8.193]
 11   811 ms   771 ms   771 ms  sl-gw2-sj-0-0-155M.sprintlink.net [144.232.3.38]
 12   641 ms   651 ms   641 ms  sl-cais-1.sprintlink.net [144.228.111.18]
 13   801 ms   811 ms   861 ms  hssi9-0-0.hk-T3.hkt.net [202.84.128.253]
 14   801 ms     *      811 ms  f5-0.yck06.hkt.net [205.252.130.201]
 15   821 ms   831 ms   822 ms  a6-0.tmh08.hkt.net [205.252.130.81]
 16  1402 ms  1342 ms  1362 ms  s4-3b.tmh08.hkt.net [205.252.128.158]
 17  1381 ms  1362 ms  1352 ms  202.84.251.6
 18  1362 ms  1362 ms  1352 ms  202.62.120.6
 19  1422 ms  1372 ms  1392 ms  202.62.125.134
 20  1412 ms  1382 ms  1412 ms  kula.usp.ac.fj [144.120.8.11]

Trace complete.
```

Program traceroute pracuje na obdobném principu, avšak neodesílá ICMP-pakety „Echo request“, ale generuje protokolem UDP datagramy (UDP-port je možné modifikovat parametrem –p). Je-li na cestě k cílovému počítači použita filtrace na směrovači, pak vhodnou volbou čísla UDP-portu lze mnohdy nalézt „díru“ ve filtru a objevit cestu až k cílovému počítači. Dobrým typem je pro takový případ číslo portu 53 (-p 53), který používá DNS.

```
$ /usr/sbin/traceroute -p 20000 libor.pvt.net
traceroute to libor.pvt.net (194.149.104.198), 30 hops max, 40 byte packets
 1  cbuN003f00.pvt.net (194.149.105.17)  1 ms  1 ms  1 ms
 2  Libor.pvt.net (194.149.104.198)  1 ms  1 ms  1 ms
```

Cílový počítač zpravidla odpoví ICMP-paketem „Port unreacheble“ (typ=3, kód=3). Kromě času a hvězdičky program traceroute ještě může vypsat !H (nedostupný uzel), !N (nedostupná síť), !A (síť administrativně uzavřena) či !S (explicitní směrování selhalo).

5.2.9 Žádost o masku

Tímto ICMP-paketem může bezdisková stanice žádat o masku své sítě poté, co protokolem RARP obdržela svou IP-adresu.

Tento mechanismus je v praxi dnes již málo běžný. Stanice může získat masku své sítě protokolem BOOTP, kterým získá i další informace. Avšak i protokol BOOTP je dnes vytlačován protokolem DHCP, který je komplexnější, tj. poskytuje více informací. Protokoly BOOTP a DHCP jsou aplikační protokoly.

5.2.10 Časová synchronizace

Tímto ICMP-paketem se žádá cílový počítač o čas. Mechanismus je znázorněn na obr. 5.13.

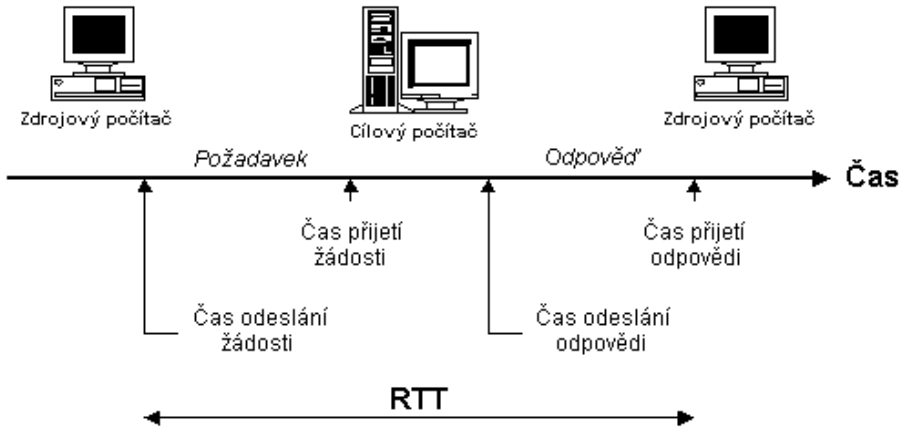
Zdrojový počítač do ICMP-paketu „**Požadavek na časovou synchronizaci (*timestamp request*)**“ vyplní čas odeslání žádosti.

Cílový počítač vyplní do své odpovědi „**Odpověď na časovou synchronizaci (*timestamp reply*)**“ dva časy:

- Čas přijetí žádosti.
- Čas odeslání odpovědi.

Zdrojový počítač si zjistí čas přijetí odpovědi (ten se pochopitelně nepřepravuje v žádném ICMP-paketu). Odečtením času odeslání požadavku od času přijetí odpovědi se získá doba procházky od zdrojového počítače k cílovému a zpět (anglicky *Round Trip Time – RTT*).

Obr. 5.13 Časová synchronizace



Čas se udává v milisekundách od poslední půlnoci Světového času – GMT. (Místo GMT bychom měli správně dnes psát UTC – *Coordinated Universal Time*, jde spíše o zvyk).

5.3 Fragmentace

IP-datagramy jsou baleny do linkových rámců. Linkové protokoly umožňují přenášet ve svých datových rámcích data pouze do určité maximální velikosti. Tato maximální velikost dat, která lze vložit do jednoho linkového rámce se označuje MTU (*Maximum Transfer Unit*).

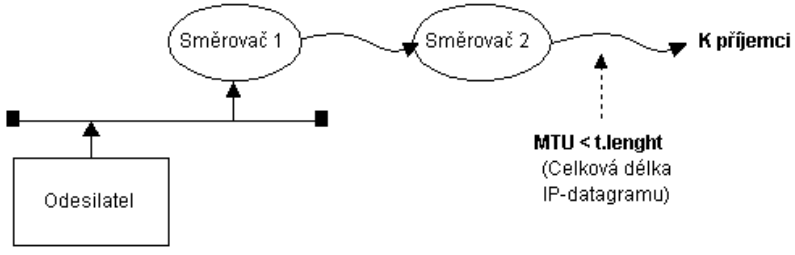
Tab. 5.4

Linkový protokol	MTU
Ethernet	1500
FDDI	4478

Z tabulky 5.4 je zřejmé, že linkové protokoly mají nejčastěji MTU řádkově v jednotkách KB. Na linkách spojujících vzdálené lokality se někdy dokonce setkáváme s MTU menším než 1 KB. Pole celková délka IP-datagramu je však dlouhé 16 bitů, takže teoreticky je možné vytvořit IP-datagram až 64 KB dlouhý.

Co se však stane, když IP-datagram na své pouti od odesílatele k příjemci dorazí na směrovač (na obrázku 5.14 směrovač 2) z něhož směrem k příjemci vede linka, která má menší MTU než je velikost našeho IP-datagramu.

Obr. 5.14



Směrovač není schopen takový IP-datagram poslat dále. Směrovač se rozhoduje co dále na základě příznaku „Fragmentace možná" (*DF bit*) v záhlaví IP-datagramu (ponecháváme stranou možnost, že k příjemci vede ještě jiná linka, byť s horší metrikou). Příznak „Fragmentace možná" může být buď nastaven nebo ne. Jsou tedy dvě možnosti:

1. Fragmentace je možná, pak se provede fragmentace, jak je popsáno dále v této kapitole.
2. Fragmentace není možná, pak směrovač IP-datagram zahodí a odesílatele o tom informuje ICMP-signalizací „Fragmentace zakázána, avšak pro další přenos by byla nutná (*Fragmentation needed but don't fragment bit set*)".

Zakážeme-li příznakem fragmentaci, pak můžeme i zjistit jaké nejmenší MTU je mezi odesílatelem a příjemcem, tj. jak velké IP-datagramy nebude nutné fragmentovat.

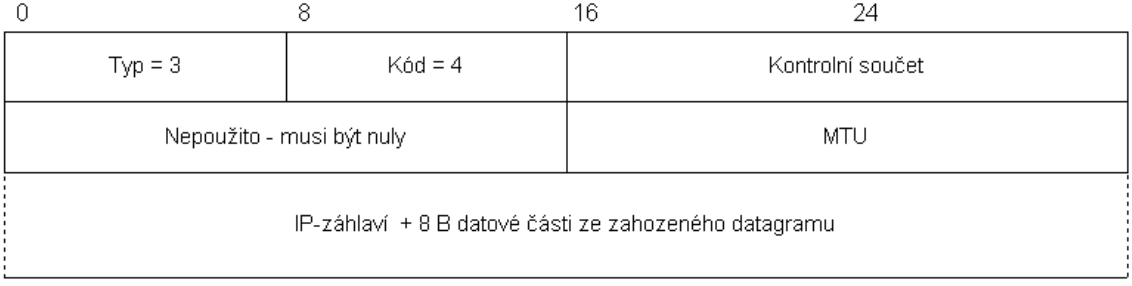
Může nám k tomu posloužit např. příkaz ping. Implementace příkazu ping firmy Microsoft umožňuje zakázat fragmentaci pomocí parametru –f a nastavit délku IP-datagramu pomocí parametru -l. Takže příkaz

```
C:\> ping -f -l 2000 příjemce
```

buďto sdělí, že příjemce je funkční a zobrazí nám čas procházky odesílatel-příjemce-odesílatel (RTT) nebo naopak zobrazí chybovou hlášku, takže se dozvíme, zdali na cestě byla fragmentace nutná pro IP-datagram dlouhý 2000 B. V případě, že fragmentace byla nutná můžeme velikost odesílaného IP-datagramu zmenšit a pozorovat, zdali tentokrát bude fragmentace nutná či nikoliv. Tak můžeme postupovat tak dlouho, až zjistíme hranici, od které je fragmentace nutná.

Podstatně jednodušší by bylo, kdyby ICMP-signalizace obsahovala hodnotu MTU, která platí pro inkriminovanou linku. Původně nebylo s touto možností počítáno, avšak později byl ICMP-paket pro tento případ doplněn o pole MTU. Tato možnost je jen zřídka implementována.

V ICMP-paketu byly využity druhé dva bajty z nevyužitých čtyř bajtů záhlaví. Stuktura ICMP-paketu je znázorněna na obr. 5.15.

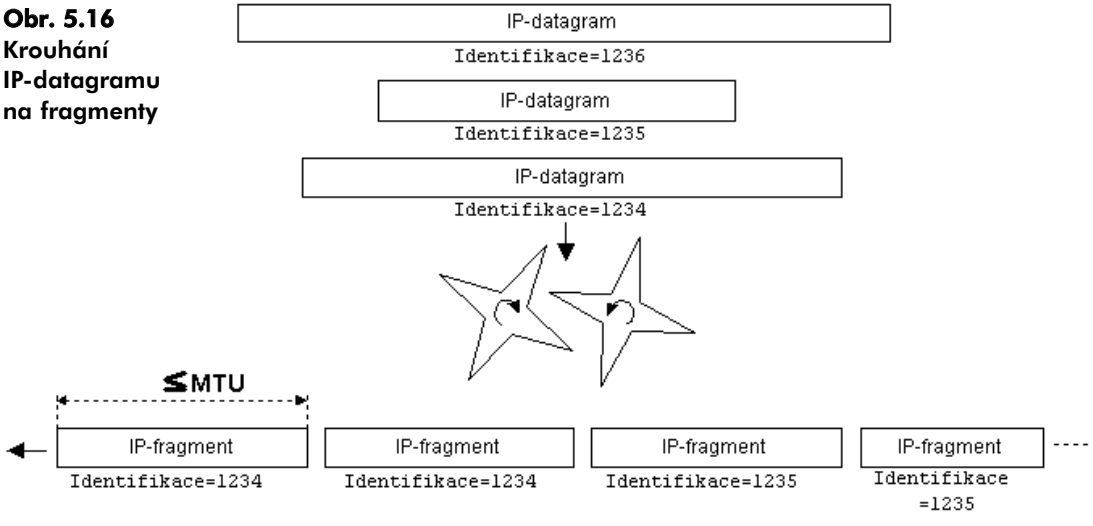


Obr. 5.15 Rozšíření ICMP zprávy „fragmentace zakázána"

Pokud je pole MTU nulové, pak směrovač nepodporuje toto nové rozšíření.

Nyní se vraťme k situaci, kdy v IP-paketu je nastaveno, že fragmentace je možná, pak směrovač dělí delší IP-datagramy na fragmenty, jejichž celková délka (*total length*) je menší nebo rovná MTU následující linky – viz obr. 16.

Obr. 5.16 Krouhání IP-datagramu na fragmenty



Každý IP-datagram má ve svém záhlaví svou identifikaci, kterou dědí i jeho fragmenty. Díky identifikaci příjemce pozná, ze kterých fragmentů má datagram složit. Nikdo jiný než příjemce není oprávněn z fragmentů skládat původní datagram, tj. ani např. směrovač ze kterého vede linka s takovým MTU, do kterého by se celý datagram již vešel. Důvod je prostý, Internet negarantuje, že jednotlivé fragmenty půjdou stejnou cestou (ani negarantuje pořadí v jakém dojdou). Takže směrovač, který by se pokoušel datagram sestavit by mohl být na závadu spojení, protože fragmentů, které by šly jinou cestou by se nikdy nedočkal.

Identifikace IP-datagramů může být jednoznačná pouze v rámci jednoho protokolu vyšší vrstvy, protože záhlaví IP-datagramu obsahuje ještě pole „Protokol vyšší vrstvy". Globální identifikace může být brána jako zřetězení polí identifikace a protokol vyšší vrstvy (+ pochopitelně IP-adresa odesílatele a příjemce). Takže teoreticky mohou být za sebou odeslány dva IP-datagramy o stejné identifikaci, jeden však nese TCP-paket a druhý UDP-paket. Toto je opět méně běžná implementace.

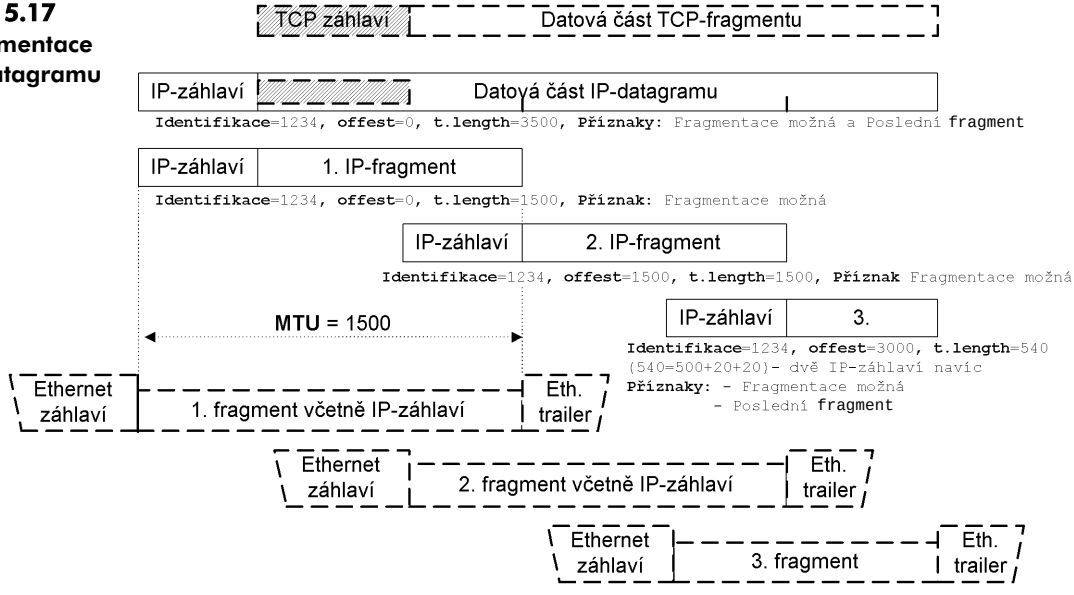
Každý fragment tvoří samostatný IP-datagram. Při fragmentaci je nutné vytvořit pro každý fragment nové IP-záhlaví. Některé údaje (jako protokol vyšší vrstvy, či IP-adresa odesílatele a příjemce) se získají ze záhlaví původního IP-datagramu.

Při fragmentaci vstupuje do hry pole „Posunutí fragmentu od počátku IP-datagramu (*fragment offset*)", které vyjadřuje kolik bajtů datové části původního IP-datagramu bylo vloženo do předchozích fragmentů. Pole „Celková délka IP-datagramu (*total length*)" obsahuje délku fragmentu, nikoliv délku původního datagramu. Aby příjemce poznal jak je původní datagram dlouhý, tak je poslední fragment opatřen příznakem „Poslední fragment". Celý mechanismus je znázorněn na obr. 5.17.

Síť nerozlišuje mezi přenosem fragmentu a přenosem celého (nefragmentovaného) IP-datagramu. Nefragmentovaný datagram je fragment s posunutím nula a příznakem „Poslední fragment". Proto se často slova IP-datagram a fragment zaměňují.

Mechanismus fragmentace umožňuje i fragmentovat fragment dorazí-li na směrovač jehož odchozí linka má ještě menší MTU.

Obr. 5.17 Fragmentace IP-datagramu



Důležité je, že každý další fragment znamená zatížení o minimálně 20 B jeho záhlaví. Pro zajímavost je na obrázku 5.17 znázorněn také TCP-paket, který je vložen do IP-paketu. Co je na tom zajímavého?

Zajímavé je to, že TCP-záhlaví je obsaženo pouze v prvním IP-fragmentu. Takže pokud se provádí na směrovači filtrace IP-datagramů na základě nejen informací z IP-záhlaví, ale též na základě informací z TCP-záhlaví, pak lze filtrovat pouze první fragment a ostatní se propouští. Příjemce pak po určeném časovém intervalu zjistí, že mu schází první fragment z IP-datagramu a signalizuje to příjemci ICMP-zprávou „Vypršel čas na sestavení IP-datagramu z jeho fragmentů (*time to live equals 0 during reasembly)*". Takže při filtraci TCP-paketů je nutné nezapomenout v protisměru filtrovat i tyto ICMP-pakety, pokud útočníkovi nechceme poskytnout informaci o tom, že se chráníme filtrací.

Fragmentace je považována za jakési nutné zlo, aplikace vyžadující extrémně bezpečnou komunikaci fragmentaci zakazují.

5.4 Volitelné položky IP-záhlaví

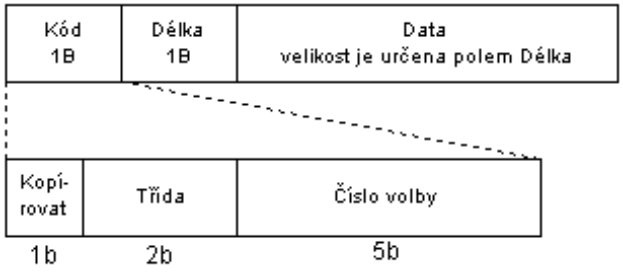
Volitelné položky v záhlaví IP-datagramu patří k zajímavostem protokolů TCP/IP. Ukážeme si jak nebezpečné může být využití těchto položek a proč mnozí poskytovatelé Interentu IP-datagramy s některými volitelnými položkami zahazují. Z hlediska protokolu TCP/IP je však toto jednání poskytovatelů neomluvitelné (byť v dobré víře) a lze je přirovnat k doporučení, aby každý občan s sebou nosil berle pro případ, že by si zlomil nohu.

Pokud příjemce obdrží IP-datagram s některou z voleb, pak v odpovědi by rovněž měl použít tuto volbu.

Volitelné položky rozšiřují IP-záhlaví. Vzhledem k omezené délce IP-záhlaví na 60 B (z čehož je 20 B povinných) jsou volitelné položky shora omezeny 40 B. Tč. existuje několik možností rozšíření IP-záhlaví:

1. Zaznamenávej směrovače (*record route*).
2. Zaznamenávej čas (*timestamp*).
3. Explicitní směrování (*loose source routing*).
4. Striktní explicitní směrování (*strict source routing*).
5. Upozornění pro směrovač (*IP Router Alert Option).*
6. Bezpečnostní omezení dle normy RFC-1108.

Volitelné položky v IP-záhlaví následují povinné položky. Volitelné položky mají obecně formát znázorněný na obr. 5.18.



Obr. 5.18
Volitelné položky záhlaví IP-datagramu

Kde bit **Kopírovat** specifikuje, je-li nastaven na 1, že tato volba má být kopírována do všech fragmentů vzniklých z tohoto datagramu. Je-li bit nastaven na 0, pak se kopíruje pouze do záhlaví prvního fragmentu.

Dva bity tvořící pole **Třída** nabývají:

- Hodnoty 0 v případě, že se jedná o IP-datagram nesoucí běžná data nebo data určená pro řízení sítě.
- Hodnoty 2 (=10_2) v případě, že se jedná o IP-datagram sloužící k ladění nebo měření sítě.

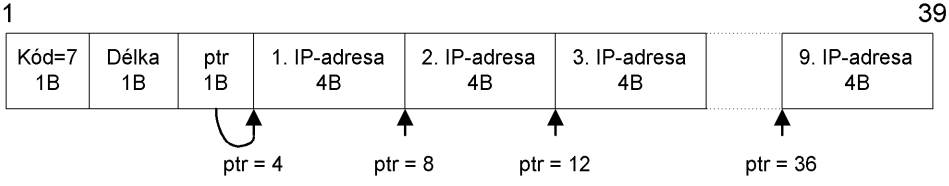
Pole Číslo volby pak specifikuje konkrétní volbu. Často používané kódy jsou uvedeny v tab. 5.5.

Kód	Šestnáctkově	Desítkově	Délka	Volba
0 00 00000	00	00	Není	Konec voleb (End of options list). Použije se v případě, že volby nekončí s koncem IP-záhlaví. Pole délka a data se nepoužijí.
0 00 00001	01	1	Není	Prázdná volba – výplň záhlaví na násobek čtyř bajtů. Pole délka a data se nepoužijí.
0 00 00111	07	7	Proměnná	Zaznamenávej směrovače (Record route).
0 10 00100	44	68	Proměnná	Zaznamenávej čas (Timestamp).
1 00 00011	83	131	Proměnná	Explicitní směrování (Loose source routing).
1 00 01001	89	137	Proměnná	Explicitní striktní směrování (Strict source routing).
1 00 10100	94	148	4	Upozornění pro směrovač (IP Router Alert Option).

Tab. 5.5

5.4.1 Zaznamenávej směrovače

Obsahuje-li záhlaví volbu s vyplněným polem kód=7, pak všechny směrovače na cestě k příjemci IP-datagramu doplní do IP-záhlaví IP-adresu svého výstupního rozhraní. Jednotlivá čtyřbajtová pole v záhlaví IP-datagramu pro IP-adresy se nazývají sloty. Do IP-záhlaví je možno vložit až 9 slotů pro IP-adresy.



Obr. 5.19 Volitelná položka záhlaví „Zaznamenávej směrovače"

Pole délka obsahuje celkovou délku rozšíření a pole ptr ukazuje na první volný slot, který je možno vyplnit (další směrovač vždy zapíše novou IP-adresu a zvětší položku ptr o 4).

Datagram na své pouti od odesílatele k příjemci nasbírá do svých slotů odchozí IP-adresy. Důležité je, že pokud odesílatelův stroj rovněž podporuje tuto volbu, pak tuto volbu ve své odpovědi rovněž použije a navíc do odesílaného datagramu nejprve zkopíruje všechny sloty z datagramu přijatého.

Takže příkazem ping podporujícím volbu „zaznamenej směrovače" zjistíme seznam odchozích adres nejen při cestě datagramu od odesílatele k příjemci, ale i cestou zpět.

V příkazu ping implementovaném firmou Microsoft lze rozšíření „zaznamenávej směrovače" vytvořit pomocí parametru –r následovaného počtem vytvořených slotů. Např.

```
D:\>ping -r 5 ns.pvt.net
```

Vygeneruje ICMP-paket s rozšířením „zaznamenávej směrovače" s pěti sloty pro IP-adresy. Přičemž odesílatel vytvoří sice pět slotů, ale ani jeden není naplněn, takže ukazatel prvního volného slotu ptr ukazuje na první slot.

IP-záhlaví je rozšířeno o 3 + 5 x 4 = 23 bajtů, jak je vidět z odchyceného IP-datagramu na obr. 5.20.

```
+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP: DOD Internet Protocol
  IP: ID = 0x673D; Proto = ICMP; Len: 84
      IP: Version = 4 (0x4)
      IP: Header Length = 44 (0x2C)
    + IP: Service Type = 0 (0x0)
      IP: Total Length = 84 (0x54)
      IP: Identification = 26429 (0x673D)
    + IP: Flags Summary = 0 (0x0)
      IP: Fragment Offset = 0 (0x0) bytes
      IP: Time to Live = 32 (0x20)
      IP: Protocol = ICMP - Internet Control Message
      IP: Checksum = 0xCB51
      IP: Source Address = 194.149.104.198
      IP: Destination Address = 194.149.105.18
      IP: Option Fields = 7 (0x7)
          IP: Record Route Option = 7 (0x7)
              IP: Option Length = 23 (0x17)
              IP: Next Slot Pointer = 4 (0x4)
              IP: Route Traveled = 0 (0x0)
          IP: End of Options = 0 (0x0)
      IP: Data: Number of data bytes remaining = 40 (0x0028)
+ ICMP: Echo,     From 194.149.104.198 To   194.149.105.18

00000:  00 60 3E 1D 90 00 00 20 AF FA 25 89 08 00 4B 00   .`>.... ..%...K.
00010:  00 54 67 3D 00 00 20 01 CB 51 C2 95 68 C6 C2 95   .Tg=.. ..Q..h...
00020:  69 12 07 17 04 00 00 00 00 00 00 00 00 00 00 00   i...............
00030:  00 00 00 00 00 00 00 00 00 00 08 00 1C 5C 01 00   .............\..
00040:  30 00 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E   0.abcdefghijklmn
00050:  6F 70 71 72 73 74 75 76 77 61 62 63 64 65 66 67   opqrstuvwabcdefg
00060:  68 69                                             hi
```

Obr. 5.20

5

Uživateli se zobrazí odpověď:

```
Pinging ns.pvt.net [194.149.105.18] with 32 bytes of data:

Reply from 194.149.105.18: bytes=32 time<10ms TTL=63
    Route: 194.149.105.17 ->
           194.149.105.18 ->
           194.149.104.193
```

Takže na cestě mezi odesílatelem a příjemcem (194.149.105.18) a zpět je jeden směrovač, který má směrem k příjemci adresu 194.149.105.17 a směrem k odesílateli adresu 194.149.104.193 (odesílatelem rozumíme uživatele, který vydal příkaz ping).

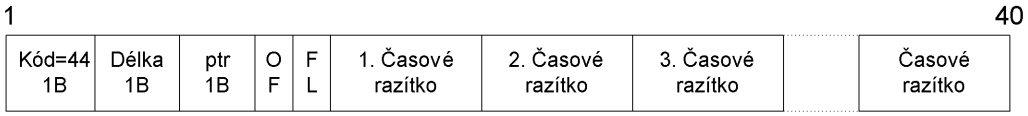
Tato odpověď vznikla z ICMP-paketu Echo. Odchycená odpověď s vyplněnými sloty je na obr. 5.21

```
+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP: DOD Internet Protocol
  IP: ID = 0x2DD8; Proto = ICMP; Len: 84
      IP: Version = 4 (0x4)
      IP: Header Length = 44 (0x2C)
    + IP: Service Type = 0 (0x0)
      IP: Total Length = 84 (0x54)
      IP: Identification = 11736 (0x2DD8)
    + IP: Flags Summary = 0 (0x0)
      IP: Fragment Offset = 0 (0x0) bytes
      IP: Time to Live = 63 (0x3F)
      IP: Protocol = ICMP - Internet Control Message
      IP: Checksum = 0x3334
      IP: Source Address = 194.149.105.18
      IP: Destination Address = 194.149.104.198
      IP: Option Fields = 7 (0x7)
          IP: Record Route Option = 7 (0x7)
              IP: Option Length = 23 (0x17)
              IP: Next Slot Pointer = 16 (0x10)
              IP: Route Traveled = 194 (0xC2)
                  IP: Gateway = 194.149.105.17
                  IP: Gateway = 194.149.105.18
                  IP: Gateway = 194.149.104.193
          IP: End of Options = 0 (0x0)
      IP: Data: Number of data bytes remaining = 40 (0x0028)
+ ICMP: Echo Reply, To 194.149.104.198 From 194.149.105.18

00000:  00 20 AF FA 25 89 00 60 3E 1D 90 00 08 00 4B 00   . ..%..`>.....K.
00010:  00 54 2D D8 00 00 3F 01 33 34 C2 95 69 12 C2 95   .T-...?.34..i...
00020:  68 C6 07 17 10 C2 95 69 11 C2 95 69 12 C2 95 68   h......i...i...h
00030:  C1 00 00 00 00 00 00 00 00 00 00 00 24 5C 01 00   ............$\..
00040:  30 00 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E   0.abcdefghijklmn
00050:  6F 70 71 72 73 74 75 76 77 61 62 63 64 65 66 67   opqrstuvwabcdefg
00060:  68 69                                             hi
```

Obr. 5.21

5.4.2 Zaznamenávej čas

Tato volba je obdobou volby zaznamenávej směrovače. Každý směrovač zapíše do záhlaví IP-datagramu časové razítko, kdy datagram směrovačem procházel. Čas se opět zaznamenává v milisekundách od poslední půlnoci GMT (obdobně jako u časové synchronizace ICMP).




```
OF - overflow (4b) - směrovač nastaví toto pole, když už nemá volný slot
     na přidání razítka.

FL  - flags (4b)
  FL = 0 Každý směrovač do slotu zapíše pouze čas, nikoliv IP-adresu
         (slot 4B dlouhý)
  FL = 1 Každý směrovač do slotu zapíše čas i IP-adresu (slot 8B dlouhý)
         Odesilatel inicializuje až čtyři sloty dlouhé 8B, tj. pro IP-adresu i
         pro časové razítko. Odesilatel vyplní IP-adresy směrovačů, když
         datagram pak prochází přes směrovač uvedený v tomto seznamu, pak tento
         směrovač do slotu doplní časové razítko.
```

Obr. 5.22 Volitelná položka „Zaznamenávej čas"

Pole kód pro tuto volbu má hodnotu $44_{16} = 68_{10}$. Formát této volby je rozšířen o dvě čtyřbitová pole OF a FL.

Příkaz ping implementovaný firmou Microsoft pomocí parametru –s vygeneruje ICMP-paket s požadavkem „zaznamenávej čas". Číslo uvedené za parametrem –s udává počet alokovaných slotů. V případě, že se požaduje jak časové razítko, tak IP-adresa, pak je možno alokovat max. 4 sloty.

```
D:\>ping -s 3 194.149.105.18
```

Vygeneruje IP-datagram (zkrácený výpis):

```
….
          IP: Option Fields = 68 (0x44)
          IP: Internet Timestamp Option = 68 (0x44)
              IP: Option Length = 28 (0x1C)
              IP: Time pointer = 5 (0x5)
              IP: ....0001 = Both time stamps and IP addresses
              IP: Missed stations = 0 (0x0)
              IP: Time Route = 0 (0x0)
                  IP: Gateway = 0.0.0.0
                  IP: Time Point = 0 (0x0)
                  IP: Gateway = 0.0.0.0
                  IP: Time Point = 0 (0x0)
                  IP: Gateway = 0.0.0.0
                  IP: Time Point = 16792576 (0x1003C00)
          IP: Data: Number of data bytes remaining = 40 (0x0028)
….
```

Uživateli se pak zobrazí IP-adresy a časová razítka. Milisekundy je nutno převést na hodiny, minuty, vteřiny a nesmí se také zapomenout na letní čas.

```
Pinging 194.149.105.18 with 32 bytes of data:

Reply from 194.149.105.18: bytes=32 time<10ms TTL=63
    Timestamp: 194.149.105.17 :  52251609 ->
               194.149.105.18 :  52531841 ->
               194.149.104.193 : 52251610
```

Což přinesl IP-datagram (zkrácený výpis):

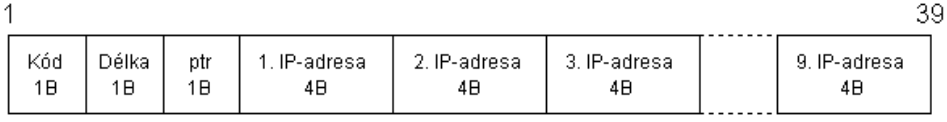
```
…
           IP: Option Fields = 68 (0x44)
           IP: Internet Timestamp Option = 68 (0x44)
               IP: Option Length = 28 (0x1C)
               IP: Time pointer = 29 (0x1D)
               IP: ....0001 = Both time stamps and IP addresses
               IP: Missed stations = 0 (0x0)
               IP: Time Route
                   IP: Gateway = 194.149.105.17
                   IP: Time Point = 52251609 (0x31D4BD9)
                   IP: Gateway = 194.149.105.18
                   IP: Time Point = 52531841 (0x3219281)
                   IP: Gateway = 194.149.104.193
                   IP: Time Point = 52251610 (0x31D4BDA)
            IP: Data: Number of data bytes remaining = 40 (0x0028)
….
```

5.4.3 Explicitní směrování

Explicitní směrování *(source routing)* umožňuje explicitně zadat přes které směrovače má být IP-datagram Internetem dopravován. Což je dobrá zpráva pro hackera, protože pro něj to znamená, že pomocí explicitního směrování lze odklonit dopravu IP-datagramů dle jeho potřeby.

Rozeznáváme dva typy explicitního směrování:

1. Explicitní směrování (kód = 83_{16}), kdy IP-datagram je dopravován přes vyjmenované směrovače. Vyjmenovat se však nemusí všechny směrovače přes které je IP-datagram dopravován.
2. Explicitní striktní směrování (kód = 89_{16}), kdy seznam směrovačů musí obsahovat všechny směrovače, přes které je IP-datagram směrován. V případě, že by bylo nutné směrovat datagram přes jiný směrovač, pak směrování selže.

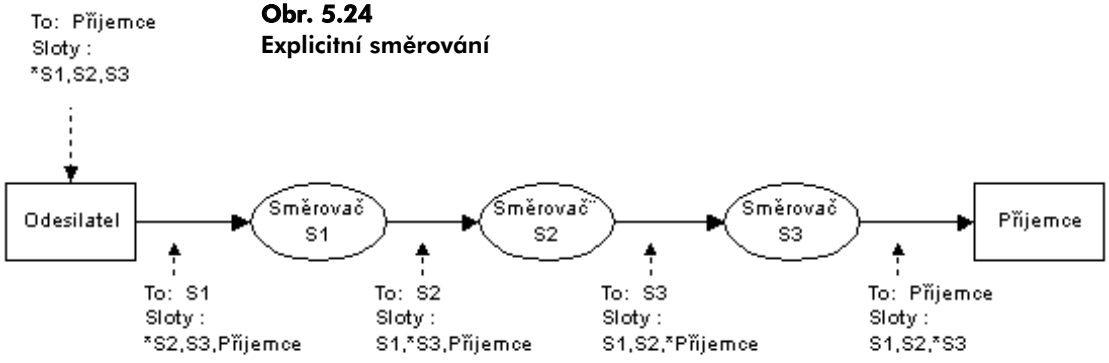


Obr. 5.23 Volitelná položka „Explicitní směrování"

Mechanismus explicitního směrování je poměrně komplikovaný. Jednotlivé zúčastněné směrovače opravují nejen pole ptr, ale dokonce i adresu příjemce v IP-datagramu.

I když odesílatel adresuje z aplikace přímo příjemce, tak ve skutečnosti adresa příjemce v IP-datagramu vždy obsahuje následující směrovač (následující hop) ze seznamu směrovačů. Celý proces automaticky zabezpečuje vrstva IP-protokolu, která na odesílatelově stroji vezme z prvního slotu IP-adresu, kterou nahradí původní adresu příjemce. Obsah jednotlivých slotů posune vlevo (první slot se uvolnil po vložení adresy do pole pro adresu příjemce). Do posledního (volného) slotu uschová původní adresu příjemce. Ukazatel ptr ukazuje na slot s IP-adresou ob jeden hop dále.

Obdobně postupují i následující směrovače. Celý proces je znázorněn na obrázku 5.24, kde hvězdička vyjadřuje slot, na který ukazuje pole ptr:



Obr. 5.24
Explicitní směrování

Příkaz ping implementovaný firmou Microsoft umožňuje specifikovat explicitní směrování pomocí paramerů –j pro explicitní směrování a parametru –k pro explicitní striktní směrování. Parametr je následován seznamem IP-adres přes které má být směrováno.

Příklad:

```
D:\>ping -j 195.47.1.1 10.1.1.1
Výpis:

 ….
       IP: Source Address = 194.149.104.198
       IP: Destination Address = 195.47.1.1
       IP: Option Fields = 131 (0x83)
           IP: Loose Source Routing Option = 131 (0x83)
               IP: Option Length = 7 (0x7)
               IP: Routing Pointer = 4 (0x4)
               IP: Route To Go
                    IP: Gateway = 10.1.1.1
           IP: End of Options = 0 (0x0)
       IP: Data: Number of data bytes remaining = 40 (0x0028)
 ….
```

Uživatel obdrží např. hlášku (aby vše nebylo bez chyby):

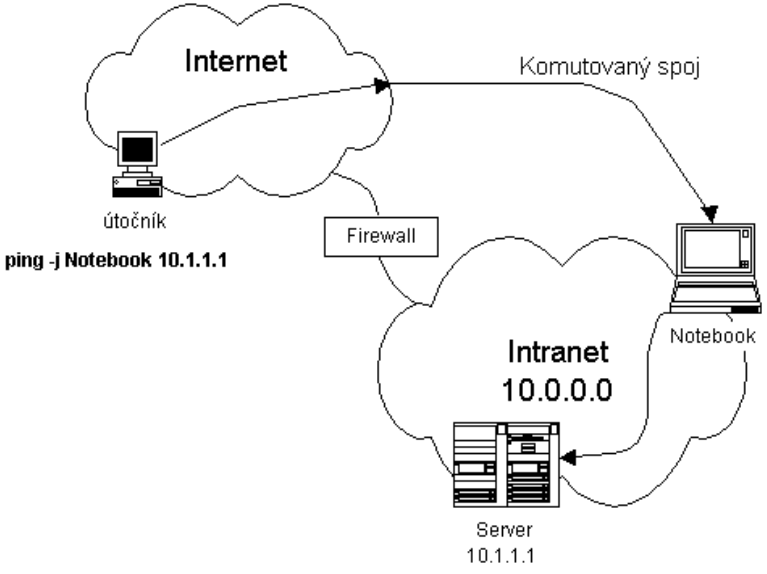
```
Pinging 172.17.101.1 with 32 bytes of data:
Reply from 194.149.104.193: Invalid source route specified.
```

Tato hláška je důsledkem skutečnosti, že na uvedeném směrovači bylo zakázáno explicitní směrování.

Proč se explicitní směrování zakazuje? Důvody jsou bezpečnostní. Explicitní směrování lze zneužít dvěma způsoby.

- Pomocí explicitního směrování odklonit dopravu IP-datagramů přes jiný směrovač, na kterém budu data sledovat, případně měnit.
- Pomocí explicitního směrování je možné útočit z Internetu dovnitř do intranetu, i když intranet používá adresaci pro privátní sítě (např. 10.0.0.0/8). Tyto privátní sítě nejsou z Internetu přímo adresovatelné – je to jedna z možných ochran intranetů. Pro průchod se použije buď přímo firewall, pokud umožňuje explicitní směrování (méně pravděpodobná varianta). Schůdnější je cesta přes počítač, který z nějakých důvodů má přímou konektivitu do Internetu. Např. se jedná o notebook zaměstnance, který se může v případě, že není na pracovišti přímo přes komutovanou linku připojit do Internetu. Pro práci na pracovišti má pak síťovou karu pro práci na LAN. Stačí, když naváže spojení na obě strany, bude mít v operačním systému povoleno předávání (*IP-forwarding*) a bude podporovat explicitní směrování. Mechanismus je znázorněn na obr. 5.25.

Obr. 5.25
Útok proti intranetu za využití explicitního směrování



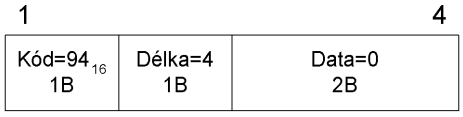
5.4.4 Upozornění pro směrovač *(IP Router Alert Option)*

IP-datagram je dopravován Internetem přes řadu směrovačů. Směrovač se za normálních okolností obsahem dopravovaného IP-datagramu příliš nezabývá, podobně jako při dopravě pošty se poštovní úředníci nezabývají obsahem přepravovaných dopisů.

Jenže kromě běžných IP-datagramů jsou Internetem dopravovány i datagramy směrovacích protokolů, které jsou směrovačům určeny. Tyto IP-datagramy mají vyplněnu IP-adresu příjemce (byť se jedná o adresu směrovače). Směrovače na cestě k adresátovi (k cílovému směrovači) tyto IP-datagramy zpracovávají jako jakékoliv jiné IP-datagramy. Avšak informace nesená v těchto IP-datagramech může být zajímavá i pro směrovače na cestě, tj. ty, kterým není přímo adresována. Za normálních okolností se směrovač na cestě ani nedozví, že předával informace, které mu mohly být užitečné. odesílatel zase nevěděl, že na cestě je nějaký směrovač, kterému by měl takovou informaci přímo poslat.

Upozornění pro směrovač je volba v záhlaví IP-datagramu, která říká všem směrovačům na cestě: „Tento IP-datagram není sice přímo pro tebe určen, ale přenáší informace, které mohou být i pro tebe zajímavé. Pokud to umíš, tak se na tyto informace podívej a využij je."

Obr. 5.26 Volitelná položka „Upozornění pro směrovač"



5.5 Protokoly ARP a RARP

Jsem-li stanice na lokální síti a chci protokolem IP komunikovat s jinou stanicí na téže síti, pak ji v protokolu IP adresuji čtyřbajtovou IP-adresou. Pro komunikaci znám IP-adresu odesílatele (svou IP-adresu) a IP-adresu příjemce. Jsem tedy schopen sestavit IP-datagram. Jenže potíž je v tom, že tento IP-datagram musí být zabalen do linkového rámce – např. do ethernetového rámce. Abych vytvořil ethernetový rámec, tak potřebuji linkovou (6B) adresu příjemce i odesílatele. odesílatel jsem já a svou linkovou adresu znám, avšak neznám linkovou adresu příjemce. Jak takovou adresu zjistím? To řeší protokol ARP.

Protokol ARP (*Address Resolution Protocol*) řeší problém zjištění linkové adresy protější stanice ze znalosti její IP-adresy. Řešení je jednoduché, do LAN vyšle linkový oběžník (linková adresa FF:FF:FF:FF:FF:FF) s prosbou: „Já stanice o linkové adrese HW1, IP-adrese IP1, chci komunikovat se stanicí o IP-adrese IP2, kdo mi pomůže s nalezením linkové adresy stanice o IP-adrese IP2? Stanice IP2 takovou žádost uslyší a odpoví. V odpovědi uvede svou linkovou adresu HW2.

ARP-paket je balen přímo do Ethernetu, tj. nepředchází mu žádné IP-záhlaví. Protokol ARP je vlastně samostatný, na IP nezávislý protokol. Proto jej mohou používat i jiné protokoly, které s protokoly TCP/IP nemají nic společného.

Pole **typ linkového protokolu** specifikuje linkový protokol používaný na LAN. Linkovému protokolu Ethernet II je vyhrazeno číslo 1. Seznam přidělených čísel je uveřejněn na http://www.iana.org.

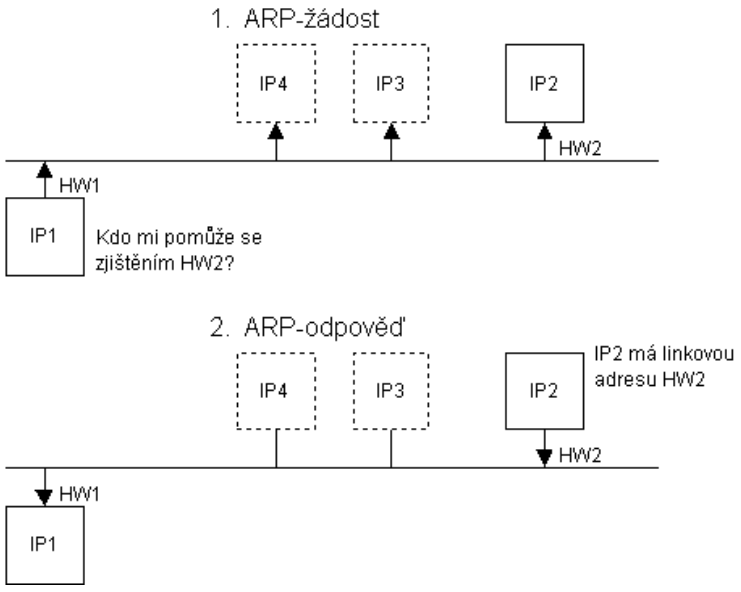
Typ síťového protokolu specifikuje typ síťového protokolu, používají se stejná čísla jako pro pole protocol v protokolu Ethernet II, tj. IP-protokol má přiděleno číslo 800_{16}.

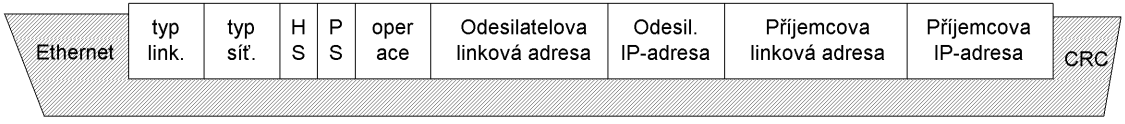
Pole **HS** určuje délku linkové adresy a pole **PS** délku síťové adresy. Standardně je tedy HS=6 a PS=4.

Pole **operace** určuje o jakou operaci jde. Žádost (ARP *request*) má hodnotu 1 a odpověď (ARP *reply*) má hodnotu 2. Toto pole je definováno rovněž pro reverzní překlad (protokol RARP), kdy RARP žádost používá hodnotu 3 a RARP odpověď hodnotu 4.

Pak již následuje linková adresa odesílatele, IP-adresa odesílatele, linková adresa příjemce (v dotazu vyplněna nulami) a IP-adresa příjemce.

Obr. 5.27
Protokol ARP





Obr. 5.28 **Paket protokolu ARP**

Žádost je posílána linkovým oběžníkem a v poli příjemcova linková adresa má vyplněny nuly. Odpověď pak má již vyplněna všechny pole a nemusí být odesílána oběžníkem. Je třeba zdůraznit, že v odpovědi je odesílatelem dotazovaný a příjemce tazatel (došlo k výměně příjemce a odesílatele). Vše je patrné z následujícího příkladu.

```
C:\> ping 194.149.104.126
```

Tento příkaz, než může vyslat první IP-datagram (ICMP-paket echo request), tak musí zjistit pomocí směrovací tabulky zdali je příjemce na LAN nebo za směrovačem (musí zjistit následující hop). Pokud je příjemce za směrovačem, pak hledá linkovou adresu směrovače. Pokud příjemce není za směrovačem, pak přímo hledá linkovou adresu příjemce (náš případ).

Nyní již víme, že příjemce má IP-adresu 194.149.104.126 a je přímo na LAN. Je třeba zjistit jeho linkovou adresu. Operační systém odesílatele vygeneruje ARP žádost:

```
+ FRAME: Base frame properties
  ETHERNET: ETYPE = 0x0806 : Protocol = ARP:  Address Resolution Protocol
    + ETHERNET: Destination address : FFFFFFFFFFFF
    + ETHERNET: Source address : 0020AFFA2589
      ETHERNET: Frame Length : 42 (0x002A)
      ETHERNET: Ethernet Type : 0x0806 (ARP:  Address Resolution Protocol)
```

```
      ETHERNET: Ethernet Data: Number of data bytes remaining = 28 (0x001C)
  ARP_RARP: ARP: Request, Target IP: 194.149.104.126
      ARP_RARP: Hardware Address Space = 1 (0x1)
      ARP_RARP: Protocol Address Space = 2048 (0x800)
      ARP_RARP: Hardware Address Length = 6 (0x6)
      ARP_RARP: Protocol Address Length = 4 (0x4)
      ARP_RARP: Opcode = 1 (0x1)
      ARP_RARP: Sender's Hardware Address = 0020AFFA2589
      ARP_RARP: Sender's Protocol Address = 194.149.104.121
      ARP_RARP: Target's Hardware Address = 000000000000
       ARP_RARP: Target's Protocol Address = 194.149.104.126

00000:  FF FF FF FF FF FF 00 20 AF FA 25 89 08 06 00 01   ....... ..%.....
00010:  08 00 06 04 00 01 00 20 AF FA 25 89 C2 95 68 79   ....... ..%...hy
00020:  00 00 00 00 00 00 C2 95 68 7E                     ........h~
```

Příjemce okamžitě odpoví paketem:

```
+ FRAME: Base frame properties
  ETHERNET: ETYPE = 0x0806 : Protocol = ARP:  Address Resolution Protocol
    + ETHERNET: Destination address : 0020AFFA2589
    + ETHERNET: Source address : 00603E1D9001
      ETHERNET: Frame Length : 60 (0x003C)
      ETHERNET: Ethernet Type : 0x0806 (ARP:  Address Resolution Protocol)
      ETHERNET: Ethernet Data: Number of data bytes remaining = 46 (0x002E)
  ARP_RARP:
      ARP_RARP: Hardware Address Space = 1 (0x1)
      ARP_RARP: Protocol Address Space = 2048 (0x800)
      ARP_RARP: Hardware Address Length = 6 (0x6)
      ARP_RARP: Protocol Address Length = 4 (0x4)
      ARP_RARP: Opcode = 2 (0x2)
      ARP_RARP: Sender's Hardware Address = 00603E1D9001
      ARP_RARP: Sender's Protocol Address = 194.149.104.126
      ARP_RARP: Target's Hardware Address = 0020AFFA2589
      ARP_RARP: Target's Protocol Address = 194.149.104.121
      ARP_RARP: Frame Padding

00000:  00 20 AF FA 25 89 00 60 3E 1D 90 01 08 06 00 01   . ..%..`>.......
00010:  08 00 06 04 00 02 00 60 3E 1D 90 01 C2 95 68 7E   .......`>.....h~
00020:  00 20 AF FA 25 89 C2 95 68 79 00 00 00 00 00 00   . ..%...hy......
00030:  00 00 00 00 00 00 00 00 00 00 00 00               ............
```

Ze kterého si do pracovní paměti (*ARP cache*) systém automaticky doplní položku říkající jaká linková adresa přísluší udané IP-adrese. Při příští komunikaci s počítačem 194.149.104.126 se již ARP dotaz nebude generovat, ale použije se tato položka. ARP cache můžeme vypsat příkazem:

```
D:\> arp -a
Interface: 194.149.104.121
       Internet Address                  Physical ADRESS       Type
       194.149.104.126              00-60-3e-1d-90-01     dynamic
       10.1.1.1                          00-01-11-11-ff-08     static
```

V ARP cache mohou být položky získané ARP dotazem, ty jsou typu dynamic. Do ARP cache můžeme také zapsat položky explicitně příkazem arp. Takové položky jsou typu static. Rovněž je možné položky ARP cache příkazem arp rušit.

Příklad vložení statické položky:

```
D:\> arp -s 10.1.1.1 00-01-11-11-ff-08
```

Příklad zrušení položky

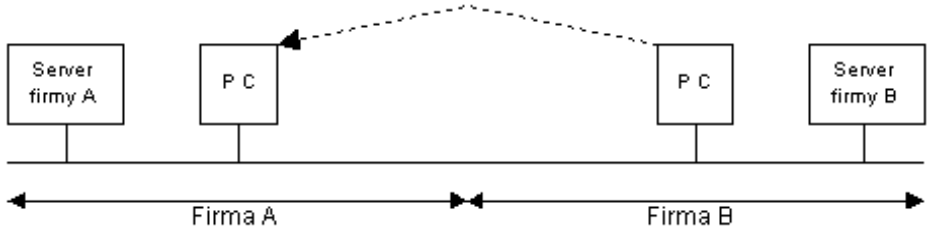
```
D:\> arp -d 10.1.1.1
```

Jak dlouho zůstávají dynamické položky v ARP cache? Tento interval je parametrem jádra operačního systému. Nejčastěji mají položky dobu života 20 minut. Do tabulky se však zaznamenávají i negativní odpovědi, ty mají zpravidla dobu života 3 minuty. Než dojde k negativnímu závěru, tak se ARP žádost opakuje po 5,5 a ještě po dalších 24 sekundách znovu.

5.5.1 Filtrace ARP

ARP filtrace není žádnou filtrací, ale její účinek má podobný efekt jako filtrace. Používá se v případě, že na jedné LAN jsou dvě firmy (resp. dvě samostatné části firmy).

Obr. 5.29
Filtrace ARP



Problém spočívá v zamezení zaměstnanci firmy B v přístupu na server firmy A, tj. aby zaměstnanec firmy B nemohl prohlásit své PC za PC zaměstnance firmy A.

Řešení spočívá v tom, že na serveru firmy A staticky naplníme ARP-cache. Server pak bude odpovídat stále na linkovou adresu PC zaměstnance firmy A aniž by použil ARP-protokol, takže hacker má smůlu.

Význam filtrace ARP je ale omezený, protože zaměstnanec firmy B může podvrhnout i linkovou adresu, to však není tak triviální a začínajícího hackera to odradí.

5.5.2 RARP

Protokolem ARP je také možné odeslat žádost s vyplněnou IP-adresou odesílatele i příjemce a také s oběma vyplněnými linkovými adresami. Takovou žádost je možné chápat jako: „Neexistuje náhodou na LAN ještě jiná stanice, která používá stejnou IP-adresu jako já?". V případě, že se obdrží odpověď, tak se uživateli signalizuje zpráva „*Duplicate IP address sent from Ethernet address xx:xx:xx:xx:xx:xx*". To pochopitelně signalizuje chybu v konfiguraci jedné ze stanic používajících tuto adresu.

Zatímco protokol ARP slouží k překladu IP-adres na linkové adresy reverzní ARP označované jako RARP slouží k překladu linkové adresy na IP-adresu. Avšak proč takový překlad provádět?

Smysl protokolu RARP je u bezdiskových stanic. Bezdisková stanice po svém zapnutí nezná nic jiného než svou linkovou adresu (tu má uloženu výrobcem v paměti ROM). Po svém zapnutí se potřebuje dozvědět svou IP-adresu. Proto do LAN vyšle oběžník s prosbou: „Já mám linkovou adresu HW1, kdo mi řekne jakou mám IP-adresu". Na LAN pak musí být RARP-server, který jí IP-adresu přidělí a sdělí v odpovědi. Protokol RARP používá stejný formát paketu jako protokol ARP. Pouze hodnota pole operace je zvětšena o jedničku. V RARP žádosti pochopitelně není vyplněna ani IP-adresa žadatele.

Protokol RARP se v praxi téměř nepoužívá, nahradil jej protokol DHCP, který je komplexnější.

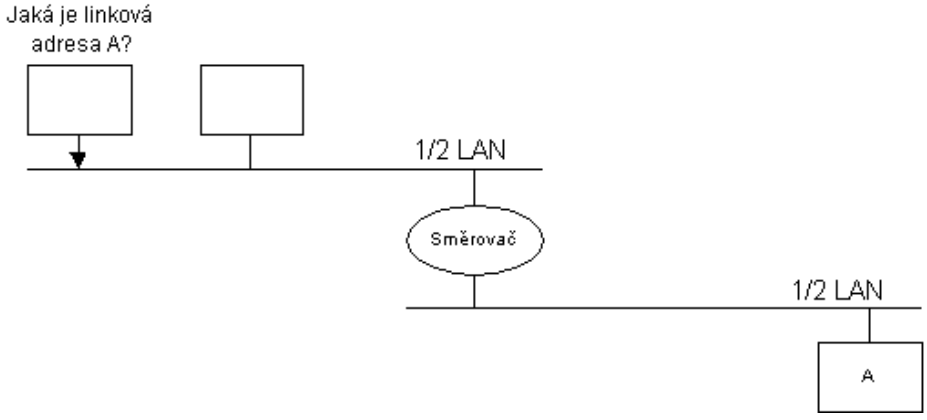
5.5.3 Proxy ARP

Protokol ARP pracuje pouze v rámci LAN, tj. mezi dotazovaným a odpovídajícím počítačem nemůže být směrovač. Důvod je prostý: v dotazu je adresou příjemce všeobecný oběžník, který směrovače nešíří.

Jak si však počít, když mám dvě (nebo více) částí LAN odděleny směrovačem? Řešením je proxy ARP. Proxy ARP běží na směrovači.

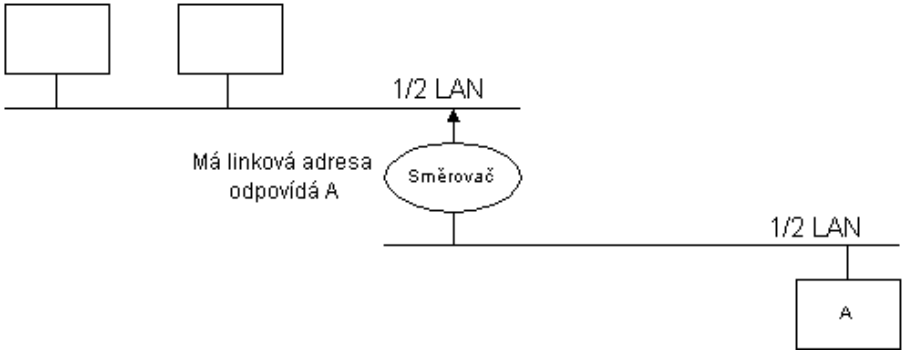
Počítač chce ARP-dotazem zjistit linkovou adresu k IP-adrese A, která leží na druhé polovině LAN za směrovačem.

Obr. 5.30 LAN jejíž část je za jiným rozhraním směrovače



Směrovač nemůže propustit takový dotaz, avšak pokud je nakonfigurován jako proxy ARP, pak odpoví, že IP-adrese A odpovídá linková adresa samotného směrovače.

Obr. 5.31 Proxy ARP



Počítač pak, když chce odeslat linkový rámec A, adresuje směrovač, který IP-datagram předá cílovému počítači A.

5.6 IGMP

Protokol IGMP je podobně jako protokol ICMP služebním protokolem (podmnožinou) protokolu IP. Pakety IGMP-protokolu jsou baleny do IP-datagramů.

Protokol IGMP slouží k šíření adresných oběžníků (*multicasts*). Nyní je aktuální protokol IGMP verze 2 podle normy RFC-2236.

Struktura IGMP-paketu verze 2:

Obr. 5.32 Struktura IGMP paketu verze 2

Pole **typ** nabývá hodnot:

Hodnota (šestnáctkově)	Význam
11	Dotaz směrovače: "Jsou na LAN ještě nějací členové" *(Membership query)*
16	Požadavek na členství ve skupině *(Membership report)*
17	Opuštění skupiny *(Leave group)*
12	Požadavek IGMP v1 na členství ve skupině *(Version 1 membership report)*

Tab. 5.6

Pole **MRT** (*Maximum response time*) se používá pouze v dotazu směrovače a specifikuje v desetinách sekundy čas do kterého musí členové skupiny opakovat požadavky na členství ve skupině. Ve všech ostatních případech má pole MRT hodnotu 0.

Kontrolní součet se počítá stejně jako u protokolu ICMP. Pole IP-adresa adresného oběžníku je nulové u všeobecného dotazu, v ostatních případech specifikuje konkrétní IP-adresu adresného oběžníku.

IP-adresy adresných oběžníků jsou v intervalu 224.0.0.0 až 239.255.255.255. Interval 224.0.0.0 až 224.0.0.255 je určen pro vyhrazené účely na LAN (viz tab. 5.7). Jelikož jsou oběžníky s těmito adresami určeny výhradně pro LAN, tak mívají v položce TTL nastavenu hodnotu 1.

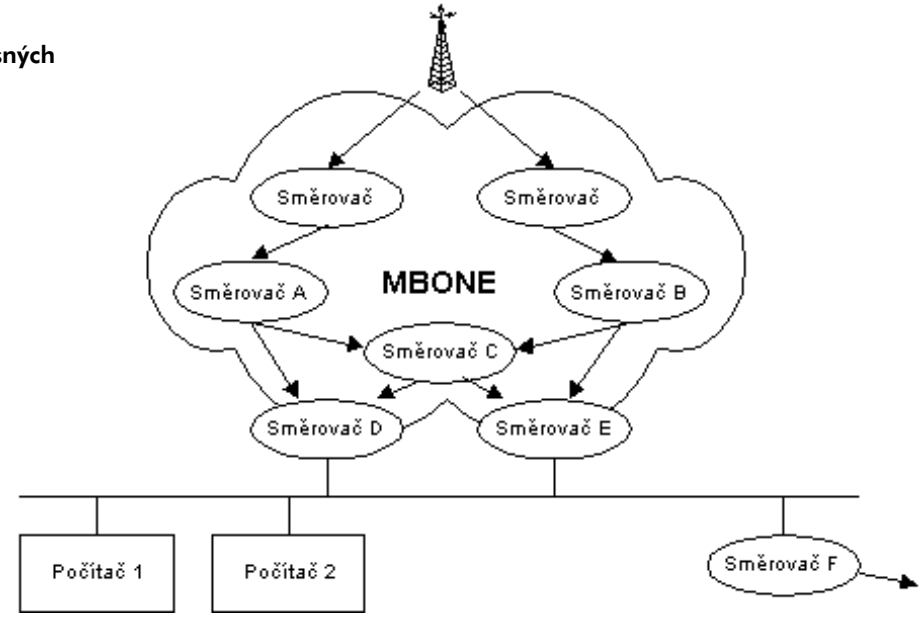
IP-adresa	Vyhrazeno pro adresaci
224.0.0.1	Všechny systémy na LAN
224.0.0.2	Všechny směrovače na LAN
224.0.0.4	*Distance Vector Mulitcast Routing Protocol – viz RFC-1075*
224.0.0.5	*OSPF All Routers – viz RFC-1583*
224.0.0.6	*OSPF Designated Routers – viz RFC -1583*
224.0.0.9	*RIP-2 atd.*

Tab. 5.7

Všechny IGMP pakety mají v IP-záhlaví nastavenu položku TTL=1. Pakety protokolu IGMP verze 2 používají volbu (rozšíření) IP-záhlaví „Upozornění pro směrovač (*IP Router Alert Option)*".

Jádrem Internetu je tzv. Mbone (zkráceno z *Mulicast Backbone*), kde je zabezpečeno šíření adresných oběžníků. Že to není jednoduché, je vidět z obrázku 5.33, kdy zdroj adresných oběžníků – např. internetová rozhlasová stanice šíří svá data pomocí adresných oběžníků. Kdyby se oběžníky šířily nekontrolovaně lavinovitě, tak by se mohla data postupně duplikovat. Např. na směrovač C by ta samá data mohla dorazit ze směrovače A i ze směrovače B.

Obr. 5.33
Šíření adresných oběžníků



Protokol IGMP řeší šíření adresných oběžníků v rámci LAN.

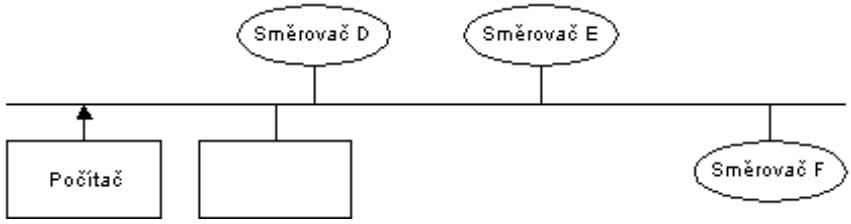
Představme si situaci, kdy jsme na LAN a některé směrovače přijímají adresné oběžníky z MBONE a řeší otázku, zdali je mají šířit dále na LAN. Obecně, pokud žádný počítač na LAN adresné oběžníky nepotřebuje, pak je zbytečné je šířit – pouze by se zvětšilo zatížení LAN.

Jsme tedy v situaci, kdy některé směrovače na LAN mohou LAN zásobovat adresnými oběžníky, ale nečiní tak, protože adresné oběžníky na LAN nejsou vyžadovány.

Pro každou IP-adresu adresného oběžníku se na LAN definuje tzv. skupina členů adresného oběžníku. Směrovače udržují seznam skupin. V případě, že se nějaký počítač na LAN přihlásí do konkrétní skupiny, pak směrovače začnou daný oběžník na LAN šířit.

V případě, že poslední člen skupinu opustí, pak se šíření adresného oběžníku na LAN zastaví. Čili existence skupiny znamená šíření oběžníků. Přitom není důležité, kolik má skupina členů, ale jestli má alespoň jednoho člena nebo nikoliv.

Spustí-li se na počítači aplikace, která chce poslouchat rozhlasovou stanici např. 226.1.1.1, pak počítač vyšle požadavek na členství ve skupině 226.1.1.1 – viz obr. 5.34.



Obr. 5.34
Počítač na LAN odesílá požadavek na členství ve skupině

Vyplnění polí požadavku:

IP-záhlaví:

TTL=1
IP-adresa odesílatele=Počítač
IP-adresa příjemce=224.0.0.2 (všem směrovačům na LAN)

IGMP-paket:

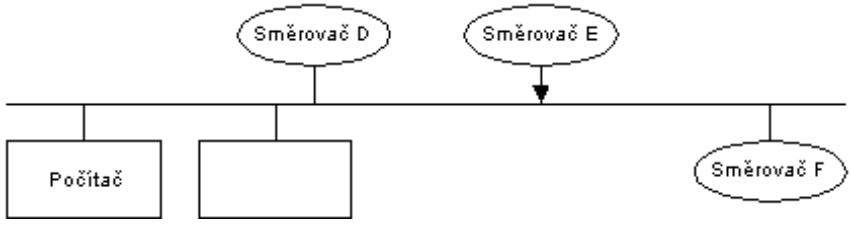
Typ=16_{16}
MRT=0
IP-adresa oběžníku=226.1.1.1

Pokud na LAN dosud nejsou šířeny adresné oběžníky 226.1.1.1, pak se s jejich šířením začne.

Pokud počítač odebírá adresné oběžníky a je posledním členem skupiny, pak může šíření zastavit odesláním obdobného IGMP-paketu, ale s polem Typ=17_{16}.

Jenže co když na počítači není aplikace řádně ukončena, počítač je jednoduše vytažen ze zásuvky, pak nemá šanci vyslat „vypínací paket". Představme si, že adresné oběžníky na LAN šíří směrovač E (viz obr. 5.35). Směrovač E, aby zjistil, zdali je třeba naši skupinu stále ještě šířit, tak čas od času pošle na LAN IGMP-paket „Jsou na LAN ještě nějací členové" (*Membership query*), tj. pole typ=11_{16}. Tento paket má dvě varianty:

1. Všeobecný dotaz (pole IP-adresa oběžníku je vyplněno nulami), který se ptá na všechny skupiny a jednotlivé počítače musí postupně zopakovat své požadavky na členství v každé skupině do MTR desetin sekund. V opačném případě se chápe, že skupinu opustily.
2. Adresný dotaz na konkrétní skupinu (pole IP-adresa oběžníku je v IGMP-paketu vyplněno). Všichni členové uvedené skupiny musí do MTR desetin vteřin zopakovat požadavek na členství.



Obr. 5.35
Směrovač E: „Jsou na LAN ještě nějací členové?"

Kde:

IP-záhlaví:

TTL=1
IP-adresa odesílatele=Směrovač E
IP-adresa příjemce=224.0.0.1 (všem systémům na LAN)

IGMP-paket:

Typ=11_{16}
MRT>0 pro verzi 2, =0 pro verzi 1 (pak je konstantně 10 s)
IP-adresa oběžníku=226.1.1.1 (adresný dotaz), 0.0.0.0 (všeobecný dotaz).

Otázkou zůstává, jak se mezi sebou dohodnou jednotlivé směrovače na LAN v případě, že jich je tam více než jeden. Směrovače vzhledem k protokolu IGMP pracují ve dvou režimech:

1. Dotazovač, který posílá na LAN dotazy na členství ve skupinách.
2. Posluchač, který není aktivní, pouze naslouchá provoz a pokud je na LAN nějaký dotazovač, tak nevstupuje do hry.

Směrovač po svém zapnutí začíná pracovat jako dotazovač, zjistí-li však, že se na LAN vyskytují i dotazy směrovače s vyšší IP-adresou, pak se přepne do režimu posluchač.

5.7 Všeobecné oběžníky a linkový protokol

Zatím jsme popisovali odesílání všeobecných oběžníků, ale problém na LAN spočívá v určení linkové adresy jejich příjemce.

Protokol ARP určil jednoznačný vztah mezi jednoznačnou IP-adresou příjemce (*unicast*) a linkovou adresou příjemce. To je možné tehdy, když mezi IP-adresami a linkovými adresami existuje jednoznačný vztah. Tento vztah se anglicky nazývá *mapping*, který se česky (asi ne zcela správně) označuje za mapování IP-adres na linkové adresy.

Jiným případem na LAN je všeobecný oběžník (*broadcast*), který se posílá všem systémům na LAN. Pro tyto účely slouží linkovému protokoluvšeobecný oběžník, který pro Ethernet, FDDI apod. je ff:ff:ff:ff:ff:ff.

Jenže jak to udělat na LAN v případě adresného oběžníku (*multicast*), který není určen jednou adresátovi na LAN ani všem systémům na LAN, nýbrž několika konkrétním adresátům?

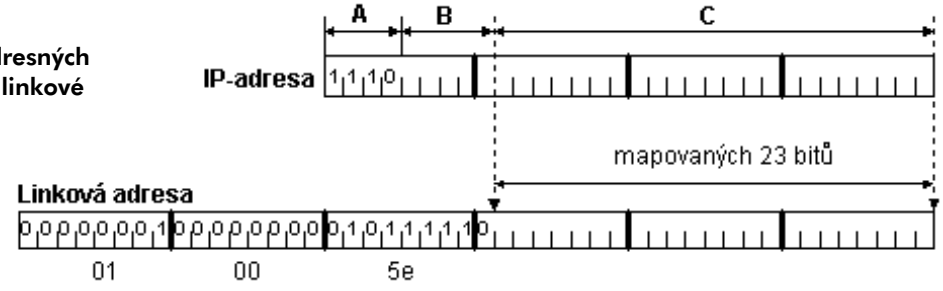
V čem je problém? Příjemce za normálních okolností zpracovává pouze rámce, které jsou všeobecnými oběžníky nebo adresovány příjemcovou linkovou adresou. (Je možné síťovou kartu realizující síťové rozhraní přepnout do tzv. promiskuitního módu, kdy přijímá vše, ale tento případ není považován za běžný.)

Linkové protokoly umožňují též linkové adresné oběžníky (*multicast*). Jsou to takové linkové adresy, kde nejnižší bit prvního bajtu linkové adresy je nastaven na 1, tj. všeobecný linkový oběžník je zvláštním případem takovéhoto adresného oběžníku. Jenže jak mapovat IP-adresu adresného oběžníku na linkový adresný oběžník.

Není to však tak jednoduché jak to vypadá na první pohled. Šestibajtová linková adresa se skládá ze tří bajtů specifikujících výrobce a tří bajtů čísla karty v rámci výrobce.

IANA (nejvyšší autorita Internetu) se nechala zaregistrovat jako fiktivní výrobce síťových karet a obdržela pro sebe identifikaci 00:00:5e. První polovinu těchto adres použila pro mapování adresných IP-oběžníků na adresné linkové oběžníky (viz obr. 5.36). Bohužel tato polovina má pouze 23 bitů, takže mapování nemůže být jednoznačné.

Obr. 5.36
Mapování adresných oběžníků na linkové adresy



První bajt linkové adresy musí mít nastaven nejnižší bit na jedničku, protože se jedná o adresný linkový oběžník. Takže prefix ve skutečnosti nebude 00:00:5e ale 01:00:5e.

Část A IP-adresy specifikuje adresný oběžník, je tedy vždy konstantní. Část B není mapována.

Takže pokud se dva adresné oběžníky liší pouze v části B, pak jsou mapovány na stejnou linkovou adresu. Např. IP-adresy 224.0.1.1, 224.128.1.1 a 225.0.1.1 jsou mapovány vždy na 01:00:5e:00:01:01.

Linková vrstva počítače akceptuje linkové rámce.

- Adresované jednoznačnou linkovou adresou počítače (*unicast*).
- Všeobecné linkové oběžníky (*broadcast*).
- Adresné linkové oběžníky, jejichž seznam je linkové vrstvě předán vyššími vrstvami.

Seznam přijímaných adresných linkových oběžníků zahrnuje adresu 224.0.0.1 a s každým oběžníkem i všechny adresné oběžníky, které vznikly díky nejednoznačnosti mapování. Nadbytečné oběžníky musí odfiltrovat IP-protokol. Některé implementace softwaru přepnou síťovou kartu do promiskuitního módu pro interval všech adresných oběžníků a vše ponechají na IP-protokolu, to však zbytečně zvyšuje zatížení operačního systému.

6

IP-adresa

Protokol IP verze 4 používá IP-adresu o délce čtyři bajty. IP-adresa adresuje jednoznačně síťové rozhraní systému. Anglicky se takováto jednoznačná adresa nazývá *unicast*. Pokud má systém více síťových karet (více síťových rozhraní) a na všech je provozován protokol IP, pak každé rozhraní má svou IP-adresu. Je to podobné jako s adresou domu. Pokud má dům vchod ze dvou ulic, pak má i dům dvě adresy.

Je možná i opačná varianta, kdy na jedné síťové kartě (fyzicky jednom síťovém rozhraní) podporujeme několik IP-adres. První adresa se obvykle nazývá primární a další adresy pak sekundární nebo aliasy. Využití sekundárních IP-adres je běžné např. pro WWW-servery, kdy na jednom počítači běží WWW-servery několika firem a každý se má tvářit jako samostatný WWW-server.

V praxi se však využívání sekundárních IP-adres pro WWW-servery považuje za plýtvání – používají se tzv. virtuální WWW-servery, kdy mnoha WWW-serverům stačí jedna společná IP-adresa. Specifikace serveru se pak provádí na aplikační úrovni v protokolu HTTP (pomocí hlavičky *host*).

Jelikož má většina počítačů jedno síťové rozhraní, tak se přeneseně místo IP-adresa rozhraní říká IP-adresa počítače.

IP-adresa je tvořena čtyřmi bajty. IP-adresa se zapisuje notací, kde jednotlivé bajty se mezi sebou oddělují tečkou. Rozeznáváme:

- Dvojkovou notaci, kde jednotlivé bity každého bajtu se vyjádří jako dvojkové číslo, např.: 10101010.01010101.11111111.11111000
- Desítkovou notaci – čtyři osmiciferná dvojková čísla se převedou do desítkové soustavy, tj. pro náš příklad: 170.85.255.248
- Šestnáctkovou notaci – jednotlivé bajty IP-adresy se vyjádří šestnáctkově (hexadecimálně), tj. náš příklad: aa.55.ff.f8

IP-adresa se skládá ze dvou částí:

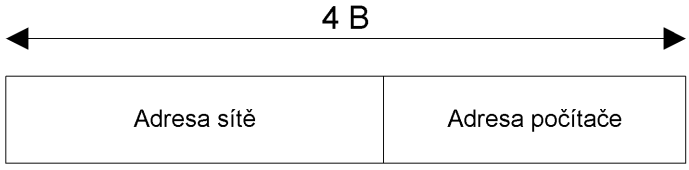
1. Adresy (lokální) sítě.
2. Adresy počítače v (lokální) síti.

Problém je v tom jak zjistit, která část IP-adresy je adresou sítě a která adresou počítače. Není ani zcela jasné co to znamená slovo síť, protože jeho význam se postupně měnil a kromě slova síť se zavedly pojmy subsíť a supersíť. K tomu však musíme dospět postupně.

6.1 Síť – historická epocha I

Tato epocha byla od počátku Internetu až do roku 1993. V této epoše bylo slovo síť specifikováno normou RFC-796 (J.Postel, 1.9.1981). Těchto dvanáct let je poznamenáno představou, že čtyři bajty na IP-adresy přece musí stačit.

Obr. 6.1
Struktura IP-adresy



IP-adresa se dělí na adresu sítě a adresu počítače v rámci této sítě (viz obr. 6.1).

Kolik bajtů z IP-adresy tvoří adresu sítě určují počáteční bity prvního bajtu IP-adresy. IP-adresy se dělí do pěti tříd:

- Třída A, kde nevyšší bit prvního bajtu má hodnotu 0. Zbylých 7 bitů prvního bajtu tvoří adresu sítě a zbytek je určen pro adresu počítače v rámci sítě. V třídě A máme 126 sítí (0 a 127 mají zvláštní význam. V každé síti je 2^{24}-2 adres pro počítače (adresy tvořené samými nulami a samými jedničkami mají zvláštní význam).
- Třída B, kde nejvyšší dva bity prvního bajtu mají hodnotu 10_2. Zbylých 6 bitů a následující druhý bajt je určen pro adresy sítí. Můžeme tedy mít celkem 2^{14} sítí a v každé síti 2^{16}-2 počítačů.
- Třída C, kde nevyšší tři bity prvního bajtu mají hodnotu 110_2. Zbylých 5 bitů a následující dva bajty jsou určeny pro adresu sítě. Můžeme tedy mít 2^{22} sítí a v každé síti 128-2 počítačů.
- Třída D, kde nejvyšší čtyři bity prvního bajtu mají hodnotu 1110_2. Zbytek IP-adresy se pak už nedělí na adresu sítě a adresu počítače. Zbytek IP-adresy tvoří adresný oběžník (*multicast).*
- Třída E tvořící zbytek adres je tč. rezervou.

Tab. 6.1
Třídy IP-adres

Třída	1. bajt IP-adresy	2. bajt IP-adresy	3. bajt IP-adresy	4. bajt IP-adresy
A	0ssssss 1-127$_{10}$	adresa počítače		
B	10ssssss 128-191$_{10}$	ssssssss	adresa počítače	
C	110sssss 192-223$_{10}$	ssssssss	ssssssss	adresa počítače
D	1110mmmm 224-239$_{10}$	mmmmmmmm	mmmmmmmm	mmmmmmmm
E	>239$_{10}$			

Jednotlivé třídy adres jsou shrnuty v tab. 6.1, kde `s` vyznačuje bity používané pro adresu sítě a `m` bity používané pro adresný oběžník.

Z tabulky je dále patrné, že sítí třídy A může být celkem 128 – 2 = 126 a v každé může být 2^{8+8+8}= 16 M adres. Obdobně třídy B může být 14 K a každá může obsahovat až 64 K adres. A konečně sítí třídy C může být 2 M a každá může obsahovat až 256 adres. Některé adresy jsou však vyhrazeny pro speciální účely.

6.1.1 Speciální IP-adresy

IP-adresa je obecně tvaru:

síť.počítač

kde síť je v případě třídy A tvořena jedním bajtem, v případě třídy B tvořena dvěma bajty a v případě třídy C tvořena třemi bajty.

Jsou–li na místě sítě nebo počítače binárně samé nuly (00…0), pak se to vyjadřuje slovem „tento". Jsou-li tam naopak samé jedničky (11…1), pak se to vyjadřuje slovem „všichni" (či oběžník).

Přehled speciálních adres ve dvojkové notaci je uveden v tab. 6.2.

Typ adresy	**Význam**
0.0.0.0	Tento počítač na této síti.
00…0.počítač	Počítač na této síti
síť.00…0	Adresa sítě jako takové
síť.11...1 (samé jedničky na místě adresy počítače)	Všeobecný oběžník *(broadcast)* zasílaný do sítě síť – možno poslat i na vzdálenou síť
11...1 (samé jedničky, tj. desítkově 255.255.255.255)	Všeobecný oběžník na lokální síti *(limited broadcast)* – směrovače jej nepředávají dále
127.cokoliv	Programová smyčka *(loopback)* – nikdy neopouští počítač, zpravidla se používá adresa 127.0.0.1

Tab. 6.2 Speciální IP-adresy

Každé síťové rozhraní (*interface*) má alespoň jednu jednoznačnou adresu (*unicast*), kromě toho celý systém má jednu adresu programové smyčky 127.0.0.1. Adresa 127.0.0.1 není v Internetu jednoznačná, protože ji má každý počítač (*host*).

Příklad: Síť 192.168.6.0 je síť třídy C. Jaké jsou všechny běžící počítače na této síti? Řešení je jednoduché. Všeobecný oběžník (*broadcast*) na této síti má IP-adresu 192.168.6.255. Po vydání příkazu:

```
ping 192.168.6.255
```

všechny běžící počítače na této síti odpoví ICMP-paketem echo. Implementace příkazu ping firmou Microsoft bohužel nezobrazí všechny odpovědi, většina ostatních implementací nám všechny odpovědi zobrazí, takže zjistíme, které počítače na síti běží.

Obdobně lze příkazem ping (s TTL=1) zjistit, které počítače na LAN zpracovávají které adresné oběžníky. Např:

```
ping 224.0.0.1
```

6.1.2 Síťová maska

Síťová maska se používá pro určení adresy sítě. Adresa sítě je částí IP adresy. Síťová maska určuje, které bity v IP-adrese tvoří adresu sítě.

Síťová maska je opět čtyřbajtové číslo. Toto číslo vyjádřené v dvojkové soustavě má v bitech určujících adresu sítě jedničky a v ostatních bitech nuly.

Princip síťové masky se dobře pochopí, používáme-li dvojkovou notaci.

Jednotlivé třídy sítí používají jako adresu sítě různě dlouhou část IP adresy. Třída A používá pro adresu sítě první bajt. Čili standardní síťová maska pro adresy třídy A má v prvním bajtu samé jedničky a ve zbylých třech bajtech samé nuly:

```
11111111.00000000.00000000.00000000
```

což vyjádřeno v desítkové soustavě je:

```
255.0.0.0 (šestnáctkově ff.00.00.00)
```

Obdobně standardní síťová maska pro třídu B je desítkově:

```
255.255.0.0 (šestnáctkově ff.ff.00.00)
```

Konečně pro třídu C:

```
255.255.255.0 (šestnáctkově ff.ff.ff.00).
```

Síťové masky odpovídající třídám A, B a C se nazývají standardní síťové masky.

Síťová maska slouží k řešení úlohy: Jak určit adresu sítě, na které leží počítač o IP adrese:

```
170.85.255.248, tj. dvojkově 10101010.01010101.11111111.11111000
```

Řešení je jednoduché: Nejprve se podíváme do tabulky tříd IP-adres a zjistíme, že naše adresa je třídy B. Používáme standardní síťovou masku, pak maska pro třídu B je:

```
11111111.11111111.00000000.00000000
```

Vynásobíme-li nyní IP-adresu bit po bitu se síťovou maskou, pak získáme adresu sítě:

```
        10101010.01010101.11111111.11111000
x       11111111.11111111.00000000.00000000
        ———————————————————————————————————
        10101010.01010101.00000000.00000000
```

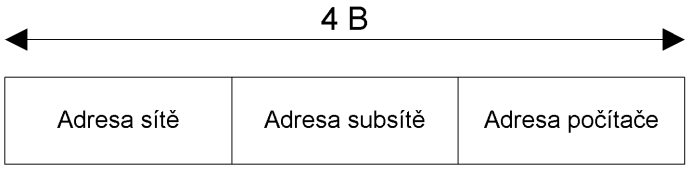
Výsledek převedeme do desítkové soustavy a zjistíme, že počítač leží na síti 170.85.0.0.
Tato metoda určení adresy sítě se může zdát až příliš komplikovanou v případě, že se používají standardní síťové masky. Může se zdát, že síťová maska je důležitá tak pro tvůrce operačního systému, nikoliv však pro správce. Význam síťové masky doceníme v následující historické epoše.

6.2 Síť – historická epocha II

V roce 1993 vyšly normy RFC-1517 až 1520. Tyto dnes již málo citované normy od základu změnily pohled na slovo síť jak je chápáno v Internetu. Přestalo se na sítě hledět přes třídy, ale výhradně přes síťové masky.

Jenže ono úplně nejde abstrahovat od sítí, takže v podstatě dělení IP-adresy na adresu sítě a adresu počítače zůstalo, pouze část IP-adresy dříve odpovídající adrese počítače se rozdělila na dvě části: na adresu subsítě a adresu počítače.

Obr 6.2
Struktura IP-adresy



Z hlediska síťové masky je adresa sítě i subsítě jeden celek. Ta část IP-adresy, kde jsou v masce jedničky je prostě síť. Jenže nyní dochází k nejednoznačnosti v terminologii. Jednou slovo síť označuje ve smyslu třídy (A, B, nebo C) a podruhé je sítí myšleno obecně část IP-adresy, kde v odpovídající masce jsou jedničky. Pokud na čas zapomeneme na třídy a budeme používat libovolné masky, pak už nestačí mluvit o síti např. 192.168.0.0 ale vždy k ní musíme dopsat masku, abychom vyjádřili co touto sítí míníme. Pokud bychom uvažovali třídně, pak se pro tuto síť použije vždy maska 255.255.255.0, protože se jedná o síť třídy C. Maska 255.255.255.0 pro síť 192.168.0.0 se nazývá standardní síťovou maskou.

Kromě subsítí se používají i supersítě, u kterých je počet jedniček masky menší než u standardní síťové masky.

Jako příklad je v tab. 6.3 uvedeno dělení sítě 192.168.0.0 na subsítě s různými maskami (standardní maska je zobrazena tučně).

Adresy s maskami majícími méně jedniček než je standardní maska se nazývají adresy supersítí (v tabulce nahoře) a adresy s maskami o více jedničkách než má standardní maska se nazývají adresy subsítí (dolní část tabulky).

Jelikož dvojkové vyjádření síťové masky je tvořeno zprava souvislou řadou jedniček, tak se místo vyjádření „síť 192.168.0.0 s maskou 255.255.255.252" častěji zkrácuje na 192.168.0.0/30, kde číslo 30 vyjadřuje počet jedniček masky.

Už slyším jak se čtenář čílí, proč je maska tvořena souvislou řadou jedniček. Ano teoreticky to tak být nemusí, je to jen nepsané pravidlo, ale docela dobré pravidlo.

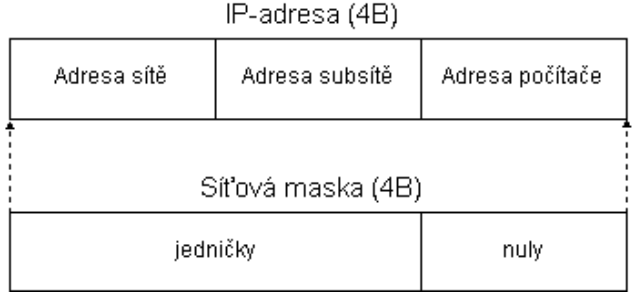
> *Vezměte si např. síť 192.168.0.0 s maskou 255.255.255.95. 95 je binárně 01011111, tj. k dispozici jsou změny na místech x1x11111, takže adresa sítě je 00000000 (desítkově 0), adresy pro počítače jsou 00100000 (desítkově 32) a 10000000 (desítkově 128) a oběžník je 10100000 což je desítkově 160. Těžko řešitelným problémem je pak mezi tyto adresy vložit další subsítě.*
>
> *Myslíte, že byste takovou síť chtěli spravovat? Pointa spočívá v tom, že většina softwaru takové sítě i podporuje.*

Maska	Počet jedniček v masce (zleva)	Síť je tvořena intervalem IP-adres	Zkrácený zápis sítě (včetně masky)
255.248.0.0	13	192.168.0.0 až 192.175.255.255	192.168.0.0/13
255.252.0.0	14	192.168.0.0 až 192.171.255.255	192.168.0.0/14
255.254.0.0	15	192.168.0.0 až 192.169.255.255	192.168.0.0/15
255.255.0.0	16	192.168.0.0 až 192.168.255.255	192.168.0.0/16
255.255.248.0	21	192.168.0.0 až 192.168.7.255	192.168.0.0/21
255.255.252.0	22	192.168.0.0 až 192.168.3.255	192.168.0.0/22
255.255.254.0	23	192.168.0.0 až 192.168.1.255	192.168.0.0/23
255.255.255.0	24	192.168.0.0 až 192.168.0.255	192.168.0.0/24
255.255.255.128	25	192.168.0.0 až 192.168.0.127	192.168.0.0/25
255.255.255.192	26	192.168.0.0 až 192.168.0.63	192.168.0.0/26
255.255.255.224	27	192.168.0.0 až 192.168.0.31	192.168.0.0/27
255.255.255.240	28	192.168.0.0 až 192.168.0.15	192.168.0.0/28
255.255.255.248	29	192.168.0.0 až 192.168.0.7	192.168.0.0/29
255.255.255.252	30	192.168.0.0 až 192.168.0.3	192.168.0.0/30
255.255.255.254	31	192.168.0.0 až 192.168.0.1 Pozor, takováto síť je nesmysl, protože má jen dvě IP-adresy, tedy adresu sítě samotné a adresu oběžníku, nedostávají se už adresy pro počítače na této síti.	192.168.0.0/31
255.255.255.255	32	Adresa samostatného počítače (host address) 192.168.0.0	192.168.0.0/32

Tab. 6.3 Příklad dělení sítě 192.168.0.0 na subsítě

6.2.1 Subsítě

Síťová maska nerozlišuje mezi částí IP-adresy určené pro síť a pro subsíť.



Obr. 6.3 IP-adresa a její síťová maska

Používají se také vyhrazené adresy – viz tab. 6.4.

síť.subsíť.00…0	Adresa subsítě jako takové
síť.00…0.00…0	Adresa sítě
síť.subsíť.11…1	Oběžník na subsíti
síť.11…1.11…1	Pozor, toto je oběžník pro všechny subsítě dané sítě

Tab. 6.4 Speciální adresy

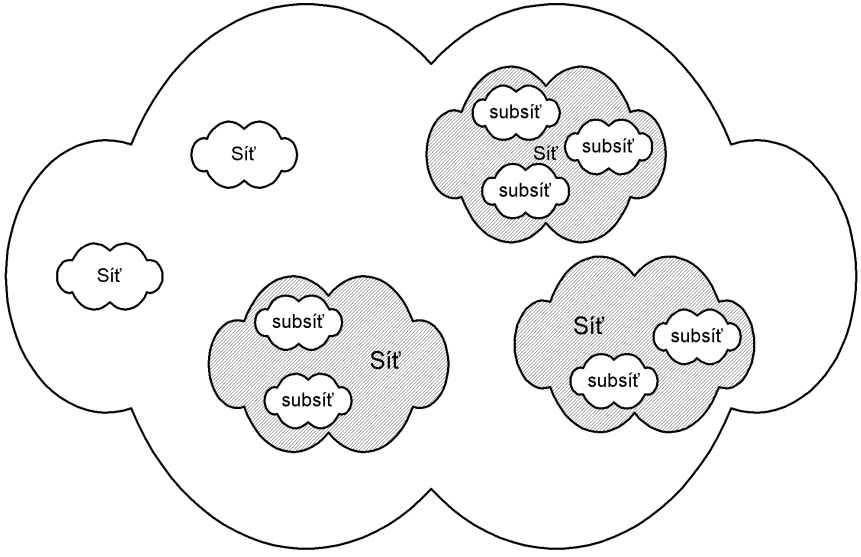
Problém je pochopitelně se subsítí, která má na místě pro subsíť nuly, pak se těžko rozlišuje mezi adresou sítě a subsítě. Rovněž u případu, kdy na místě pro subsíť jsou samé jedničky je nejednoznačnost, zdali oběžník je oběžníkem na subsíť nebo na všechny subsítě. Proto se tyto subsítě snažíme nepoužívat. Mnohý software takové subsítě vůbec nepodporuje, jiný software je v případě použití takovýchto subsítí třeba speciálně konfigurovat.

Avšak oběžník na všechny subsítě dané sítě je stejně jen teorie. Nesetkal jsem se s případem, že by to šlo využít, protože směrovač nemá informaci na jaké subsítě je vzdálená síť dělena.

Subsítě se používají v rámci firem na konfiguraci jednotlivých lokálních sítí. Vzhledem k nedostatku IP-adres má dnes většina firmem přidělenu jen subsíť sítě třídy C. Tuto subsíť si pak dělí na ještě menší subsítě. Dokonce se firmám přidělují jen subsítě sítě třídy C a ty si firmy drobí na menší subsítě.

Subsíť je část Internetu odpovídající jedné firmě nebo části firmy.

Obr. 6.4 Internet je tvořen sítěmi, sítě se mohou dělit na subsítě



Prakticky: Mám za úkol připojit firmu k Internetu. Získal jsem adresu třídy C (např. 194.149.115.0 desítkově, tj. 11000010.10010101.1110011.00000000 dvojkově), která má standardní síťovou masku 255.255.255.0. Měl jsem tedy štěstí, že jsem dostal celou adresu třídy C.

Problém je v tom, že můj podnik má složitější strukturu – jeho síť se skládá z řady menších lokálních sítí (LAN) a řady sériových linek propojujících tyto LAN. Je tedy nutné rozdělit přidělenou síť na subsítě. Navenek se pochopitelně bude celý podnik tvářit jako jedna síť se standardní maskou.

Vzhledem k tomu, že první tři bajty přidělené IP-adresy zůstávají stále stejné, budu v dalších úvahách psát jen posledním bajt v IP-adrese (první tři bajty jsou konstantně např. 194.149.115).

Pro rozdělení sítě na subsítě se na první pohled naskýtá možnost nepoužít standardní síťovou masku pro adresu třídy C, tj. 255.255.255.0, ale např. nestandardní ale **konstantní síťovou masku** 255.255.255.240 (dvojkově 11111111.11111111.11111111.11110000 – všimněte si, že první polovina posledního bajtu slouží pro adresu subsítě), která nám umožní rozdělit přidělenou adresu třídy C na 16 subsítí a každá subsíť může mít 16 adres.

Subsíť dvojkově (poslední bajt z IP-adresy 194.149.115.0)	Síťová maska (dvojkově)	Adresa subsítě (desítkově)	Síťová maska (desítkově)	Max. počet počítačů v subsíti (bez adresy subsítě a oběžníku)
00000000 až 00001111	11110000	.0	.240	0 (nejednoznačná subsíť)
00010000 až 00011111		.16		14
00100000 až 00101111		.32		14
00110000 až 00111111		.48		14
01000000 až 01001111		.64		14
01010000 až 01011111		.80		14
01100000 až 01100000		.96		14
01110000 až 01111111		.112		14
10000000 až 10001111		.128		14
10010000 až 10011111		.144		14
10100000 až 10101111		.160		14
10110000 až 10111111		.176		14
11000000 až 11001111		.192		14
11010000 až 11011111		.208		14
11100000 až 11101111		.224		14
11110000 až 11111111		.240		0 (nejednoznačná subsíť)

Tab. 6.5 Konstantní síťová maska

Každá subsíť má 16 adres, ale použitelných je pouze 14, protože dvě adresy mají speciální význam. Samé nuly označují adresu subsítě a samé jedničky oběžník na subsíti. Např. adresa 194.149.115.32 označuje „třetí“ subsíť jako takovou a adresa 194.149.115.47 oběžník na této subsíti (194.149.115.255 je oběžník na síti 194.149.155.0 jako takové). Přidělovat je tedy možné na subsíti 194.149.115.32 pouze adresy 194.149.155.33 až 46.

Druhým problémem je, že není jednoznačně určeno, zdali 194.149.155.255 je oběžník pro všechny subsítě sítě 194.149.155.0 nebo jen na subsíti 194.149.115.240. Proto se poslední subsíť zpravidla nepoužíí-

vá. Podobným problémem je kolize při určení adresy sítě a subsítě 194.149.115.0. Proto se též první subsíť zpravidla nepoužívá.

V praxi často nepotřebujeme rozdělit přidělenou adresu na stejné části. Např. pro sériové linky je 14 adres na subsíť zbytečný luxus a naopak pro mnohé LAN je to nedostatečné. Pro rozdělení sítě na různě dlouhé subsítě používáme **variabilní síťovou masku**. Např. viz tab. 6.6.

Subsíť dvojkově (poslední bajt)	Síťová maska (dvojkově)	Adresa subsítě (desítkově) 194.149.115.	Síťová maska (desítkově)	Max. počet počítačů v subsíti (bez adresy subsítě a oběžníku)
00000000 až 00000011	11111100	.0	.252	0 (nejednoznačná subsíť)
00000100 až 00000111	11111100	.4/30	.252	2
00001000 až 00001111	11111000	.8/29	.248	6
00010000 až 00011111	11110000	.16/28	.240	14
00100000 až 00111111	11100000	.32/27	.224	30
01000000 až 01111111	11000000	.64/26	.192	62
10000000 až 10111111	11000000	.128/26	.192	62
11000000 až 11011111	11100000	.192/27	.224	30
11100000 až 11101111	11110000	.224/28	.240	14
11110000 až 11100011	11111000	.240/29	.248	6
11111000 až 11111011	11111100	.248/30	.252	2
11111100 až 11111111	11111100	.252/30	.252	0 (nejednoznačná subsíť)

Tab. 6.6 Variabilní síťová maska

Z předchozí tabulky je vidět, že nejdelší subsíť má 64 prvků, potřebujeme-li na jednu LAN více jak 62 rozhraní, pak je dobré použít celou adresu třídy C.

Nyní si můžeme dát nový příklad. Určete adresu sítě, na které leží počítač o IP-adrese 10.0.0.239 používáme-li síťovou masku 255.255.255.240.

IP-adresu i masku převedeme do dvojkové soustavy a bit po bitu vynásobíme:

```
        00001010.00000000.00000000.11101111 tj. 10.0.0.239
x       11111111.11111111.11111111.11110000 tj. 255.255.255.240
        ———————————————————————————————————-
        00001010.00000000.00000000.11100000 = 10.0.0.224
```

Adresa leží na síti 10.0.0.224. Ale může to být adresa počítače? Ne. Proč? Oddělíme adresu sítě a adresu počítače:

```
   00001010.00000000.0000000.1110|1111
             <————————-síť————————>|<poč>
```

adresa je tvaru síť.jedničky, nejedná se tedy o adresu počítače, ale o adresu oběžníku na síti 10.0.0.224.

Podobně jako u sítí je možné poslat příkazem ping všeobecný oběžník, tak i v našem případě:

```
ping 10.0.0.224
```

zjistí zdali na subsíti je nějaký počítač „živý" a UNIXové implementace příkazu ping nám sdělí které počítače na subsíti jsou „živé".

6.2.2 Supersítě, autonomní systémy

Zatímco subsítě se používají na LAN, tj. používají se v konfiguraci jednotlivých síťových rozhraní (síťových karet), tak supersítě se používají pro agregace IP-adres. Agregace IP-adres je výhodná pro směrování a pro administrativu při přidělování IP-adres.

V současné době je při pohledu z velké vzdálenosti (z Měsíce) Internet soustavou vzájemně propojených poskytovatelů Internetu. Poskytovatel Internetu (*provider*) poskytuje připojení k Internetu buď pro komerční nebo nekomerční účely. Kromě poskytovatelů tvoří Internet ještě několik organizací, které se zabývají správou a vývojem v této oblasti, avšak ze síťového hlediska se od poskytovatelů neliší.

Poskytovatelé dopravují IP-datagramy buď v rámci sebe nebo mezi sebou. Dva poskytovatelé si mohou vyměňovat IP-datagramy mezi sebou, existují však i tranzitní poskytovatelé, kteří přes sebe IP-datagramy tranzitují.

Neříká se, že se Internet dělí na poskytovatele, ale z hlediska dopravy IP-datagramů se Internet dělí na autonomní systémy. Každý poskytovatel má pak přidělen jeden nebo více autonomních systémů. Autonomní systém je reprezentován dvoubajtovým číslem.

Internet je tedy z hlediska směrování (tj. dopravy IP-paketů) rozdělen na autonomní systémy (AS). Pro směrování mezi AS se dříve používal protokol EGP, nyní se však masově přechází k protokolu BGP verze 4.

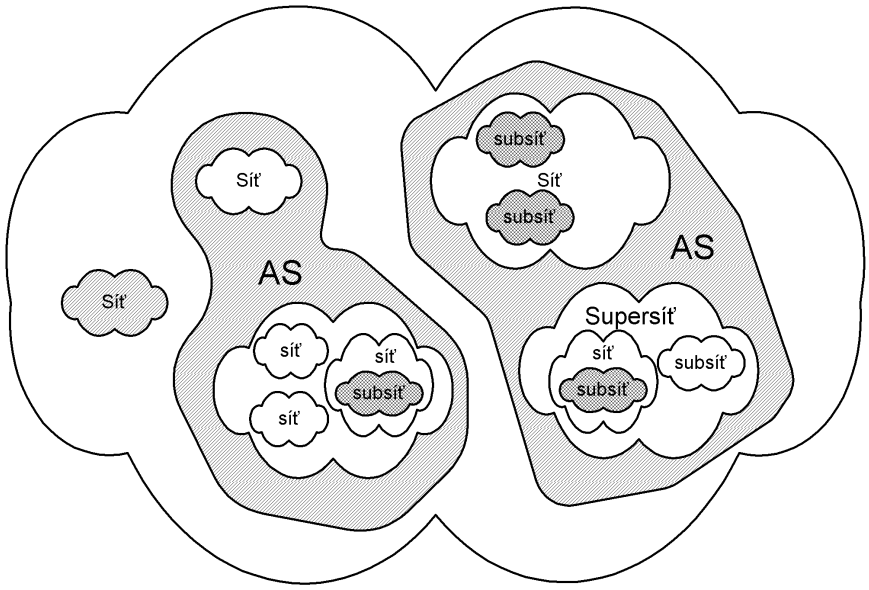
Poskytovatelé Internetu jsou správci autonomního systému. Správci AS žádají o intervaly IP-adres, které přidělují sobě a svým zákazníkům. Jednotliví poskytovatelé jsou pak i se svými zákazníky součástí konkrétního AS. V kapitole o administrativní stránce věci se dozvíme, že problém je ještě složitější, že o přidělení IP-adresy žádají tzv. lokální Internet Registry, kteří mohou mít svůj vlastní AS nebo mohou sdílet AS s někým dalším.

Proč je snaha v rámci autonomního systému používat intervaly adres? Důvod je prostý. Interval adres je možné agregovat do jedné adresy supersítě. Ve směrových tabulkách směrovačů vzdálených autonomních systémů může celý interval adres vystupovat jako jedna položka, čímž se šetří paměť směrovače a zjednodušuje se správa.

Agregace je jednoduchá. Např. je-li přidělen interval adres sítí 194.149.96.0 až 194.149.128.0, pak je možné jej agregovat na adresu supersítě 194.149.96.0 se síťovou maskou 255.255.224.0. Častěji se však píše adresa 194.149.96.0/19 (maska 255.255.224.0 je tvořena 19 jedničkami).

Zatímco při dělení sítě na subsítě obsahovala maska více jedniček, tak při agregaci je tomu naopak, ale princip je týž. O agregovaných sítích se hovoří jako o supersítích. Z pohledu vzdáleného AS se supersíť jeví jako jeden celek. Pro správce AS je síť jeden celek. A pro správce sítě jsou zase jednotlivými celky subsítě.

Problém ale spočívá v tom, když firma přejde od jednoho poskytovatele Internetu k jinému, který je navíc v jiném autonomním systému. Pak musí od nového poskytovatele získat nové IP-adresy a následně všechny používané sítě přečíslovat. Jména počítačů (tj. i e-mailová adresa) však mohou zůstat zachovány.



Obr. 6.5
Internet se dělí na autonomní systémy, autonomní systémy se mohou dělit na supersítě, supersítě na sítě a sítě se mohou dělit na subsítě

Existují i adresy na poskytovateli nezávislé (*provider independent*), které jsou poskytovány pro firmy připojené k více poskytovatelům současně nebo LAN tvořící jádro NIX (*Neutral Internet eXchange*), kde si jednotliví poskytovatele kolektivně mezi sebou předávají IP-datagramy.

Vraťme se k jednomu z předešlých hypotetických příkladů, kdy firmě byl přidělen interval IP-adres 194.149.115.32/24. Firma byla situována do několika lokalit, a tak správce sítí přidělil lokalitě v Českých Budějovicích rozsah 194.149.115.32/28. V této lokalitě je počítač o IP-adrese 194.149.115.33. Tento počítač chce komunikovat se serverem na Fidži (stáhnout Web-stránku). IP-datagramy jsou dopravovány Internetem na Fidži. Nyní nám chce fidžijský server odpovědět. Zformuluje odpověď a zkoumá adresu 194.149.115.33 kudy odpovědět (do jakého podmořského kabelu odpověď strčit).

Adresa 194.149.115.33 patří do intervalu 194.0.0.0/7 přidělených pro RIPE (organizace přidělující IP-adresy v Evropě a okolních teritoriích). Z hlediska serveru na Fidži je adresát někde mezi Marokem, norským Svalbardem a Alma Atou. Na Fidži by se tedy teoreticky ani nemuseli zabývat otázkou do jakého autonomního systému adresa 194.149.115.33 patří v případě, že směrem do Evropy strkají všechno do jednoho kabelu. Teoreticky by jim stačilo mít ve směrovací tabulce pro celou Evropu a přilehlá teritoria jen jednu položku:

194.0.0.0 s maskou 254.0.0.0.

Z Fidži je to do Evropy geograficky přibližně stejně daleko směrem ze západu, na východ, na jih i na sever, avšak náš IP-datagram tč. pošlou do USA. V Americe už to není tak jednoduché, z Ameriky do Evropy vede řada spojů, je tedy nutné nejprve zjistit směrem ke kterému spoji to v Americe dopravit a pak do kterého spoje to v Americe strčit (během dopravy může dojít i ke změně názoru na to, do jakého spoje do Evropy náš IP-datagram strčit). Američané zjistí, že adresa 194.149.115.33 patří do intervalu 194.149.92/19 patřícího k autonomnímu systému AS5490. Ve směrovací tabulce svých směrovačů již pro každý interval adres, který má přidělen autonomní systém AS5490 musí mít jednu položku. V našem případě:

194.149.92.0 s maskou 255.255.224.0.

Důležité směrovače ležící v USA na hranici autonomních systémů musí tedy mít pro každý interval IP adres přidělený nějakému poskytovateli kdekoliv na Zemi jednu položku. Říkáme, že takový směrovač má úplné směrovací tabulky Internetu. Tč. je třeba mít pro takové tabulky směrovač alespoň se 128 MB operační paměti. Takovéto směrovače s úplnými směrovacími tabulkami jsou nutné jako hraniční směrovače tzv. tranzitních autonomních systémů, tj. autonomních systémů přes které se IP-datagramy dopravují do dalších autonomních systémů. Jestliže náš IP-datagram se v USA objevil na západním pobřeží, tak pravděpodobně bude muset být dopraven skrze tranzitní autonomní systémy USA na východní pobřeží, odkud pravděpodobně povedou podmořské kabely do Evropy.

Předávání IP-datagramů mezi autonomními systémy nezávisí pouze na technických faktorech, tj. technicky nejvýhodnějším spojení směrem k adresátovi, ale také na směrovací politice (*routing policy*) mezi jednotlivými autonomními systémy – lidově řečeno jestli druhá strana platí nebo ne. V případě výskytu nějakých překážek může být náš IP-datagram dopravován komplikovanější cestou, nebo směrování může být i administrativně zakázáno.

Náš IP-datagram dorazil z Ameriky na hraniční směrovač autonomního systému AS5490. Nyní již musí směrovač zkoumat IP-adresu podrobněji a zjistí, že IP-adresa 194.149.115.33 patří do intervalu 194.149.115/24 přiděného naší firmě.

Ve směrovacích tabulkách směrovačů uvnitř autonomního systému AS5490 je položka:

194.149.115.0 s maskou 255.255.255.0.

IP-datagram poskytovatel dopraví na hraniční směrovač naší firmy. Směrovač naší firmy zkoumá kam má IP-datagram dopravit v rámci naší firmy, proto zkoumá adresu 194.149.115.33 a zjistí, že má položku směrovací tabulky (LAN v Českých Budějovicích):

194.149.115.32 s maskou 255.255.255.224

Náš IP-datagram je dopraven do Českých Budějovic na směrovač. Ten zjistí, že síť 194.149.115.32/28 je síť přímo připojená na jeho lokální rozhraní. Protokolem ARP zjistí šestibajtovou linkovou adresu příjemce (pokud ji nemá v ARP-cache) a příjemci dopraví náš IP-datagram. Příjemce zahodí IP-záhlaví a z TCP-záhlaví zjistí, že informace je určena pro Web-prohlížeč, zahodí se TCP-záhlaví a obsah v protokolu HTTP se zobrazí (interpretuje) na obrazovce. Právě jsem to vyzkoušel a tč. celý tento proces mezi serverem v Suvu (Fidži) a klientem v Českých Budějovicích trvá cca 2 vteřiny. To nevypovídá jen o rychlosti a propustnosti přenosových linek, ale i o ohromném výkonu hraničních směrovačů, které musí ohromnou rychlostí vyhledávat ve směrovacích tabulkách, proto často bývají vybaveny specializovanými procesory, které se zabývají obsluhou směrovacích tabulek.

Čísla autonomních systémů přidělují mezinárodní regionální agentury pro přidělování IP-adres (*Internet Registry*). V Evropě je takovou agenturou RIPE. Tyto agentury udržují ve svých databázích informace o intervalech přidělených IP-adresách i o přidělených číslech autonomních systémech.

RIPE na svém serveru ftp://ftp.ripe.net nabízí skript *prtraceroute*, který v sobě volá příkaz *traceroute*, avšak z výstupu příkazu *traceroute* dohledává informace v databázích regionálních agentur příkazem *whois*. Takže, např. cesta na Fidži z hlediska autonomních systémů vypadá:

```
./prtraceroute kula.usp.ac.fj
** WARNING ** Destination AS unknown for kula.usp.ac.fj (144.120.8.11)
** WARNING ** Policy information is not possible - setting to „?“
traceroute to kula.usp.ac.fj (144.120.8.11) with AS and policy additions
```

```
 1 AS5490  cbuN002e00.pvt.net             194.149.104.193    [?]
 2 AS5490  phucbu.pvt.net                 194.149.96.13      [?]
 3  AS701  951.Hssi5-0.GW1.NYC2.ALTER.NET 157.130.0.117      [?]
 4  AS702  143.ATM2-0.XR2.EWR1.ALTER.NET  146.188.177.62     [?]
 5  AS702  192.ATM2-0-0.BR1.EWR1.ALTER.NE 146.188.176.53     [?]
 6  AS701  sl-pen-11-h3.sprintlink.net    137.39.44.130      [?]
 7 AS1790  sl-bb10-pen-0-1.sprintlink.net 144.232.5.5        [?]
 8 AS1790  sl-bb22-stk-6-0.sprintlink.net 144.232.8.178      [?]
 9 AS1790  sl-bb23-stk-8-0.sprintlink.net 144.232.4.110      [?]
10 AS1790  sl-bb10-sj-6-0.sprintlink.net  144.232.8.193      [?]
11 AS1790  sl-gw2-sj-0-0-0-155M.sprintlin 144.232.3.38       [?]
12 AS1239  sl-cais-1.sprintlink.net       144.228.111.18     [?]
13 AS4637  hssi9-0-0.hk-T3.hkt.net        202.84.128.253     [?]
14 AS3491  f5-1.yck06.hkt.net             205.252.130.233    [?]
15 AS3491  a6-0.tmh08.hkt.net             205.252.130.81     [?]
16 AS3491  s4-3b.tmh08.hkt.net            205.252.128.158    [?]
17 AS4637  202.84.251.6                   202.84.251.6       [?]
18    ???  202.62.120.6                   202.62.120.6       [?]
19  AS681  202.62.125.134                 202.62.125.134     [?]
20    ???  kula.usp.ac.fj                 144.120.8.11       [?]

AS Path followed:  AS5490 AS701 AS702 AS701 AS1790 AS1239 AS4637 AS3491 AS4637 ??? AS681
???
AS5490 = PVTNET
 AS701 = Alternet
 AS702 = UUNET
AS1790 = SprintLink Washington D.C.
AS1239 = SprintLink Backbone
AS4637 = Hong Kong Telecom
AS3491 = CAIS Internet
 AS681 = KAWAIHIKO-1
```

První sloupec nám uvádí číslo hopu, druhý sloupec číslo autonomního systému (v desítkové soustavě předcházené řetězcem AS), třetí sloupec uvádí název rozhraní na směrovači, čtvrtý IP-adresu rozhraní směrovače a pátý sloupec obsahuje v hranatých závorkách typ směrování (směrovací politiku). Typ směrování může být např. I – směrování uvnitř autonomního systému nebo E – externí směrování.

Prostřední samostatný řádek vypisuje cestu, kde se hopem nerozumí směrovač, ale celý autonomní systém. V dolní části je uveden název organizace (firmy), které autonomní systém náleží.

Jelikož o síti na Fidži se nenašel záznam v příslušné databázi, tak místo čísla autonomního systému jsou otazníky a směrovací politika není vůbec vyplněna.

Z příkladu je patrné, že spojení z Fidži je komplikovanější než se nám v daleké Evropě jeví – všimněte si, že IP-datagramy putovaly přes Hongkong.

V Evropě se velice dbá na správnou údržbu evropských databází, tak se při cestě Evropou jen zřídka setkáváme s otazníky (s parametrem –v se vypisují i časy okružní procházky):

```
./prtraceroute -v  is.eunet.cz
traceroute with AS and policy additions [Oct 16 05:06:18 UTC]
                from AS5490 info.pvt.net (194.149.104.203)
                to   AS2819 is.eunet.cz (193.85.1.11)
```

```
 1 AS5490  cbuN002e00.pvt.net            194.149.104.193    [I] 3 1 1 ms
 2 AS5490  phucbu.pvt.net                194.149.96.13      [I] 132 14 17 ms
 3 AS5490  194.149.101.226               194.149.101.226    [I] 16 10 20 ms
 4 AS2819  acc-gw.eunet.cz               193.85.3.65        [E1] 35 16 18 ms
 5 AS2819  is.eunet.cz                   193.85.1.11        [I] 15 11 13 ms
AS Path followed:  AS5490 AS2819
AS5490 = PVTNET
AS2819 = EUnet Czechia AS
```

Příkaz prtraceroute je velice užitečný pro správce autonomních systémů, protože není-li někam spojení, pak příkazem prtraceroute zjistí do jakého autonomního systému je až spojení. Příkazem

$ whois AS4637

```
aut-num: AS4637
as-name: HKT-NET-BORDER-AS
descr: Hong Kong Telecom NET Border AS
admin-c: WW3-HK
tech-c: WW3-HK
notify: dbmon@apnic.net
mnt-by: MAINT-HKT
changed: davidc@apnic.net 951030
source: APNIC

person: William Wong
address: 29/F Hongkong Telecom Tower, Taikoo Place
address: 979 King's Road, Quarry Bay, Hong Kong
phone: +852-28837866
fax-no: +852-29625005
e-mail: wwong@hkt.net
nic-hdl: WW3-HK
mnt-by: MAINT-HKT
changed: noc@hkt.net 19981012
source: APNIC
```

zjistí kontakt na správce autonomního systému, ze kterého není dále spojení a požádá jej o pomoc.

6.3 IP-adresy v intranetu

Použití technologie Internetu uvnitř uzavřené firemní sítě se nejprve označovalo internet (s malým i), později se objevilo slovo intranet, které se uchytilo (zlí jazykové tvrdí, že za to mohou Němci, protože v němčině byl internet s malým počátečním i nepřekonatelným trnem v oku němčinářům).

IP-adresy musí být v Internetu přidělovány celosvětově jednoznačně. Ještě před časem mnohé podniky budovaly svou uzavřenou podnikovou síť na bázi protokolu TCP/IP a ani ve snu je nenapadlo, že by se někdy připojovaly k Internetu. I zvolili si naprosto libovolné adresy vlastních sítí. Dnes chtějí tyto sítě propojit přes firewall do Internetu a zjišťují, že stejné adresy už v Internetu někdo používá. Jsou nuceni své sítě přečíslovat, což je velice nepříjemná operace.

Většinou firmy používající v intranetu adresy, které kolidují s adresami v Ineternetu z počátku hledají nějaká netradiční řešení jak se vyhnout přečíslování intranetu. Takovým řešením je např. NAT (*Network*

Address Translator), avšak tato řešení přinášejí jiná negativa, proto po zbytečně vynaloženém úsilí firmy stejně nakonec přistoupí k přeadresování celého intranetu.

Pro uzavřené podnikové sítě si zvolte IP-adresy sítí podle RFC1918 uvedené v tab.6.7.

Tab. 6.7

Třída A	10.0.0.0/8	10.0.0.0 až 10.255.255.255
Třída B	172.16.0.0/12	172.16.0.0 až 172.31.255.255
Třída C	192.168.0.0/16	192.168.0.0 až 192.168.255.255

Použití těchto adres navíc zvyšuje bezpečnost, protože v Internetu jsou nepoužitelné (stovky podniků je používají na svých uzavřených sítích). O přidělení adres v těchto rozsazích není třeba nikoho žádat.

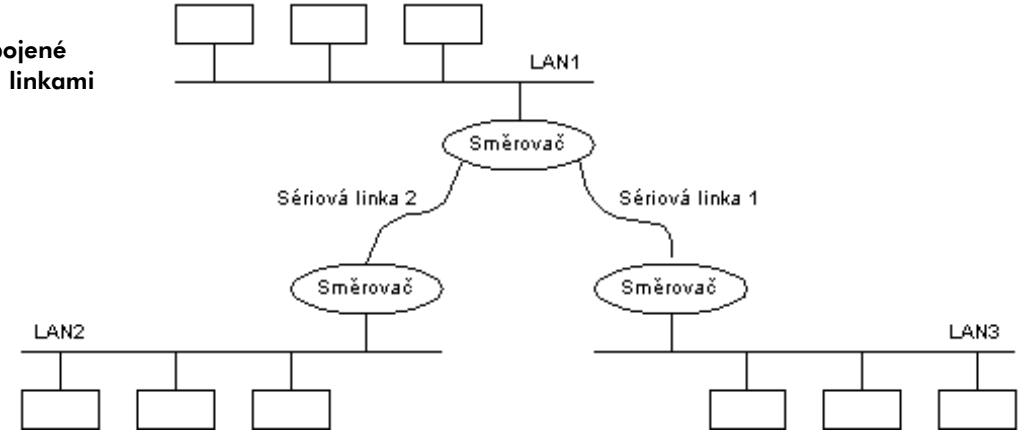
Častou otázkou je jak to poskytovatelé Internetu dělají, že tyto adresy nelze použít, oni je nějak filtrují? Filtrace není třeba, oni je prostě jen nemají ve směrovacích tabulkách, takže je nemohou dopravovat.

Pro intranety jsou vyhrazena i čísla autonomních systémů 64512 až 65535 (viz RFC-1930).

6.4 Nečíslované sítě

Zamysleme se nyní nad sériovými linkami spojujícími LAN. Pro každou linku potřebujeme subsíť o minimálně čtyřech IP-adresách (adresa sítě, oběžník na síti a dvě adresy pro síťová rozhraní na směrovačích).

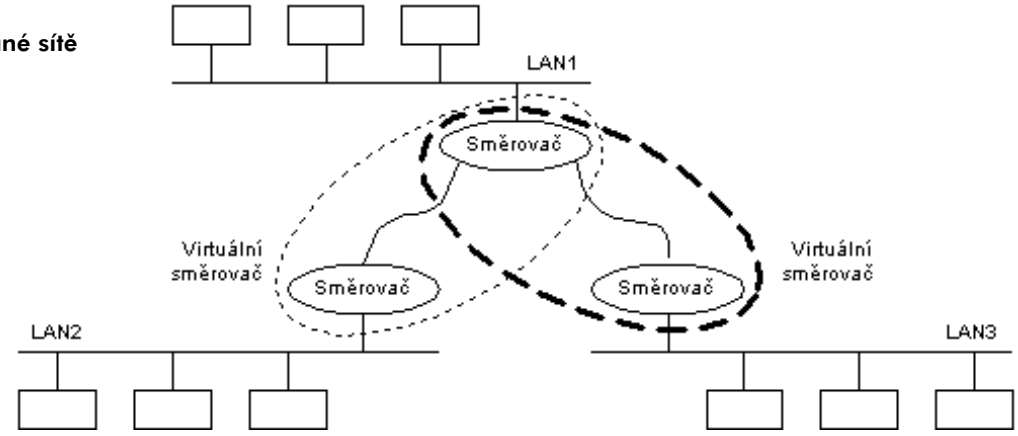
Obr. 6.6
LAN propojené sériovými linkami



Z obrázku 6.6 je patrné, že kromě tří intervalů IP-adres pro lokální sítě budeme potřebovat další adresy pro sítě tvořené sériovými linkami. Na první pohled je vidět, že by bylo efektivní pro sériové linky nepotřebovat další adresu sítě.

Současné směrovače umí na sériových linkách vytvořit „nečíslovanou" síť (*unnumbered interface*), tj. protější směrovače se chovají jako jeden virtuální směrovač. Každý fyzický směrovač pak tvoří polovinu virtuálního směrovače. Virtuální směrovač má pouze dvě rozhraní – jedno pro každou LAN.

Obr. 6.7
Nečíslované sítě



Pro sériové linky tak není třeba plýtvat IP-adresami.

Dynamicky přidělované adresy

Má-li síť již interval IP-adres přidělen, pak můžeme začít s přidělováním adres jednotlivým síťovým rozhraním na této síti. Jsou dvě možnosti:

- Staticky (trvale) přidělit IP-adresu.
- Dynamicky (na dobu připojení) přidělit IP-adresu.

Dynamické přidělování přináší výhodu i v tom, že je potřeba jen tolik IP-adres, kolik je současně přihlášených uživatelů.

Dynamické přidělování adres řeší aplikační protokol DHCP. Protokol DHCP vychází ze zkušeností a částečně v sobě zahrnuje i podporu starších protokolů z této oblasti, tj. protokolů RARP, DRARP a BOOTP. Blíže viz RFC-1531.

V protokolu DHCP žádá klient DHCP-server o přidělení IP-adresy (případně o další služby). DHCP-server může být realizován jako proces na počítači s operačním systémem UNIX, Windows NT atp. Nebo DHCP-server může být realizován i jako součást směrovače.

Zatímco přidělování IP-adres na LAN je v současné době doménou protokolu DHCP, pro přidělování IP-adres počítačům za komutovanou linkou (např. zákazníkům poskytovatele Internetu) se zpravidla přidělují IP-adresy pomocí protokolu PPP.

Protokol PPP je linkovým protokolem používaným na asynchronních sériových linkách. Neumožňuje takové služby jako protokol DHCP, avšak přidělit IP-adresu stanice umí. Více stejně pro připojení uživatele k Internetu nebývá třeba.

Dynamické přidělování IP-adres může být ještě kombinováno s nečíslovanými sítěmi. V případě, že budou sériové linky jako nečíslované sítě, pak směrovač dynamicky přidělující IP-adresy zařídí, že počítače se budou jevit, jakoby byly přímo na lokální síti.

Počítačům připojeným přes komutovanou linku na směrovač je možné přidělit adresy z rozsahu LAN nebo libovolné jiné adresy. V praxi je však výhodné přidělovat adresy buď z rozsahu LAN nebo v in-

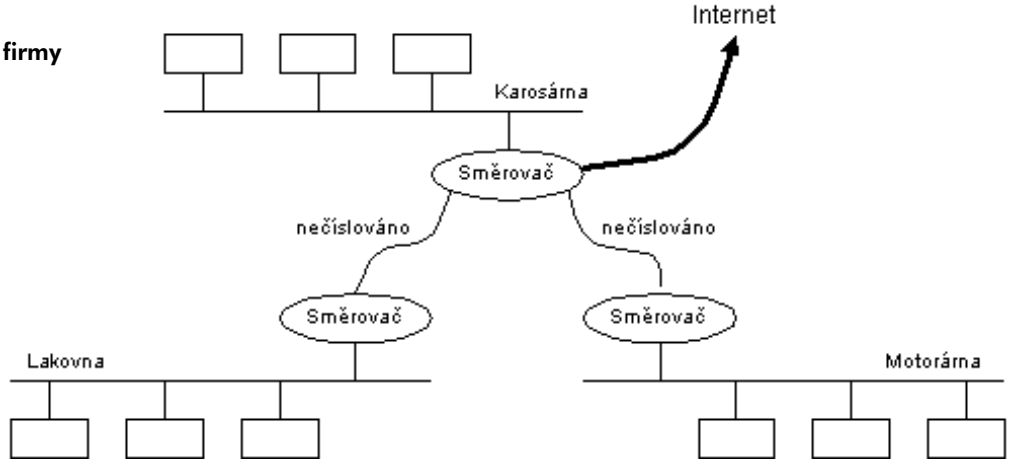
tervalu, který bezprostředně na rozsah LAN navazuje. Z hlediska vzdálenějších lokalit, pak se počítače na LAN i na sériových linkách tváří jako jeden celek – jedna supersíť.

6.5 Adresní plán

Každá firma, která se chce připojit k Internetu si musí nejprve udělat adresní plán. Ten se obvykle skládá ze dvou částí. Jednak ze schematického znázornění propojení jednotlivých LAN do WAN a jednak ze seznamu jednotlivých LAN s odhadovaným počtem síťových rozhraní na LAN. Adresní plán by měl obsahovat rezervu s výhledem na příští a přespříští rok. Jako rezerva se běžně bere dvojnásobek současného stavu. Adresní plán se pak zasílá jako požadavek poskytovateli Internetu, kterého tím žádáme o příslušný počet IP-adres.

Příklad: Máme připojit k Internetu firmu používající 3 lokální sítě: karosárna, lakovna a motorárna (nikdy se poskytovatel nespokojí s žádostí typu: tři sítě A, B a C – vždy se musí jednat o konkrétní požadavek).

Obr. 6.8
Síť fiktivní firmy



V karosárně máme 8 počítačů s výhledem na 16, v motorárně 9 počítačů s výhledem na 18 a v lakovně je 20 počítačů s výhledem na 40 počítačů.

LAN	Současný stav	Příští rok	Za 2 roky	Nejbližší možnost pro LAN	Požadováno
Karosárna	8	10	16	32 (16 nelze, protože je třeba adresa pro subsíť a oběžník)	32
Lakovna	9	15	18	32	32
Motorárna	20	35	40	64	64

Tab. 6.7

Požadujeme na poskytovateli přidělit 128 IP-adres pro tři subsítě. V případě, že by těchto 128 adres mělo tvořit jeden celek – „supersíť", pak nemůžeme požadovat supersíť o 128 adresách, protože jedna LAN by využívala nejednoznačnou subsíť sítě C, v takovém případě je třeba žádat celou síť třídy C, tj. 256 IP-adres. To aby všechny LAN z hlediska poskytovatele tvořily jeden celek ("supersíť"), je vyžadováno zejména v případě, kdy firma využívá pro připojení k Internetu komutovaný spoj. Přitom komutovaným spojem může být zálohována i pevná linka (*dialup backup*).

Příklad neřešil problém sériové linky propojující firmu s Internetem. To je třeba projednat vždy s poskytovatelem.

Možná, že s vám zdá, proč připojovat jednotlivé provozy do Interentu. Větší a velké firmy se vyznačují tím, že nepotřebují více jak 16 IP-adres. Většinou si vyberou z některého ze zapojení firewallu znázorněného na obr. 6.9:

Demilitarizovaná zóna je LAN, která je přístupná z Internetu, proto musí mít i oficiální IP-adresy. Demilitarizovaná zóna má tu výsadu, že je to jediná síť v Internetu, která je alespoň částečně dostupná z intranetu.

Nejvýše je tedy třeba IP-adresy pro:

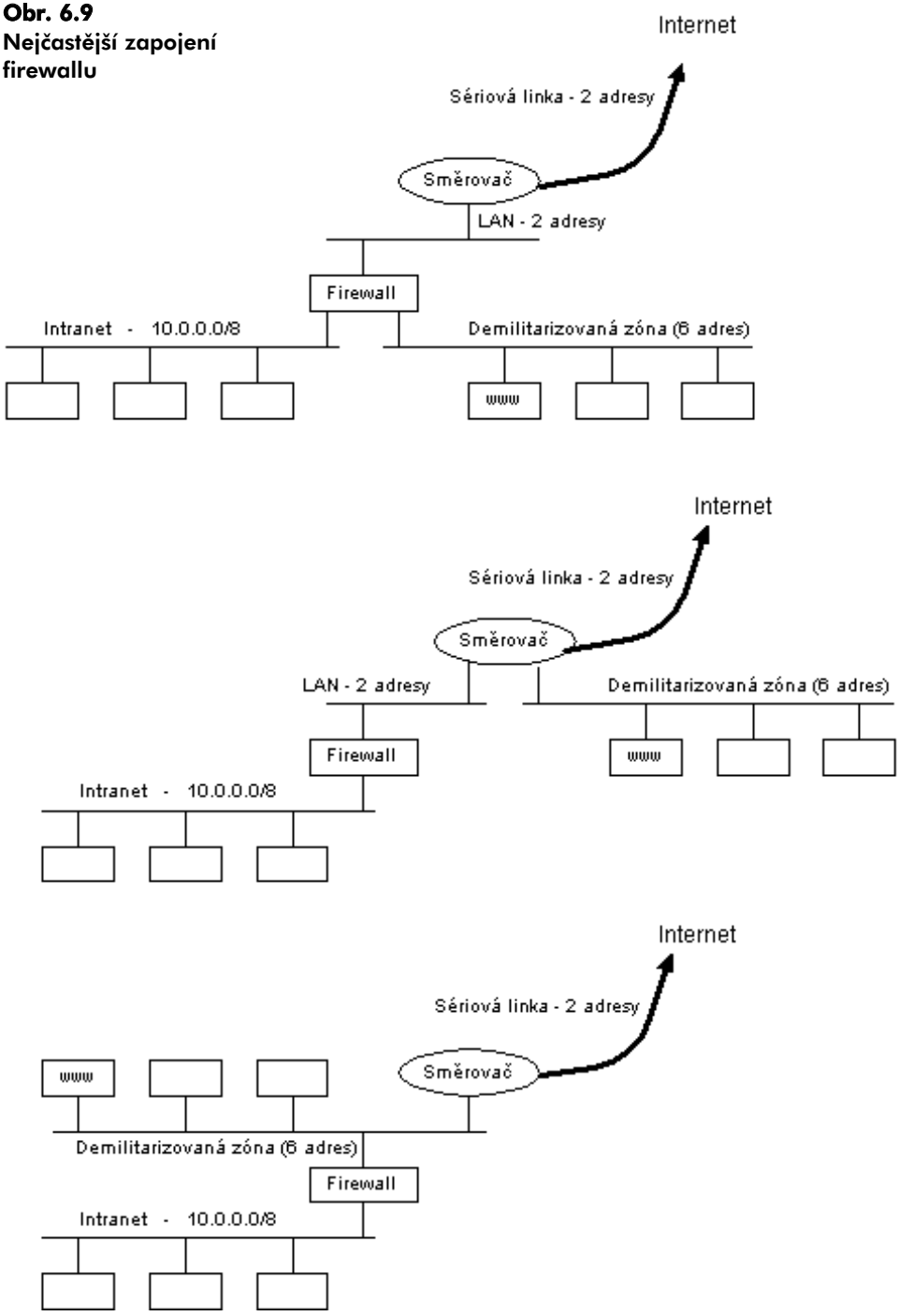
- Síť o čtyřech IP-adresách pro sériovou linku vedoucí do Internetu (může se i jednat o nečíslovanou síť).
- Síť pro „internetovskou" stranu firewallu též stačí o čtyřech adresách (není třeba pro třetí variantu zapojení firewallu).
- Síť pro demilitarizovanou zónu, kde je např. firemní WWW-server. Nezažil jsem, aby na demilitarizované zóně bylo více jak 5 počítačů.

6.6 Více jak 254 rozhraní na LAN

Někdy se stane, že na lokální síti je např. 300 počítačů. Nestačí tedy jedna síť třídy C. Přidělí se dvě sítě třídy C. Je zde nebezpečí chybné konfigurace v případě, že použijeme dvě samostatné sítě. Pak na LAN musí být i směrovač, který směruje mezi těmito sítěmi (nebo se použije proxy ARP). Počítače místo aby komunikovaly na síti přímo mezi sebou, tak musí využívat služeb směrovače. Horší na tom je, že data jdou sítí dvakrát, což při třech stech počítačích na LAN je obzvláště nepříjemné. Jednou od odesílatele na směrovač a podruhé ze směrovače k adresátovi.

Rozumné je v tomto případě použít supersíť skládající se ze dvou sítí třídy C, tj. supersíť s maskou /23, tj. 255.255.128.0.

Obr. 6.9
Nejčastější zapojení firewallu



7

Směrování

Směrování IP-datagramů (*IP routing*) a předávání IP-datagramů (*IP forwarding*) jsou dva procesy, na kterých Internet stojí.

Základní schéma směrování je zobrazeno na obr. 7.1.

Prohlédnete-li si obrázek 7.1 podrobně, pak naleznete i odpověď na otázku: „A proč musím konfigurovat programovou smyčku (*loopback*)". Vždy, když nějaké rozhraní zjistí, že by se něco mělo předat zpět na vstup, tak se to předá na programovou smyčku, která to zařídí.

Z obrázku 7.1 je také patrné, že při zpracování vstupů v některých případech operační systém informace automaticky předává na výstup (do procesu směrování), tj. aplikační programy do tohoto předávání nezasahují. Jedná se zejména o:

- Explicitní směrování (*source routing*).
- Předávání (*forwarding*).
- Požadavek o echo (*echo request*).
- Přesměrování (*redirect*).

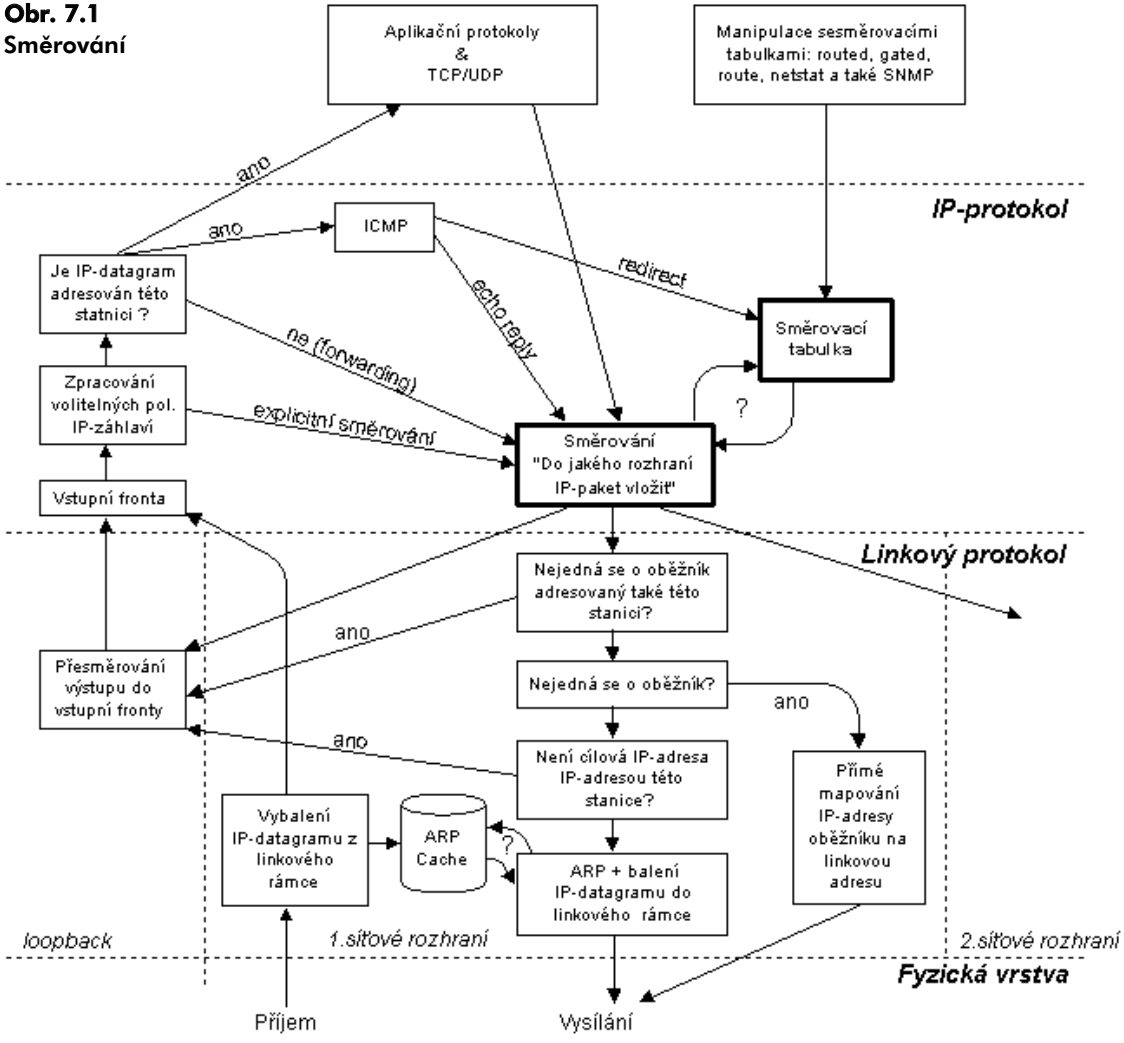
Operační systémy mají v jádře vždy nějaké parametry, kterými lze takováto automatické zpracování IP-datagramů zakázat. Velice častý je např. zákaz explicitního směrování, naopak zpracování požadavku o echo se zakazuje zřídka.

7.1 Předávání a filtrace

Předávání umožňuje stanici pracovat jako směrovač. Pokud stanice zjistí, že IP-datagram není adresován pro ní, pak se jej pokouší předat dále, tj. odeslat jako odesílá své IP-datagramy.

Předávání lze i zakázat – to bývá volba jádra operačního systému. U starších systémů bylo nutné pro takový zákaz znovu sestavit jádro operačního systému. U dnešních systémů je to možné provádět dynamicky (např. Windows NT a většina systémů UNIX). Někdy je však nutné systém po takové změně restartovat.

Obr. 7.1
Směrování

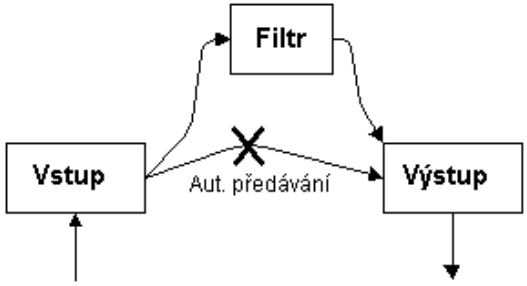


Zajímavou vlastností mnohých operačních systémů je, že IP-datagramy nepředávají mechanicky, ale provádějí filtraci (*screening*), tj. nepředávají všechny pakety, ale jen některé – prolustrované. Většinou filtrace pracuje tak, že před tím, než je IP-datagram předán, tak se celý proces předávání pozastaví a rozhodnutí zdali IP-datagram předat se ponechá na procesu (službě) běžícím na pozadí.

Předávaný IP-datagram se předá filtračnímu procesu, který buď předání schválí nebo zamítne. Filtrační proces se rozhoduje buď na základě informací v:

1. IP-záhlaví, např. není-li adresát nebo příjemce na černé listině.
2. TCP-záhlaví, např. podle čísel portu a nastavených příznaků ACK či SYN.
3. Aplikačního protokolu, což používají některé firewally.

Obr. 7.2
Filtrace



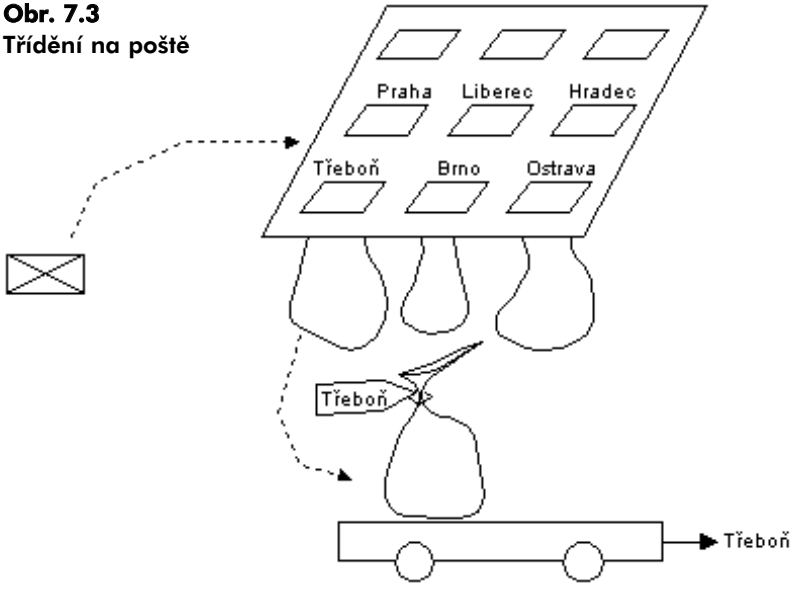
První dva typy filtrace jsou běžně implementovány na směrovačích. Třetí typ je záležitostí firewallů pracujících na principu filtrace (na rozdíl od firewallů pracujících na principu proxy).

7.2 Směrování

Směrování IP-datagramů je velice podobné třídění dopisů na poště. Na poště mají třídící stůl s vyřezanými otvory. Pod každým otvorem je přivázán poštovní pytel. Nad otvorem jsou napsány názvy měst kam je z místní pošty přímé poštovní spojení.

Třídění probíhá tak, že poštovní úředník bere dopis za dopisem. Na každém dopisu si prohlédne adresu. Je-li adresát z Brna, pak dopis vhodí do otvoru Brno. Je-li adresát z Roztok u Prahy, pak dopis vhodí do otvoru Praha (protože do Roztok není přímé poštovní spojení, to je nejblíže Roztokům do Prahy). Až poštovní úředník vytřídí všechny dopisy, pak pytel po pytli odváže z třídícího stolu. Každý pytel zaváže a přiváže k němu visačku, na kterou napíše název města, kam se má pytel odeslat. Poté se pytel naloží …

Obr. 7.3
Třídění na poště

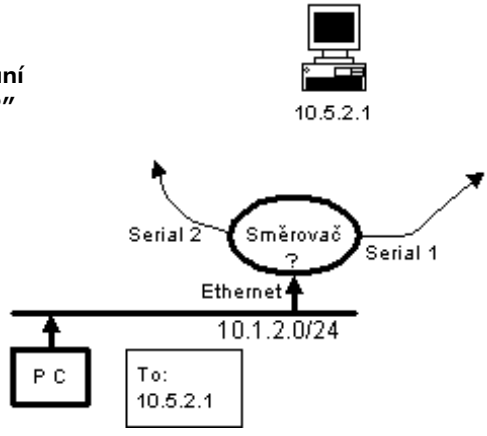


Směrovač netřídí dopisy, ale IP-datagramy. Tento proces se nazývá směrováním. Směrovač obdrží IP-datagram a musí rozhodnout, do kterého svého rozhraní jej má vhodit, kterému svému sousedovi (*next hop*) jej má poslat.

Zjednodušeně řečeno směrovač je zařízení, které předává IP-datagramy z jednoho svého rozhraní do jiného rozhraní. Směrovač umí předat IP-datagram i do téhož rozhraní, ze kterého IP-datagram přišel. Považuje to však ze výstřednost, takže o tom odesílatele IP-datagramu upozorní ICMP-paketem „*redirect*".

Na následujícím obrázku směrovač obdržel IP-datagram adresovaný stanici 10.5.2.1 a musí rozhodnou, zdali jej vložit do rozhraní Serial1, Serial2 nebo snad zpět do rozhraní Ethernet?

Obr. 7.4
Dilema směrovače: „Do kterého rozhraní IP-datagram vložit?"



Směrovači k rozhodování slouží směrovací tabulka (obdoba třídícího stolu na poště). Náš směrovač má tabulku:

Síť	Maska	Next Hop	Síťové rozhraní	Metrika
192.168.1.0	255.255.255.0	192.168.254.5	Seriál 1	4
10.1.2.0	255.255.255.0	Lokální rozhraní	Ethernet	0
10.5.1.0	255.255.255.0	10.10.10.2	Seriál 2	3
10.5.0.0	255.255.0.0	10.5.5.5	Seriál 1	2
…				
0.0.0.0	0.0.0.0	10.10.10.2	Seriál 2	1

Směrovací tabulka má v prvním sloupci IP-adresu cílové sítě. Představme si pro jednoduchost, že směrovací tabulka je podle prvního sloupce sestupně tříděna. To nám umožní snadno aplikovat základní pravidlo směrování: **Více specifická adresa cílové sítě má přednost před méně specifickou**. Více specifickou adresou sítě se rozumí adresa, která má v síťové masce více jedniček. V případě, že by se ve směrovací tabulce našly dvě či více cest k cíli, pak se zvolí více specifická cesta. V případě, že se najdou dvě stejně specifické cesty, pak se zvolí cesta s nejnižší metrikou (cenou).

7.2.1 Zpracování

V případě, že jsou řádky směrovací tabulky sestupně tříděny, pak stačí směrovací tabulku procházet od shora dolů. Na každém řádku se vezme síťová maska, kterou se bit po bitu vynásobí IP-adresa příjemce IP-datagramu. Výsledek se porovná s prvním sloupcem. Pokud se výsledek nerovná IP-adrese sítě v prvním sloupci, pak se přejde na zpracování následujícího řádku. Pokud se výsledek shoduje s IP-adresou v prvním sloupci, pak se ještě otestuje následující řádek, zdali ve směrovací tabulce neexistuje ještě k cíli jiná cesta, (pak by vstoupila do hry metrika).

Vraťme se k příkladu z obr. 7.4. Směrovač je postaven před rozhodnutí kterým svým síťovým rozhraním IP-datagram o adrese 10.5.2.1 odeslat. Prochází směrovací tabulku:

1. Řádek:

192.168.1.0	255.255.255.0	192.168.254.5	Seriál 1	4

   Vynásobením bit po bitu cílové adresy 10.5.2.1 s maskou 255.255.255.0 obdržíme 10.5.2.0, což se nerovná IP-adrese sítě v prvním sloupci (ta je 192.168.1.0). Přecházíme na vyhodnocení následujícího řádku.

2. Řádek:

10.1.2.0	255.255.255.0	Lokální rozhraní	Ethernet	0

   Vynásobením bit po bitu cílové adresy 10.5.2.1 s maskou 255.255.255.0 obdržíme 10.5.2.0 ,což se nerovná IP-adrese sítě v prvním sloupci (ta je 10.1.2.0). Přecházíme na vyhodnocení následujícího řádku.

3. Řádek:

10.5.1.0	255.255.255.0	10.10.10.2	Seriál 2	3

   Vynásobením bit po bitu cílové adresy 10.5.2.1 s maskou 255.255.255.0 obdržíme 10.5.2.0, což se nerovná IP-adrese sítě v prvním sloupci (ta je 10.5.1.0). Přecházíme na vyhodnocení následujícího řádku.

4. Řádek:

10.5.0.0	255.255.0.0	10.5.5.5	Seriál 1	2

   Vynásobením bit po bitu cílové adresy 10.5.2.1 s maskou 255.255.0.0 obdržíme 10.5.0.0, což **se rovná** IP-adrese sítě v prvním sloupci (ta je 10.5.0.0). Budeme proto náš IP-datagram vkládat do rozhraní Serial 1 a předávat jej dalšímu směrovači o IP-adrese 10.5.5.5. Pokud by se nejednalo o sériovou linku, ale např. o Ethernet, pak by bylo třeba zjistit linkovou adresu směrovače o IP-adrese 10.5.5.5 protokolem ARP.

Poslední řádek obsahující v prvním sloupci 0.0.0.0 s maskou 0.0.0.0 se nazývá *default*. Tímto implicitním směrem jsou pak odesílány všechny IP-datagramy, pro které nevyhovoval žádný jiný řádek směrovací tabulky (všimněte si, že vyhovuje každé IP-adrese: nula krát nula je nula). Implicitní směr ve směrovací tabulce může a nemusí být – závisí to na správci, jak tabulku naplnil. Implicitní směr používají např. firmy pro cestu do Internetu.

S implicitním směrem se setkáváme i na silnici. Když jedu z Budějovic do Prahy, tak implicitní směr je do Prahy, ale na mnohých křižovatkách je značeno jen odbočení. Je tam šipka do Třeboně či do Bechyně, ale mnohdy chybí přímý směr do Prahy. Implicitně každý ví, že tahle silnice vede do Prahy (tj. *default* je do Prahy), tak to přece není třeba stále na každé mezi opakovat.

7.3 Manipulace se směrovacími tabulkami

Směrovací tabulku je třeba jednotlivými položkami naplnit. Položky jsou pak v tabulce trvale dokud je někdo nezruší nebo nevypne systém. Pokud je plní směrovací aplikační protokoly, pak je sledována doba jejich života, po které jsou z tabulky vypuštěny.

V příkazech se anglicky často nepoužívá slovo *router*, ale *gateway*. S čímž se setkáváme zejména ve starší literatuře. Ve směrovací tabulce se tím rozumí následující směrovač (*next hop*).

7.3.1 Výpis obsahu směrovací tabulky v NT

Příkaz *netstat* vypisuje obsah směrovací tabulky setříděn vzestupně, takže pokud chcete vyhodnocovat tabulku jak jsem popsal, pak ji musíte procházet zdola nahoru. Trochu nezvyklé je, že rozhraní (*interface*) se jmenují svou IP-adresou, ale po chvíli si na to zvyknete. Co je na tom nezvyklé? V následujícím výpisu směrovací tabulky mělo PC rozhraní o IP-adrese 194.149.104.121. Avšak když se podíváte na první sloupec, tak IP-datagramy adresované adresátovi 194.149.104.121 se mají vkládat do rozhraní 127.0.0.1. Je to správně, protože se jedná o adresu lokálního síťového rozhraní.

```
C:\> netstat
Route Table
Active Routes:
  Network Address          Netmask  Gateway Address        Interface  Metric
          0.0.0.0          0.0.0.0  194.149.104.126  194.149.104.121       1
        127.0.0.0        255.0.0.0        127.0.0.1        127.0.0.1       1
   194.149.104.64  255.255.255.192  194.149.104.121  194.149.104.121       1
  194.149.104.121  255.255.255.255        127.0.0.1        127.0.0.1       1
  194.149.104.255  255.255.255.255  194.149.104.121  194.149.104.121       1
        224.0.0.0        224.0.0.0  194.149.104.121  194.149.104.121       1
  255.255.255.255  255.255.255.255  194.149.104.121  194.149.104.121       1
```

Síť 224.0.0.0 s maskou 224.0.0.0 označuje všechny adresné oběžníky (včetně rezervy IP-adres, tj. IP-adresy tříd D a E).

7.3.2 Výpis obsahu směrovací tabulky v UNIXu

Položky směrovací tabulky jsou opět vypisovány vzestupně, tzn. směrovací tabulku procházíme opět od spodu nahoru.

UNIX je podstatně starší operační systém. Na rozdíl od NT starší verze operačních systémů UNIX nevypisovaly síťovou masku – předpokládaly standardní síťovou masku, což při použití jiných masek vedlo k nepřehlednému výpisu.

Novější verze vypisují síťovou masku ve tvaru lomeno a počet jedniček masky. Navíc ještě před výpis směrovací tabulky vypíší všechny síťové masky, které se ve směrovací tabulce vyskytují. Např. (Digital UNIX 4.0D):

```
$ netstat -rn
Routing tables
Destination      Gateway            Flags     Refs     Use  Interface
Netmasks:
Inet             255.0.0.0
Inet             255.255.0.0
Inet             255.255.255.224

Route Tree for Protocol Family 2:
default          195.47.37.193      UGS          8    25686  tu0
10/8             195.47.37.193      UGS          0     4916  tu0
127.0.0.1        127.0.0.1          UH           1        0  lo0
172.17/16        195.47.37.193      UGS          2    21306  tu0
195.47.37.192/27 195.47.37.194      U           17    30404  tu0
```

Sloupec Refs ukazuje kolik je tímto směrem navázáno spojení protokolem TCP. Sloupec Use indikuje kolik IP-paketů bylo tímto směrem odesláno (zpravidla od startu systému).

Nejzajímavějším sloupcem je sloupec s příznaky (*Flags*). Příznaky mají následující vyznamy:

- U (up). Směr je dostupný.
- G (gateway). Příznak G určuje, že cesta k cílové síti vede přes směrovač. Tj. next hop je směrovač. Linková vrstva bude hledat linkovou adresu uvedeného směrovače, nikoliv přímo adresáta (ten není přímo dostupný).
- H (host). Příznak H určuje, že se je jedná o adresu rozhraní (počítače) nikoliv adresu sítě, tj. maska je 255.255.255.255.
- D. Položka byla vytvořena na základě ICMP-zprávy redirect.
- M. Položka byla modifikována na základě zprávy redirect.
- S (static). Jedná se o statickou položku vytvořenou příkazem route.
- R (reject). Tato položka byla rovněž vytvořena příkazem route.

7.3.3 Naplnění tabulky a rušení položek

Směrovací tabulka se plní:

- Při konfiguraci síťového rozhraní, kdy říkáme jakou má síťové rozhraní adresu a masku. V operačním systému UNIX se jedná o příkaz *ifconfig*.
- Staticky (ručně) příkazem *route*.
- Dynamicky ze ICMP-zpráv *redirect*.
- Dynamicky směrovacími (tj. aplikačními) protokoly.

Staticky se směrovací tabulka plní pomocí příkazu route. V operačním systému NT má příkaz route následující syntaxi:

```
ROUTE [-f] [command [destination] [MASK netmask] [gateway] [METRIC metric]]
```

`-f`	Vymaže nejprve obsah směrovací tabulky.
`-p`	U příkazu ADD zajistí, aby takto přidaná položka zůstala ve směrovací tabulce i po restartu PC, tj. stala se trvalou položkou. U příkazu PRINT způsobí, že se vypíší trvalé položky.
`command`	Určuje příkaz pro manipulaci se směrovací tabulkou, nabývá následujících hodnot:
	`PRINT` Vypiš obsah směrovací tabulky
	`ADD` Přidej položku do směrovací tabulky.
	`DELETE` Zruš položku ve směrovací tabulce.
	`CHANGE` Změň položku
`destination`	Specifikuje cílovou síť.
`netmask`	Specifikuje síťovou masku
`gateway`	Specifikuje next hop.
`METRIC`	Specifikuje metriku.

Příklad:

```
route -p add  10.0.0.32 mask 255.255.255.240  192.168.1.2
```

V operačním systému UNIX je příkaz route podstatně bohatší. Nejsou zde trvalé položky směrovací tabulky – po restartu se statické položky do směrovací tabulky vždy plní příkazem route (byť automaticky nějakou procedurou).

V operačním systému UNIX je i jiný repertoár příkazů pro manipulaci se směrovací tabulkou. Nebývá zde příkaz PRINT, ale naopak bývá příkaz flush (vymazání směrovací tabulky) a příkaz monitor, který způsobí živé vypisování změn ve směrovací tabulce na standardní výstup (ukončuje se ^C).

Příklad:

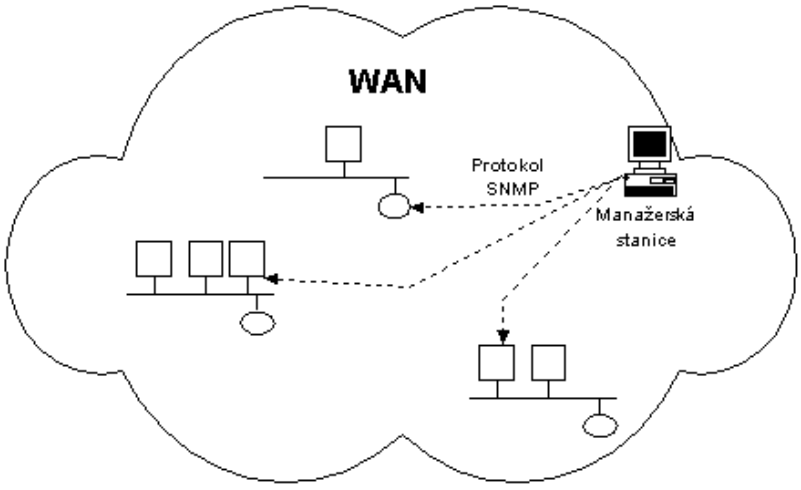
```
route add -net 10.0.0.32/28 192.168.1.2
```

7.3.4 Manipulace se směrovací tabulkou protokolem SNMP

Pokud ovšem nespravujeme jeden počítač, ale rozsáhlou síť počítačů, pak je velice náročné se postupně přihlašovat na jednotlivé systémy a tam dávat příkaz route. Zpravidla máme k dispozici manažerskou stanici a na všech aktivních prvcích sítě (počítače, směrovače, HUBy, modemy, databáze atd.) běží SNMP agenti, kteří jsou k dispozici manažerské stanici.

Z manažerské stanice je možné provádět dotazy na nejrůznější parametry jednotlivých systémů. Mj. je takovým parametrem i položka směrovací tabulky. Takže z manažerské stanice můžeme vypisovat obsah směrovacích tabulek, ale i směrovací tabulky modifikovat. Nesmíte si jen zmodifikovat směrovací tabulky tak, abyste ztratili spojeni s manažerskou stanicí …

Obr. 7.5
Protokol SNMP



7.4 Směrovací protokoly

Směrovací protokoly jsou aplikační protokoly, které neslouží uživatelům (osobám), ale směrovačům, aby si vzájemnou komunikací mezi sebou automaticky naplnily směrovací tabulky.

Je dvojí na sobě nezávislé dělení směrovacích protokolů:

- Na *Link State Protocols* (LSP) a na *Routing Vector Protocols* (RVP).
- Na IGP a EGP.

7.4.1 LSP a RVP

Protokoly RVP (*Routing Vector Protocols*) pracují tak, že si sousední směrovače mezi sebou vyměňují obsahy směrovacích tabulek (vektorem se míní jedna položka směrovací tabulky). Obdržím-li jednotlivé vektory ze směrovací tabulky svého souseda, pak si z nich mohu vybrat vektory, které ve vlastní směrovací tabulce nemám a doplnit je do vlastní směrovací tabulky. Nesmím zapomenout u takto doplněné položky zvýšit metriku. Tyto protokoly jsou jednoduché a snadno se implementují. Jejich nevýhodou je, že ve větších rozsáhlých sítích může výměna vektorů oscilovat a pak některé vzdálenější sítě mohou být chvilku dostupné a za okamžik již nikoliv. Většími sítěmi se rozumí sítě o více jak 10 LAN.

Příkladem protokolů RVP jsou protokoly RIP a RIP 2. V operačním systému UNIX je protokol RIP implementován programem *routed*. Protokolem RIP si sousední směrovače vyměňují pomocí všeobecných oběžníků (*broadcast*) obsahy svých směrovacích tabulek. Nevýhodou je, že v tomto protokolu není v položce směrovací tabulky uváděna síťová maska. Proto lze protokol RIP použít jen tehdy, když v síti používáme pouze sítě se standardní maskou. Protokol RIP 2 tuto nevýhodu odstraňuje. RIP 2 šíří obsahy směrovacích tabulek zpravidla pomocí adresného oběžníku (*broadcast*) o IP-adrese 224.0.0.9. Nevýhodou protokolu RIP 2 je, že je jen zřídka implementován.

Jelikož jsou protokoly RVP vhodné pro menší rozsáhlé sítě, pak je vždy otázkou, zda-li se bude konfigurace sítě opravdu natolik dynamicky měnit, nebo se sice pracněji, ale lépe nakonfigurují směrovače ručně pomocí statických položek.

Protokoly LSP pracují na zcela odlišném principu. Každý směrovač si zjistí, jaké směrovače má za své sousedy a v pravidelných intervalech testuje jejich dostupnost. Celou síť pak zaplavuje svými oběžníky o tom, koho má za své sousedy. Takže každý směrovač má od všech ostatních směrovačů zprávu o tom jaké mají sousedy. Takže každý směrovač má seznam všech cest v síti. Na tento seznam se pustí algoritmus nejkratší cesty, kterým se zjišťuje směr kam se má IP-datagram odeslat. Tj. položky směrovací tabulky se počítají algoritmem nejkratší cesty z dat obdržených od ostatních směrovačů.

U rozsáhlých sítí je problematické zaplavovat je velkým množstvím informací ze směrovačů, proto se takové sítě rozdělí na oblasti a zmíněný postup se aplikuje pouze v rámci této oblasti. Na hranicích se sousedními oblastmi jsou hraniční směrovače, které si pak vyměňují informace o celých oblastech.

Protokoly LSP jsou oproti protokolům RVP nesrovnatelně stabilnější a lze je aplikovat i u velmi rozsáhlých sítí. Nevýhodou je, že návrh sítě, tj. rozdělení sítě na oblasti musí provést zkušený odborník, rovněž konfigurace je netriviální. Pokud se použije protokol typu LSP bez větších zkušeností, tak je také možné, že některými linkami data prostě nepotečou a jiné budou přetížené.

Příkladem protokolů LSP jsou protokoly OSPF a IS-IS. Tyto protokoly mají různý výklad pojmu oblast. Protokol OSPF je určen pouze pro směrování IP, kdežto IS-IS podporuje směrování několika protokolů současně (např. IP a DECnet). O protokolu IS-IS platí dvojnásob, že je třeba, aby jej konfiguroval zkušený odborník. Protokol IS-IS je určen pro komunikaci mezi směrovači, pro komunikaci mezi směrovačem a koncovou stanicí zde existuje protokol IS-ES, který však v případě protokolu IP není nutný, protože koncová stanice může mít ve své směrovací tabulce položku default směrující k nejbližšímu směrovači.

7.4.2 IGP a EGP

Protokoly IGP jsou určeny pro činnost v rámci autonomního systému. Již zmíněné protokoly RIP, RIP2, OSPF i IS-IS jsou vesměs protokoly IGP.

Ovšem poskytovatelé Internetu si mezi sebou potřebují také vyměňovat směrovací informace. Poskytovatelé Internetu pro výměnu směrovacích informací mezi autonomními systémy používají protokoly EGP. V dnešní době používají protokol BGP (*Border Gateway Protocol*) verze 4.

Protokoly EGP se liší od protokolů IGP zejména tím, že ve směrování umožňují zohlednit směrovací politiku (tj. kdo komu platí).

7.4.3 Agregace

Agregace je proces, kdy se z několika položek směrovací tabulky udělá jedna položka. Tento proces je žádoucí např. při propagaci sítí vně autonomního systému.

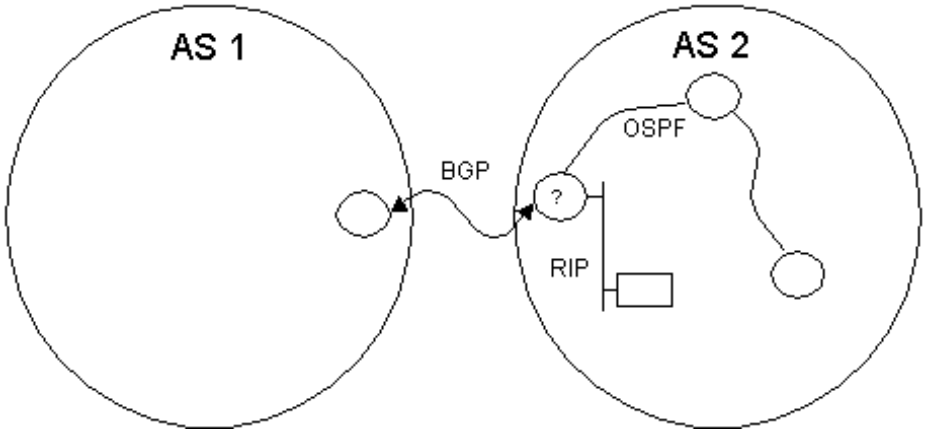
Jednu položku můžeme z více udělat tehdy, když sítě slučované do jedné položky vytvoří supersíť. Automatická agregace je spíše přání než realita. Poskytovateli většinou vypadnou z jeho supersítí některé adresy, které ještě nikomu nepřidělil, takže nelze automaticky agregovat všechny IP-adresy autonomního systému do jedné nebo několika málo položek. Prakticky se agregace provede ručně tak, že se vně autonomního systému propagují všechny IP-adresy, které jsou přiděleny.

7.4.4 Redistribuce

Na obrázku 7.6 je otazníkem označen směrovač, který si vyměňuje současně směrovací informace protokoly BGP, OSPF, RIP a k tomu možná má ve směrovací tabulce několik statických položek.

Otázka je, zdali se mají informace (položky směrovací tabulky) získané jedním směrovacím protokolem propagovat do ostatních směrovacích protokolů, tj. má-li se provést redistribuce.

Obr. 7.6
Redistribuce



Položka ve směrovací tabulce musí v sobě nést tedy také informaci jakým protokolem byla vytvořena.

8

IP nové generace

IP-protokol verze 6 se často označuje jako IP-protokol nové generace (*IP Next Generation*) odkud je i zkratka IPng.

IP-protokol verze 4 byl specifikován RFC-760 v lednu 1980, tato specifikace byla v září 1981 nahrazena RFC-791. IP-protokol verze 6 byl v prosinci 1995 specifikován RFC-1883.

Patnáct let je v Internetu ohromný časový rozdíl. IPng přináší nejen zvětšení IP-adresy ze čtyř na šestnáct bajtů, ale i filozoficky zcela nový pohled na stavbu IP-datagramu. V záhlaví IP-datagramu chybí kontrolní součet záhlaví a mnohá další málo využívaná pole jsou přesunuta ze základního záhlaví do nepovinných dalších hlaviček.

IP-datagram verze 6 se skládá ze čtyřicet bajtů dlouhého základního záhlaví následovaného rozšířeními. Čtyřicet bajtů základního záhlaví se může zdát hodně, ale je třeba si uvědomit, že jen IP-adresy odesílatele a příjemce zaberou 32 bajtů.

Struktura základního záhlaví je zobrazena na obr. 8.1. Nyní se zastavíme u významu jednotlivých polí základního záhlaví.

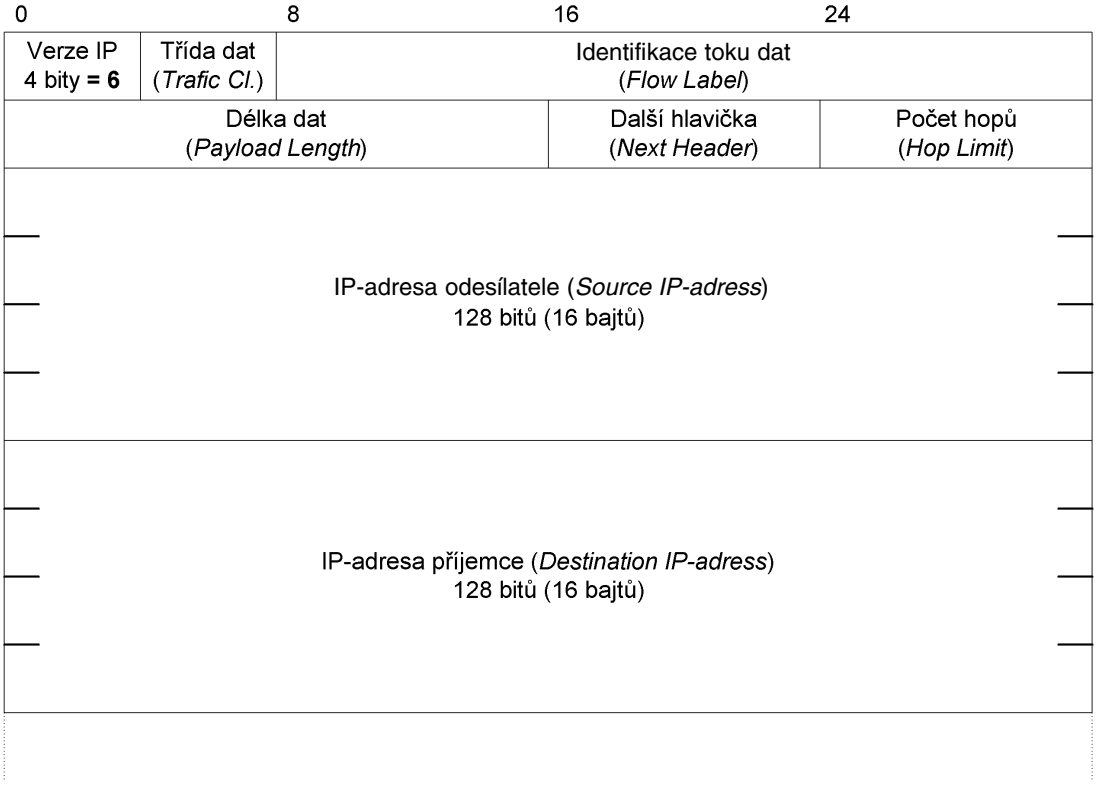
Pole **verze IP** obsahuje hodnotu 6 pro IP-protokol verze 6.

Pole **třída dat** se skládá ze čtyř bitů, tj. nabývá hodnoty 0 až 15. Specifikuje přenášená data pro případ rozhodování v okamžiku zahlcení sítě. V okamžiku zahlcení musí směrovač IP-datagramy zahazovat. V tomto poli se specifikuje, které IP-datagramy je možné zahodit dříve než jiné.

Interval 0-15 je rozdělen na dvě části:

- 0-7 je určeno pro klasický provoz:
  0 – nespecifikovaná data
  1 – provoz na pozadí (např. news)
  2 – automatický provoz (např. mail)
  4 – uživatelem (člověkem) prováděné velké přenosy dat (např. ftp)
  6 – interaktivní provoz (např. telnet, X-windows)
  7 – řízení sítě (např. směrovací protokoly, SNMP)

- 8-15 je určeno pro přenosy v reálném čase (např. audio). Datagramy s nižší hodnotou si uživatel přeje zahodit dříve než datagramy s vyšší hodnotou. To však platí pouze v intervalu 8-15, protože intervaly 0-7 a interval 8-15 se zpracovávají odděleně.



Obr. 8.1 Základní záhlaví IP-datagramu verze 6

Pole **identifikace toku dat** je naprosto neotřelá myšlenka. Spolu s adresou odesílatele jednoznačně identifikuje jeden dílčí tok dat v Internetu. Doposud se směrování provádělo výhradně na základě adresy příjemce. Nevýhodou směrování v Internetu je, že jednotlivé IP-datagramy se dopravují samostatně, tj. pokud Internetem prochází tok IP-datagramů mezi dvěma aplikacemi, pak směrovače na cestě řeší směrování pro každý procházející datagram samostatně. Např. pokud přenášíte několik MB dlouhý soubor, pak Internetem tečou tisíce datagramů. Každý směrovač na cestě se musí zabývat každým z těchto datagramů samostatně. Pokud nedojde ke změně v topologii sítě, pak pro tisíce datagramů směrovač řeší stejnou úlohu se stejným výsledkem.

Myšlenka spočívá v tom, že datagramy jednoho toku dostanou svou identifikaci. Pak stačí, aby směrovač vyřešil úlohu (do kterého rozhraní datagram předat) pro první datagram toku a do paměti si poznamenal výsledek. Pro další datagram nejprve prohlédne paměť a pokud by tam nenašel poznamena-

ný tok, tak řeší úlohu směrování. Další datagramy stejného toku pak bude předávat do stejného rozhraní aniž by řešil úlohu směrování – pouze na základě údajů v paměti.

Tok je určen adresou odesílatele a polem identifikace toku dat.

Položka v paměti směrovače nesmí zůstat déle než 6 vteřin. Nebezpečí číhá totiž v tom, že odesílatel resetuje svůj počítač. Zavede znovu operační systém a shodou okolností vygeneruje stejnou identifikaci pro jiný tok. Nepředpokládá se, že by uživatel stihl provést reset počítače do 6 vteřin.

Jinou možností je využít identifikaci toku dat k zajištění šířky přenášeného pásma. Směrovače na cestě od odesílatele k příjemci se nakonfigurují tak, aby pro datový tok o jisté identifikaci zajišťovaly určitou šířku pásma. Datagramy docházejí na směrovač, kde se umísťují do vyrovnávací paměti. Za normálních okolností se jedná o frontu datagramů, která z jednoho konce přibývá a z druhého konce se datagramy odebírají (odesílají). Směrovač však nemusí frontu zpracovávat sekvenčně, ale může přednostně vybírat datagramy tak, aby zajistil dohodnutou šíři pásma. V tomto případě pochopitelně nelze hrát na šestivteřinový limit.

Pole **délka dat** specifikuje délku IP-datagramu bez základního záhlaví. Jelikož je toto pole dlouhé 2 bajty, tak největší délka přenášeného IP-datagramu může být 65 535 bajtů. Je však možné použít volbu ohromný datagram v další hlavičce informace pro směrovače, která pak umožňuje odesílat ještě větší datagramy.

Pole **další hlavička** specifikuje typ následující hlavičky. Jako vnořená hlavička mohou být např. typy uvedené v tab. 8.1. Mechanismu dalších hlaviček je věnována kap. 8.1.

0	**Informace pro směrovače *(Hop-by-Hop Header)***
4	IP-protokol
6	Protokol TCP
17	Protokol UDP
43	**Směrovací informace *(Routing Header)***
44	**Záhlaví fragmentu *(Fragment Header)***
45	Protokol IRP
46	Protokol RRP
50	**Bezpečnostní hlavička *(Encapsulating Security Payload)***
51	**Autentizační hlavička *(Authentication Header)***
58	**Protokol ICMP**
59	**Další hlavička již nenásleduje**
60	**Jiná volba *(Destination Options)***

Tab. 8.1 Typy dalších hlaviček v IP-datagramu verze 6

Pouze typy, které jsou v tab. 8.1 uvedeny tučně jsou součástí IP-protokolu. Ostatní typy jsou typy protokolů vyšších vrstev.

Pole **počet hopů** v podstatě odpovídá položce TTL (doba života datagramu) v IP verze 4. Využít jej lze především jedním z následujících způsobů:

1. K zahazování zatoulaných datagramů. IP-datagramu je položka počet hopů snižována při průchodu každým směrovačem. Po dosažení nuly je datagram považován za zatoulaný a je zahozen.
2. K nalezení nejkratší cesty v Internetu (obdoba příkazu *traceroute*). Cílem je vyhledat nejbližšího člena určitého adresného oběžníku. Nejprve se odešle datagram jehož adresátem je adresný oběžník (*multicast*) s nastaveným polem počet hopů na jedničku. Pokud se žádný člen neozve, pak se odešle datagram s polem počet hopů nastaveným na dvojku atd.

8.1 Další hlavičky v IP-datagramu

Za základním záhlavím IP-datagramu mohou následovat další hlavičky.

Pole další hlavička v základním záhlaví ukazuje jaký typ dat ("jaká hlavička") následuje za základním záhlavím. Teoreticky může ukazovat již na TCP segment nebo jiný protokol vyšší vrstvy. Avšak může také ukazovat na další rozšiřující hlavičky IP-protokolu.

Pokud ukazuje na další rozšiřující hlavičku, pak i tato hlavička má pole „další hlavička", která ukazuje na další hlavičku. Hlavičky tak tvoří řetězec. Řetězec obsahuje jen ty hlavičky, které jsou nutné. Je to rozdíl od IP-protokolu verze 4, který ve svém záhlaví často přenáší nadbytečné informace.

Za polem další hlavička je pole délka hlavičky. Pole délka hlavičky specifikuje posunutí jaké je třeba udělat k další hlavičce. Základní záhlaví délku nemá, protože je dlouhé vždy 40 bajtů. Rovněž u záhlaví fragmentů se délka nepoužívá, protože tato hlavička je vždy 8 bajtů dlouhá.

8.1.1 Informace pro směrovače

Tato hlavička obsahuje jednotlivé informace (volby), které jsou určeny pro směrovače přepravující datagram. Každý směrovač, který datagram předává se musí těmito volbami zabývat.

Volba se skládá z pole typ dlouhého 1 B, pole délka dlouhého 1 B a pole obsahujícího vlastní volbu. Pole typ se skládá z osmi bitů:

```
aabxxxxx
```

bity `aa` sdělují co má směrovač s datagramem udělat, když nerozezná tuto volbu. Možnosti jsou:

00 Pokud volbu nerozpoznáš, pak ji ignoruj a datagram zpracovávej dále.

01 Pokud volbu nerozeznáš, pak datagram zahoď a neprováděj žádné další akce.

10 Pokud volbu nerozeznáš, pak datagram zahoď a informuj o tom odesílatele protokolem ICMP.

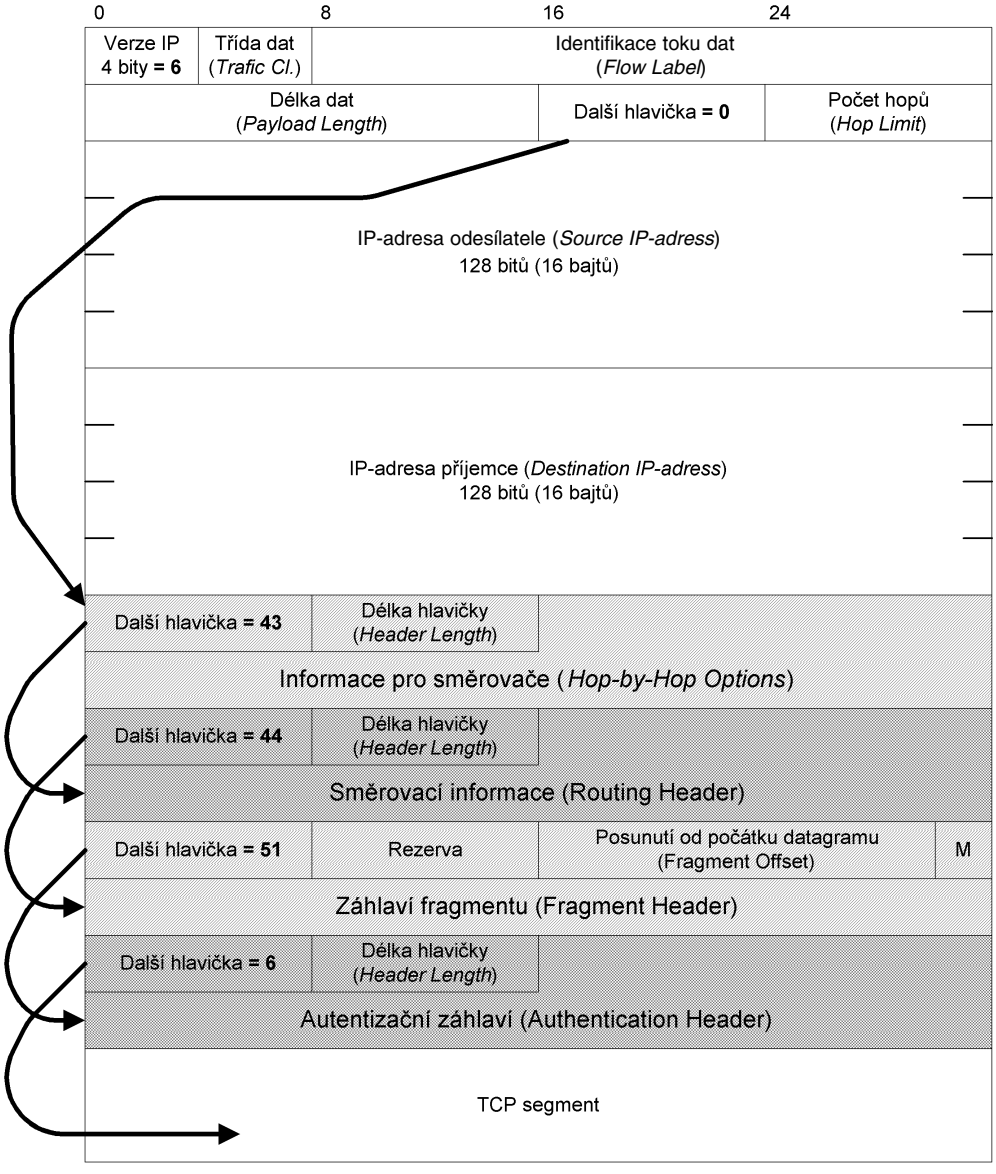
11 Pokud volbu nerozeznáš, pak datagram zahoď a v případě že datagram není adresován adresnému oběžníku, tak o tom informuj protokolem ICMP odesílatele.

Bit `b` sděluje, zdali může směrovač tuto volbu změnit:

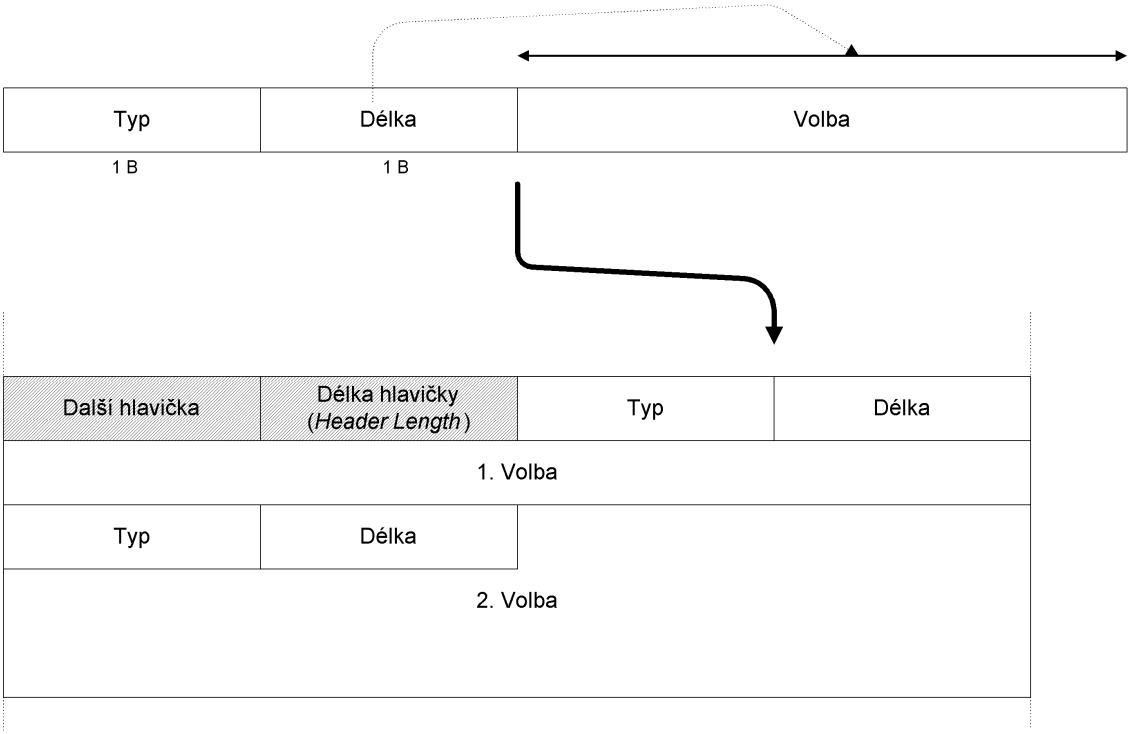
0 – Volba se nesmí změnit.

1 – Volba se může změnit.

Tabulka 8.2 uvádí některé volby. Volba výplň je dlouhá jeden bajt a skládá se pouze z typu (nula), pole délka a volba má prázdné.



Obr. 8.2 Struktura IP-datagramu verze 6, šipky vyjadřují návaznost jednotlivých hlaviček



Obr. 8.3 Struktura jedné volby z hlavičky informace pro směrovače

Typ desítkově	Typ dvojkově	Název	Délka	Hodnota v poli délka
0	0	Výplň dlouhá 1 bajt	1	není
1	1	Výplň dlouhá n bajtů	2 + n	n
194	11000010	Rozsáhlý datagram	2 + 4	4

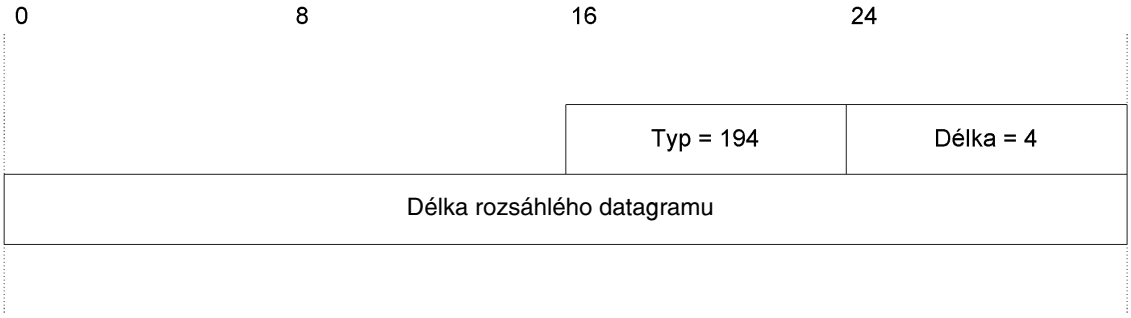
Tab. 8.2 Některé volby z hlavičky informace pro směrovače

Výplně jsou určeny pro zarovnání hlavičky na určitou délku. Volba rozsáhlý datagram využívá právě zarovnání. Tato volba musí totiž končit na hranici čtyřbajtu (viz obr. 8.4).

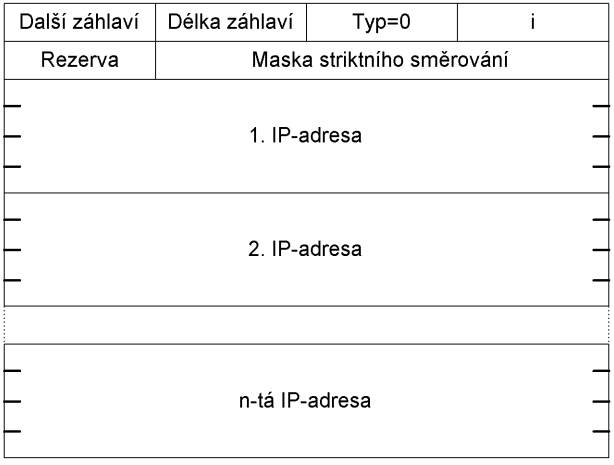
Pokud je volba délka rozsáhlého datagramu použita, pak délka datagramu v základním záhlaví je nastavena na nulu a používá se délka uvedená v této volbě. Zatímco v základním záhlaví jsou pro délku určeny 2 bajty (tj. datagram může být dlouhý až 64 KB), pak 4 bajty volby rozsáhlý datagram umožňují délku až 4 GB.

8.1.2 Směrovací informace

Hlavička směrovací informace používá tč. jedinou volbu (Typ=0), kterou je explicitní směrování. Kdy odesílatel specifikuje IP-adresy směrovačů, přes které má být datagram dopravován (viz obr. 8.5).



Obr. 8.4 Volba délka rozsáhlého datagramu končící na hranici čtyřbajtu



Obr. 8.5 Volba explicitní směrování

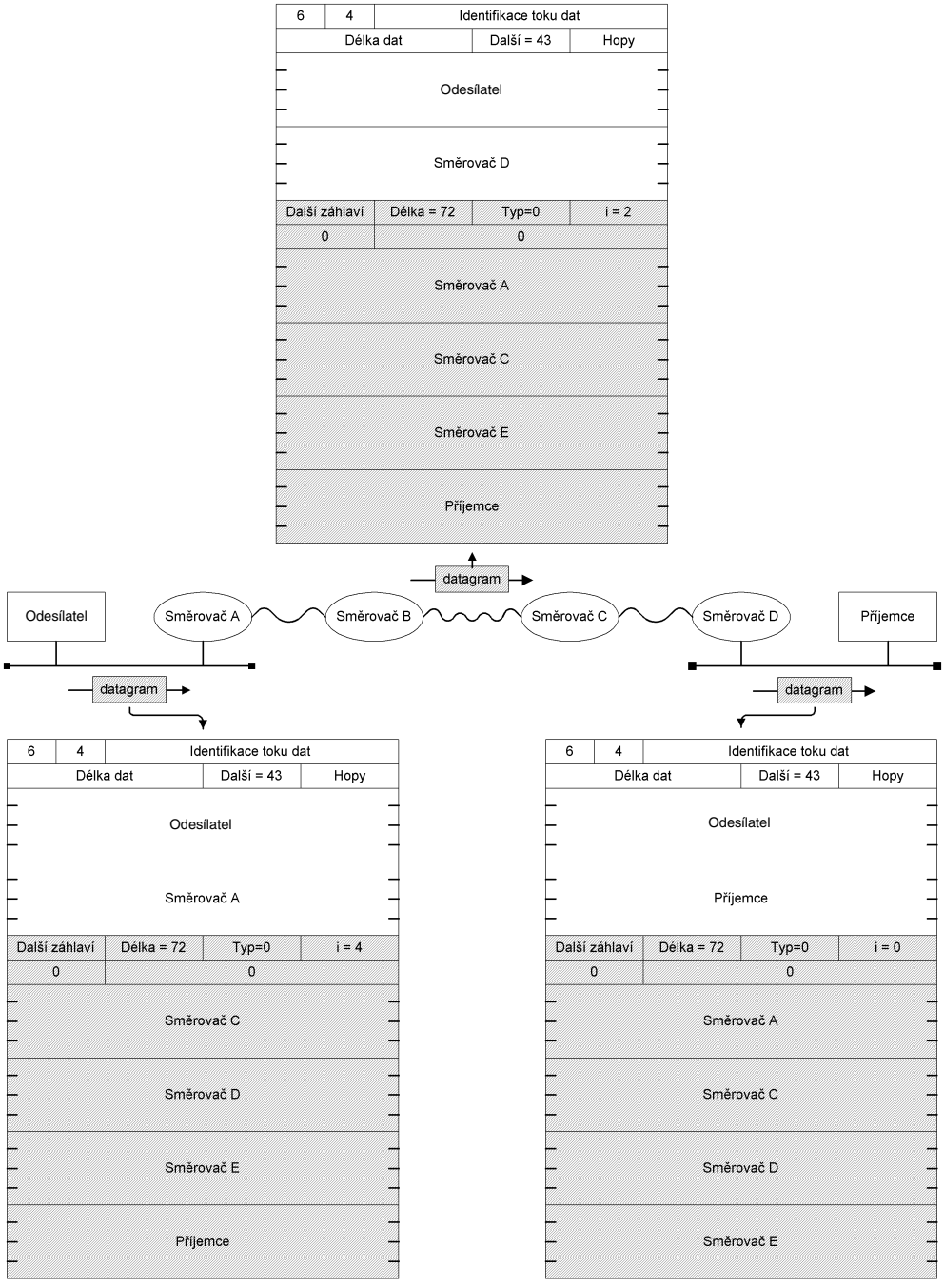
Ve spodní části hlavičky jsou uvedeny IP-adresy směrovačů, přes které si odesílatel přeje, aby byl datagram směrován.

Pole **Maska striktního směrování** obsahuje 24 bitů (bity 0 až 23 číslované zleva doprava). Každý bit odpovídá jednomu „hopu" a říká, zdali následující v hlavičce uvedený směrovač musí být sousední (tzv. striktní směrování) nebo zdali mezi ním mohou být další směrovače. Pakliže je bit nastaven na 1, pak se jedná o explicitní striktní směrování na následující „hop".

Pole i určuje přes kolik vyjmenovaných směrovačů má být datagram ještě směrován. Každý směrovač, který je v hlavičce vyjmenován snižuje hodnotu tohoto pole o 1.

Problém je ale v tom, že pokud má být datagram přes nějaký směrovač směrován, tak IP-adresa tohoto směrovače musí být uvedena v adrese příjemce. Proto je v adrese příjemce vždy vložena IP-adresa následujícího směrovače, přes který má být datagram dopravován. Adresa skutečného příjemce se pak uloží do hlavičky směrovací informace mezi ostatní vyjmenované směrovače.

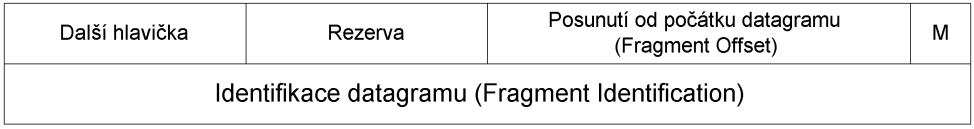
Příklad koloběhu IP-adres v hlavičce směrovací informace je uveden na obr. 8.6.



Obr. 8.6 Příklad koloběhu IP-adres při explicitním směrování

8.1.3 Záhlaví fragmentu

Fragmentovat IP-datagramy verze 6 může pouze operační systém na straně odesílatele. Směrovače, kterými datagram prochází na své cestě od odesílatele k příjemci fragmentovat na rozdíl od IPv4 již nesmí.



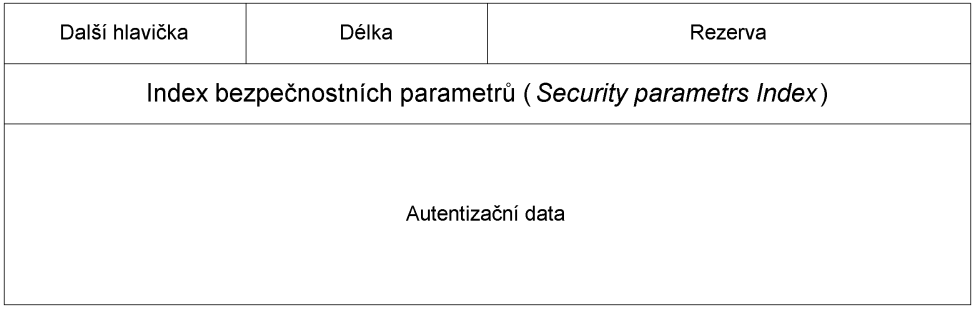
Obr. 8.7 Záhlaví fragmentu

Na rozdíl od IP-protokolu verze 4 neobsahuje každý IP-datagram (např. v základním záhlaví) identifikaci IP-datagramu. Identifikace IP-datagramu je nutná pro fragmentaci, aby příjemce zjistil, které fragmenty patří do stejného datagramu. V IPv6 je identifikace datagramu pouze v dalším záhlaví, tj. není součástí každého IP-datagramu.

Pole posunutí od počátku datagramu slouží k sestavení datagramů. Pomocí tohoto pole příjemce zjistí pořadí v jakém má fragmenty skládat za sebe. Pole posunutí od počátku datagramu neudává posunutí v bajtech, ale v násobcích osmi bajtů (v „osmibajtech"), proto se při fragmentaci musí zajistit, aby fragmenty byly odřezávány v délce, která je dělitelná osmi.

A konečně pole M (More Fragment) indikuje poslední fragment. Jednobitové pole M je nastaveno na jedničku, pouze v případě posledního fragmentu na nulu.

8.1.4 Autentizační hlavička



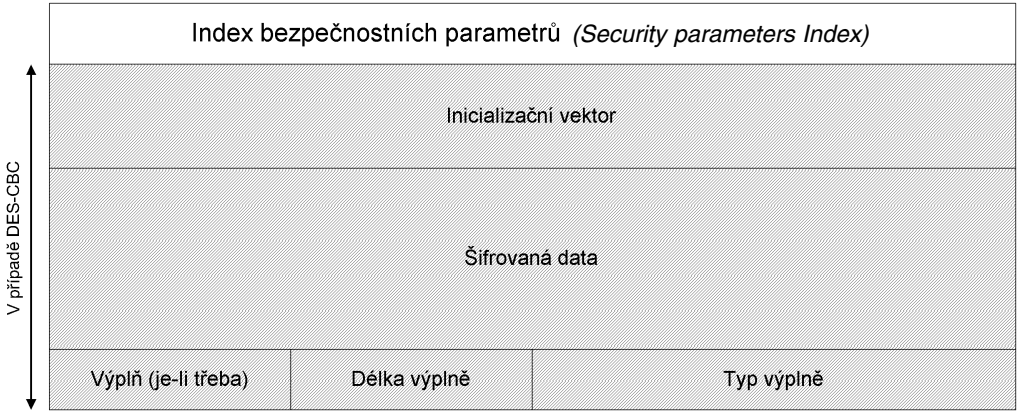
Obr. 8.8 Autentizační hlavička

Pomocí autentizační hlavičky odesílatel autentizuje data, tj. prokazuje, že data odeslal on. Datagram je tak zabezpečen proti změně IP-datagramu útočníkem i proti záměně celých IP-datagramů odesílatele za IP-datagramy vložené útočníkem.

Základní autentizace je za využití algoritmu MD-5 (RFC-1321) pro výpočet kontrolního součtu. Autentizační data jsou kontrolním součtem počítaným z tajemství zřetězeného s IP-datagramem. IP-datagram, který vstupuje do výpočtu má však vynulována všechna pole v záhlaví, která se mohou měnit (např. pole počet hopů).

Tajemství je 128 bitů dlouhý řetězec (je-li menší, tak se doplní nulami). Tento řetězec si předem musí vyměnit odesílatel s příjemcem. Těchto řetězců – tajemství může mít jak odesílatel, tak i příjemce více. Má je uloženy v tabulce. Pole Index bezpečnostních parametrů je indexem do této tabulky řetězců – tajemství.

8.1.5 Bezpečnostní hlavička



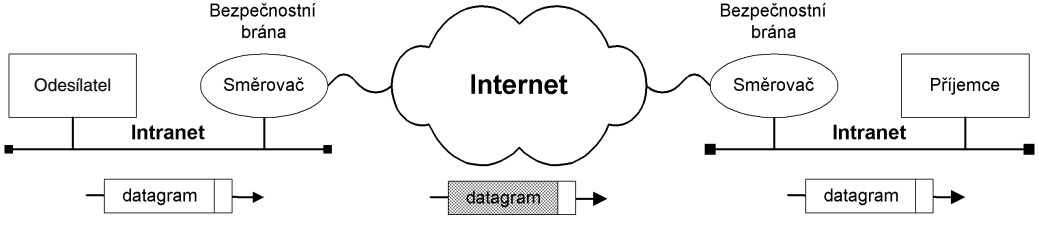
Obr. 8.9 Bezpečnostní hlavička

Bezpečnostní hlavička umožňuje přenášená data šifrovat. Musí to být poslední hlavička IP-datagramu, protože data jsou již dále šifrována, tj. nedostupná ke zpracování směrovačům dopravujícím IP-datagram.

První čtyři bajty jsou obdobně jako u autnetizační hlavičky indexem do tabulky udržující informace nutné pro šifrování (typ šifry, mód šifry, šifrovací klíče atd.). Na obr. 8.9 je příklad struktury šifrovaných dat pro případ, že je použita šifra DES v módu CBC.

Možnosti využití bezpečnostní hlavičky jsou překvapivě dvě (a jejich kombinace):

1. Šifrování provádí odesílatel, dešifrování příjemce.
2. Odesílatel ani příjemce se šifrováním nezabývají, ale směrovače přepravující datagram jej šifrují/dešifrují při průchodu méně bezpečnou sítí. Tyto směrovače se označují jako bezpečnostní brány (*Security Gateway*).



Obr. 8.10 Bezpečnostní brána

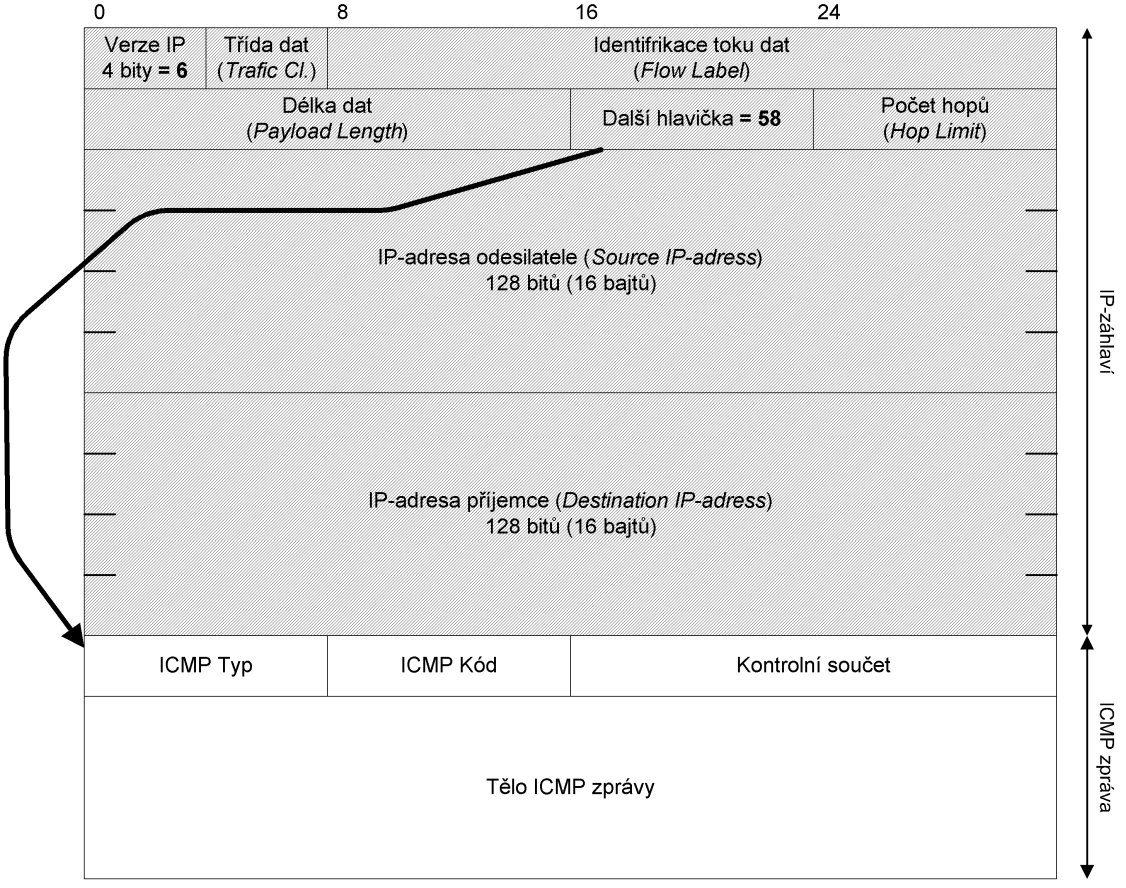
Bezpečnostní brány je praktické použít pro oddělení intranetu od Internetu. Provoz v intranetu je možný nešifrovaný, kdežto provoz mezi dvěma částmi firmy přes Internet se zabezpečí šifrováním.

8.2 Protokol ICMPv6

Podobně jako v případě IP-protokolu verze 4 se pro signalizaci chybových stavů a pro diagnostiku používá protokol ICMP, tak i v IPng se používá protokol ICMP. Protokol ICMP pro IP-protokol verze 6 je specifikován RFC-2463.

Protokol ICMPv6 v mnoha případech přináší zcela odlišné funkčnosti oproti protokolu ICMPv4. Řeší např. překlad IP-adres na linkové adresy. (Na tento problém jsou v IPv4 určeny samostatné protokoly ARP a RARP.)

Z hlediska struktury paketu se ICMP paket jeví jako protokol vyšší vrstvy, tj. je předcházen základním záhlavím IP-protokolu i případnými dalšími hlavičkami (viz obr. 8.11).



Obr. 8.11 Struktura ICMP paketu

Pole **ICMP typ** obsahuje typ zprávy (hrubší dělení zpráv) a pole **ICMP kód** obsahuje jemnější dělení zpráv.

RFC-2461 a RFC-2463 specifikují typy a kódy ICMP zpráv, které jsou uvedeny v tab. 8.3.

Typ	Kód	Popis
1		Nedoručitelný IP-datagram *(Destination Unreachable)*
	0	Ve směrovací tabulce neexistuje směr pro tuto adresu *(no route to destination)*
	1	Spojení s adresátem je administrativně uzavřeno *(communication with destination administratively prohibited)*
	3	Nedosažitelná adresa *(address unreachable)*
	4	Nedosažitelný port *(port unreachable)*
2	0	Příliš velký datagram *(Packet Too Big)*
3		Čas vypršel *(Time Exceeded)*
	0	Dosažen počet hopů *(hop limit exceeded in transit)*
	1	Vypršel čas na sestavení IP-datagramu z jeho fragmentů *(fragment reassembly time exceeded)*
4		Chybný parametr *(Parameter Problem)*
	0	Chybné pole v záhlaví *(erroneous header field encountered)*
	1	Nepodporovaný typ v poli další hlavička *(unrecognized Next Header type encountered)*
	2	Nepodporovaná volba *(unrecognized IPv6 option encountered)*
128	0	Žádost o echo *(Echo Request)*
129	0	Echo *(Echo Reply)*
133	0	Žádost o směrování *(Router Solicitation)*
134	0	Oznámení o směrování *(Router Advertisement)*
135	0	Žádost o linkovou adresu *(Neighbor Solicitation)*
136	0	Oznámení o linkové adrese *(Neighbor Advertisement)*
137	0	Změň směrování *(Redirect Message)*

Tab. 8.3 Typy a kódy ICMP zpráv

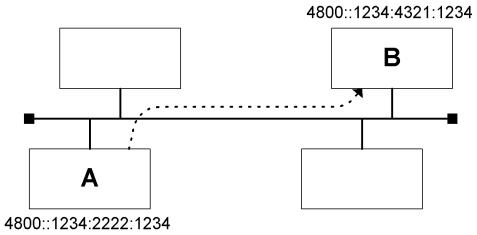
Typy ICMP zpráv se dělí na dva intervaly:

- Interval 0 až 127, což jsou chybové zprávy.
- Interval 128 až 255, což jsou informativní zprávy.

Smysl ICMP zpráv typů 1 až 129 uvedených v tab. 8.3 je obdobný ICMP zprávám v IPv4. Proto se zastavíme u zbývajících typů zpráv.

8.2.1 Překlad IP-adres na linkové adresy

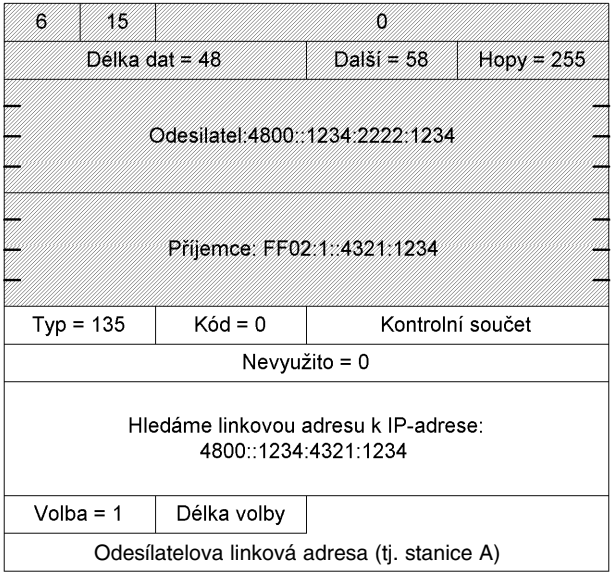
Na obrázku 8.12 chce stanice A odeslat stanici B IP-datagram. Jenže IP-datagram musí být vložen do linkového rámce. Stanice A sice zná IP-adresu stanice B (ta je 4800::1234:4321:123), avšak pro vytvoření linkového rámce potřebuje také linkovou adresu stanice B.



Obr. 8.12 Stanice A chce odeslat IP-datagram stanici B

Pro zjištění linkové adresy svého souseda na lokální síti slouží dvě ICMP zprávy:

1. **Žádost o linkovou adresu**, kterou stanice A žádá oběžníkem stanici B, aby stanici A poskytla svou linkovou adresu.
2. **Oznámení o linkové adrese**, kterou stanice B poskytuje stanici A svou linkovou adresu.



Obr. 8.13 Žádost o linkovou adresu

Žádost o linkovou adresu (viz obr. 8.13) má v poli maximální počet hopů uvedeno číslo 255. Pokud by tato žádost prošla přes směrovač (tj. přišla z jiné sítě), pak by toto pole bylo sníženo na hodnotu menší než 255 a příjemce by tak zjistil, že se jedná o zatoulanou žádost z jiné sítě, což mu znemožňuje na ni odpovědět. (Linkové adresy jsou jednoznačné pouze v rámci LAN).

Základní záhlaví má v poli příjemce uvedenu zvláštní adresu FF02:1::4321:1234. Jedná se o oběžník. Tato adresa se skládá ze dvou částí FF02:1:: a 4321:1234. První část specifikuje oběžník a druhá část jsou poslední čtyři bajty z IP-adresy příjemce. Každá stanice je totiž v IPng povinna poslouchat také oběžníky tohoto typu, tj. oběžníky které mají poslední čtyři bajty shodné s IP-adresou stanice. Tento mecha-

nismus je zaveden proto, aby se snížilo zatížení stanic na síti a žádostí se tak nemusely zabývat všechny stanice na síti, ale pouze ty, jejichž poslední čtyři bajty jsou shodné.

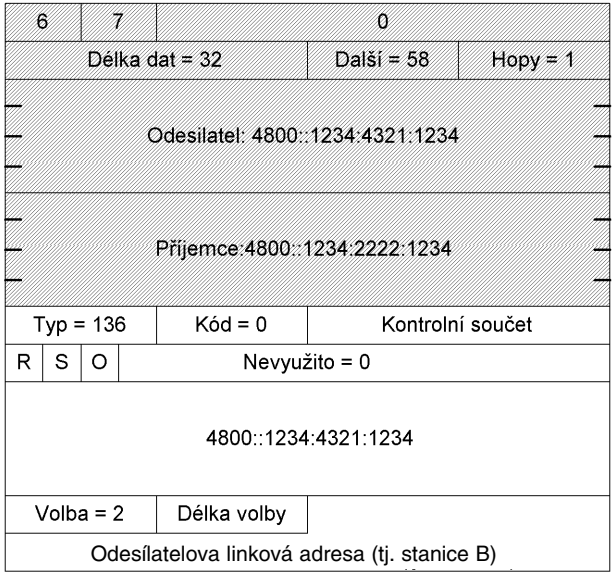
Pole typ v ICMP zprávě má hodnotu 135. Pole kód má hodnotu 0, ale směrovače mohou upozorňovat, že se jedná o směrovač nastavením pole kód na jedničku.

Tělo ICMP zprávy obsahuje IP-adresu stanice o jejíž linkovou adresu se žádá a dále obsahuje volbu, ve které je uvedena linková adresa žadatele (odesílatele). Volby se v ICMP zprávách zásadně skládají ze tří polí:

- jednobajtového pole volba obsahujícího identifikaci volby (např. odesílatelova linková adresa má identifikaci volby 1 v dotazu a volbu 2 v odpovědi),
- jednobajtového pole délka obsahující délku volby,
- vlastní volby.

Odpovědí na žádost je ICMP zpráva **oznámení o linkové adrese** (viz obr. 8.14). Příjemce si tuto odpověď uloží do paměti, aby nemusel s dalším odesílaným IP-datagramem znovu podstupovat martýrium žádosti o linkovou adresu svého souseda.

Základní záhlaví IP-datagramu obsahuje v poli hopy hodnotu 1, aby se zamezilo zatoulání odpovědi na jinou síť a v poli pro IP-adresu příjemce již není oběžník, ale IP-adresa žadatele.



Obr. 8.14 Oznámení o linkové adrese

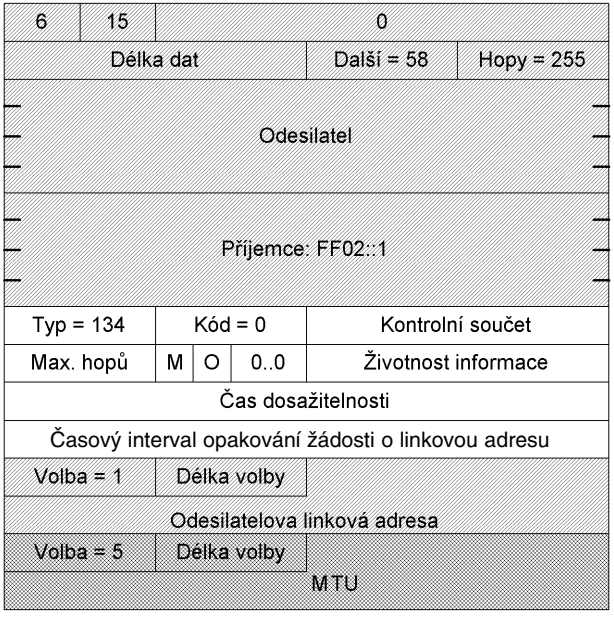
Tělo ICMP zprávy obsahuje tři tajuplné bity R, S a O, dále je v těle zprávy zopakována IP-adresa stanice o jejíž linkovou adresu se žádalo, a konečně volba o identifikaci 2 obsahuje požadovanou linkovou adresu.

Bit R je nastaven na jedničku, když odesílatel je směrovač, bit S je nastaven na jedničku jedná-li se o odpověď na žádost (tj. předcházela-li ICMP zpráva žádost o linkovou adresu) a konečně bit O je nastaven na jedničku, když odesílatel chce zdůraznit, že příjemce si má přepsat hodnoty uložené v paměti.

8.2.2 Zjištění adresy směrovače na LAN

Pokud stanici nestačí komunikace pouze na LAN, pak pro komunikaci mimo LAN musí znát adresu směrovače. Řečí směrovačů: stanice potřebuje získat položku *default* do své směrovací tabulky. Směrovače v pravidelných intervalech rozšiřují tuto informaci oběžníkem pro všechny počítače na LAN (FF02::1) pomocí ICMP zprávy oznámení o směrování – viz obr. 8.15.

Po přijetí zprávy oznámení o směrování si stanice vytvoří položku *default* do své směrovací tabulky jednoduše tak, že směr *default* bude ukazovat na odesílatele této zprávy.



Obr. 8.15 Oznámení o směrování

ICMP zpráva oznámení o směrování obsahuje následující pole:

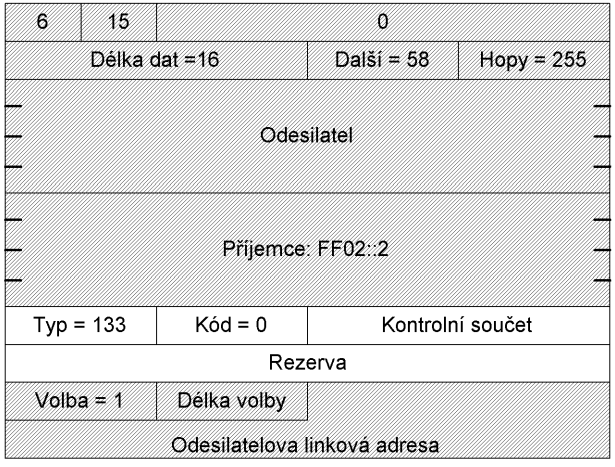
- Pole typ ICMP zprávy, které má hodnotu 134 a pole kód o hodnotě nula.
- Pomocí pole max.hopů je stanicím doporučována hodnota, kterou mají vyplňovat do pole počet hopů v základním záhlaví IP-datagramu.
- Bity M a O slouží pro protokoly vyšších vrstev zabývající se automatickou konfigurací stanice (např. protokol DHCP).
- Pole životnost informace (*Router Lifetime*) obsahuje čas ve vteřinách, po který by měla stanice položku *default* vytvořenou na základě této zprávy udržovat ve svých směrovacích tabulkách.

Hodnota nula specifikuje, že se žádná položka *default* ukazující na odesílatele tohoto datagramu nemá ve směrovacích tabulkách dále udržovat.

- Pole čas dosažitelnosti a pole časový interval opakování žádosti o linkovou adresu se týkají žádostí/odpovědí o linkovou adresu souseda (tj. předchozí kapitoly). Oba údaje se uvádí v milisekundách. Čas dosažitelnosti (*Reachability Timeout*) specifikuje čas, po který lze předpokládat, že soused je dosažitelný. Časový interval opakování žádosti o linkovou adresu je interval, po který se má udržovat v paměti položka získaná ze zprávy oznámení o linkové adrese.

Dále může ICMP zpráva oznámení o směrování také obsahovat některé volby. Na obr. 8.15 jsou uvedeny volby odesílatelova linková adresa a MTU. S volbou odesílatelova linková adresa jsme se setkali již u zprávy oznámení o linkové adrese. Zde má napomoci k tomu, aby stanice nemusela další ICMP zprávou požádat směrovač o jeho linkovou adresu. Volba MTU specifikuje maximální podporovanou velikost linkového rámce.

Stanice může také sama požádat směrovač o zaslání zprávy o směrování. Protokol ICMP má pro tento případ k dispozici ICMP zprávu žádost o směrování. Tuto zprávu (viz obr. 8.16) odesílá stanice oběžníkem pro všechny směrovače (FF02::2). Směrovač pak již neodpovídá oběžníkem pro všechny počítače, ale přímo jednoznačnou adresou tazatele pomocí zprávy oznámení o směrování.



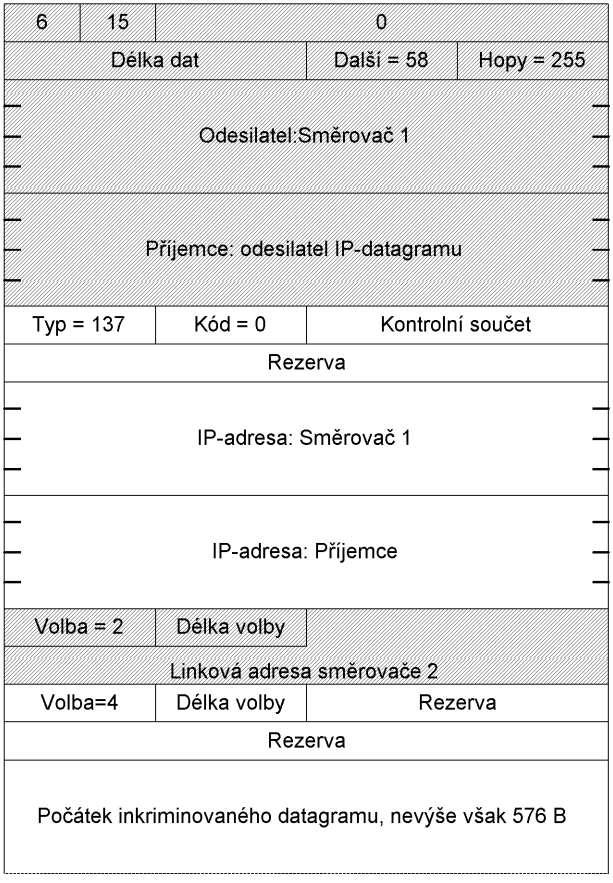
Obr. 8.16 ICMP zpráva Žádost o směrování

8.2.3 Změň směrování

Situace je znázorněna na obr. 5.11. Na LAN je více směrovačů. Odesílatel posílá IP-datagram příjemci. Jelikož ve směrovací tabulce nemá přímý směr přes směrovač 2, tak využije položku *default*, kterou získal např. z ICMP zprávy oznámení o směrování a která ukazuje na směrovač 1. Směrovač 1 přijme datagram, ale je nucen jej stejným rozhraním předat směrovači 2. Směrovač 1 to sice provede, ale neodpustí si o tom, ICMP zprávou změň směrování, informovat odesílatele. Odesílatel přijme zprávu změň směrování, ze které si vytvoří novou položku do své směrovací tabulky, která ukazuje příjemce přímo přes směrovač 2.

ICMP zpráva změň směrování je znázorněna na obr. 8.17. Na obrázku je také napsán hypotetický příkaz route, kterým by se vložila příslušná položka do směrovací tabulky odesílatele.

Zajímavé je též pole kód, které je 0 v případě, že adresátem ICMP zprávy je počítač a 1 v případě, že adresátem je směrovač.




```
"route add host Příjemce Směrovač2"
```

Obr. 8.17 Změň směrování

8.3 IP-adresa

IP-adresa je v protokolu IPng šestnáctibajtová (128 bitů). Rozeznáváme tři typy adres:

- Jednoznačná adresa síťového rozhraní (*Unicast*).
- *Anycast* – adresa skupiny síťových rozhraní, IP-datagram adresovaný adresou typu *anycast* bude doručen jednomu z těchto rozhraní.
- Oběžník (*Multicast*)

Neexistuje zde všeobecný oběžník (*Broadcast*).

8.3.1 Zápis adresy

Používají se tři zápisy IP-adresy:

- hhhh:hhhh:hhhh:hhhh:hhhh:hhhh:hhhh:hhhh, kde h je jedna šestnáctková číslice (0 až F) reprezentující 4 bity adresy.
  Příklad: ABCE:3:89AD:134:FEDC:E4D1:34:4321 (vedoucí nuly se nemusí uvádět)
- Zkrácený zápis pomocí zdvojené dvojtečky. Zdvojená dvojtečka se může v adrese vyskytnout pouze jednou. Zdvojená dvojtečka nahrazuje libovolné množství čtveřic nul.
  Příklad: Adresu 12A1:0:0:0:5:15:500C:44 je možné zkráceně zapsat jako 12A1::5:15:500C:44
  Adresu 1234:0:0:0:0:0:0:14 je možné zkráceně zapsat jako 1234::14
  Smyčku, tj. adresu 0:0:0:0:0:0:0:1 je možné zkráceně zapsat jako ::1
- hhhh:hhhh:hhhh:hhhh:hhhh:hhhh:d.d.d.d, kde poslední čtveřice je vyjádřena obdobně jako u IP-adresy verze 4, tj. každý bajt je vyjádřen desítkovou číslicí. Tato forma zápisu je vhodná v prostředí, kde se budou společně používat IPv4 a IPv6.
  Příklady:
  ::195.47.103.12
  12::A54:147.123.25.4

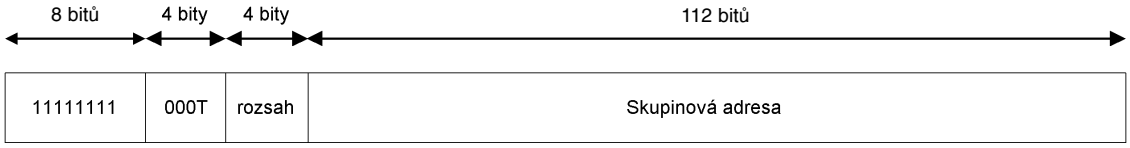
Adresy sítí se zapisují obdobně jako u IPv4 jako prefix následovaný lomítkem a počtem bitů tvořících adresu. Např. 80:1::1/64.

0:0:0:0:0:0:0:0	Nespecifikovaná adresa (nepoužívá se)
0:0:0:0:0:0:0:1	Smyčka (Loopback)
…	
010/3	Jednoznačné adresy přidělované poskytovatelům Internetu
010 10000/8	IANA
010 01000/8	RIPE (Evropa) $1001000_2=48_{16}$
010 11000/8	ARIN (Amerika)
010 00100/8	APNIC (Asie a Pacifik)
010 11111/8	Testovací blok adres (viz RFC-1897)
…	
1111 1111/8	Oběžníky (Multicasts)

Tab. 8.4 Některé části adresního prostoru IPng

8.3.2 Oběžníky

Oběžníky mají prefix obsahující v prvním bajtu samé binární jedničky, tj. šestnáctkově FF (viz obr. 1.18).



Obr. 8.18 Adresa oběžníku

Druhý bajt je rozdělen na poloviny. První polovina má významný pouze bit T, který pokud má hodnotu 0, pak je oběžník přiřazen trvale, pokud má hodnotu 1, pak se jedná o dočasné přiřazení.

Druhá polovina druhého bajtu specifikuje rozsah skupiny tvořící oběžník. Nabývá např. hodnot:

1 – oběžník v rámci lokálního uzlu,

2 – oběžník v rámci LAN,

5 – oběžník v rámci sítě,

8 – oběžník v rámci firmy,

E – globální oběžník.

Existují vyhrazené oběžníky, např.:

FFxx::1 – oběžník pro všechny stanice (počítače i směrovače)

FFxx::2 – oběžník pro všechny směrovače

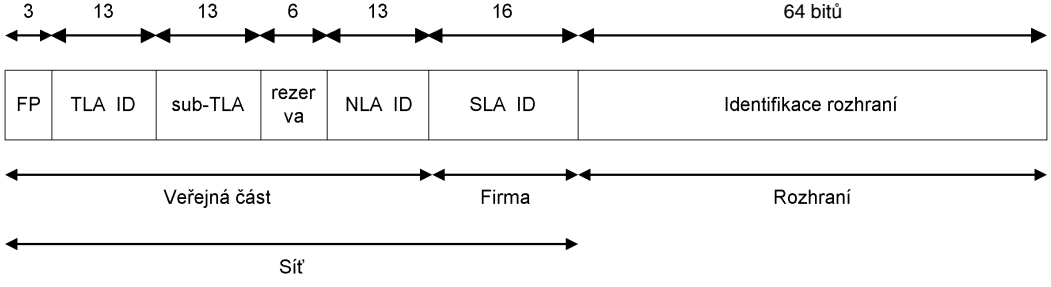
FFxx::9 – oběžník pro všechny směrovače provozující protokol RIP

atd.

Konkrétně FF02::2 je oběžník určený všem směrovačům na LAN.

8.3.3 Jednoznačné adresy

RFC-2373 specifikuje alokaci adres. Avšak nyní se ukazuje, že se bude používat mírně pozměněné schéma viz RIPE-196 (IPv6 Assignment and Allocation Policy Document), který IP-adresu rozděluje do částí znázorněných na obr. 8.19.



Obr. 8.19 IP-adresa verze 6

Význam jednotlivých polí v IPv6-adrese je:

FP — první tři bity mají pro jednoznačnou alokovanou adresu hodnotu 010.

TLA ID — (*Top-Level Agregation Identifiers*) – těchto 13 bitů je spravováno IANA (nejvyšší autorita Internetu), která přiděluje tento prostor regionálním IR (*Internet Registry*) – pro Evropu je IR organizace RIPE se sídlem v Amsterodamu. RIPE má přiděleno prvních 8 bitů: 010 01000, tj. RIPE spravuje všechny adresy, které mají v prvním bajtu hodnotu $1001000_2=48_{16}$.

sub-TLA — z těchto 13 bitů může RIPE přidělovat organizacím označovaným jako TLA registry, které budou přidělovat adresy poskytovatelům Internetu. Hodnota polí FP + TLA ID + sub TLA je přidělena konkrétnímu TLA registry.

NLA ID — (*Next Level Agregation*) – těchto 13 bitů identifikuje konkrétního poskytovatele Internetu.

SLA ID — 16 bitů, ze kterých přiděluje poskytovatel adresní prostor firmě. Nejmenší část adresního prostoru, která může být firmě přidělena je /64.

Otázkou je kdo bude TLA registry. Předpokládá se, že to budou velcí poskytovatelé Internetu. Avšak kdo je velkým poskytovatelem IPv6? Kritériem je aby měl alespoň k dalším třem TLA registry samostatné linky, na kterých si vyměňuje směrovací informace pomocí protokolu BGP. Na těchto linkách musí být komunikace IPv6 (IPv4 linky se nepočítají). Dále musí prokázat dostatečné množství zákazníků. V případě, že se jedná o začínající firmu, která zákazníky teprve očekává, pak se musí v Amsterodamu předložit projekt (stačí jej poslat elektronickou poštou). Projekt RIPE posoudí a rozhodne. (Proti rozhodnutí RIPE není odvolání, v případě zamítnutí je nutné projekt opravit či nejčastěji dopracovat. Zamítnutí je třeba považovat za osobní selhání, protože jste obtěžovali svým nedobře vypracovaným projektem tým, který má na starosti provoz 1/3 světového Internetu.)

8.4 DNS

Rozšíření a změny DNS spojené s IPv6 musí být kompatibilní s existujícími aplikacemi. Rozšířením DNS pro IPng se věnuje RFC-1886.

Záznam typu AAAA

IPv4 používá pro překlad jména na IP-adresu záznam typu A. Pro IPv6 je zaveden obdobný záznam typu AAAA. Rozdíl spočívá v tom, že záznam typu AAAA má v poli IP-adresa nikoliv čtyřbajtovou IPv4-adresu, ale 32bajtovou adresu IPv6.

Reverzní doména IP6.INTA

Pro reverzní překlad je zavedena nová doména IP6.INT (obdoba domény IN-ADDR.ARPA pro IPv4). IPv6-adresa je prvkem domény IP6.INT. Zajímavý je zápis prvku domény IP6.INT. Jednotlivé bajty se opět zapisují „pozpátku", ale nikoliv celé bajty, ale poloviny bajtů. Polovina bajtu je reprezentovaná jednou šestnáctkovou číslicí. Jednotlivé šestnáctkové číslice se oddělují tečkou (tj. oddělovačem v doménovém jméně).

Příklad: IP-adresa 4321::1:2:3:4:567:89AB bude jako prvek domény IP6.INT zapsána:
B.A.9.8.7.6.5.0.4.0.0.0.3.0.0.0.2.0.0.0.1.0.0.0.0.0.0.0.1.2.3.4.IP6.INT.
(IP-adresa 4321::1:2:3:4:567:89AB je jen zkráceným zápisem adresy
4321:0000:0001:0002:0003:0004:0567:89AB).

9

Protokol TCP *(Transmision Control Protocol)*

Protokol TCP je proti protokolu IP protokolem vyšší vrstvy. První otázkou každého začátečníka vždy je: „Proč jsou třeba dva protokoly".

Zatímco protokol IP přepravuje data mezi libovolnými počítači v Internetu, tak protokol TCP dopravuje data mezi dvěma konkrétními aplikacemi běžícími na těchto počítačích. Pro dopravu dat mezi počítači se využívá protokol IP. Protokol IP adresuje IP-adresou pouze síťové rozhraní počítače. Pokud bychom použili přirovnání k běžnému poštovnímu styku, pak IP-adresa odpovídá adrese domu a port (adresa v protokolu TCP) pak odpovídá jménu konkrétního obyvatele domu.

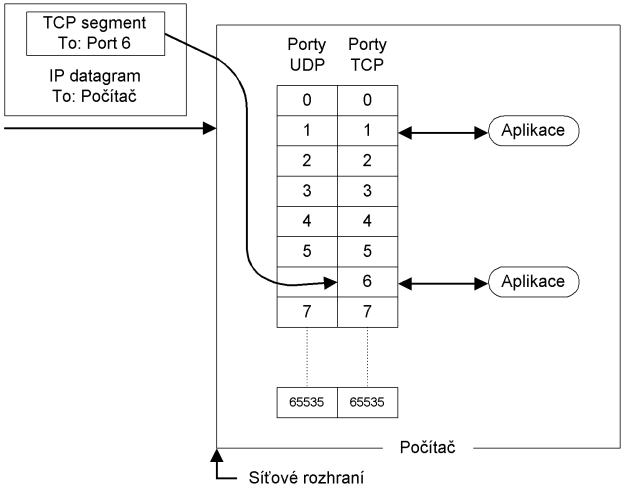
Protokol TCP je spojovanou službou (*connection oriented*), tj. službou která mezi dvěma aplikacemi naváže spojení – vytvoří na dobu spojení virtuální okruh. Tento okruh je plně duplexní (data se přenášejí současně na sobě nezávisle oběma směry). Přenášené bajty jsou číslovány. Ztracená nebo poškozená data jsou znovu vyžádána. Integrita přenášených dat je zabezpečena kontrolním součtem.

Jinými slovy aplikace používající protokol TCP si nemusí dělat starosti s tím, zdali náhodou nebyla nějaká data během přenosu ztracena nebo díky chybě přenosu modifikována. Toto zabezpečení je účinné pouze proti poruchám technických prostředků. Neklade si za cíl zabezpečit data proti inteligentním útočníkům, kteří mohou data modifikovat a současně také přepočítat kontrolní součet. Ochranou přenášených dat proti takovýmto cíleným útokům se v rodině protokolů TCP/IP zabývají např. protokoly SSL, S/MIME.

Konce spojení ("odesílatel" a „adresát") jsou určeny tzv. číslem portu. Toto číslo je dvojbajtové, takže může nabývat hodnot 0 až 65535. U čísel portů se často vyjadřuje okolnost, že se jedná o porty protokolu TCP tím, že se za číslo napíše lomítko a název protokolu (tcp). Pro protokol UDP je jiná sada portů než pro protokol TCP (též 0 až 65535), tj. např. port 53/tcp nemá nic společného s portem 53/udp.

Cílová aplikace je v Internetu adresována (jednoznačně určena) IP-adresou, číslem portu a použitým protokolem (TCP nebo UDP). Protokol IP dopraví IP-datagram na konkrétní počítač. Na tomto počítači běží jednotlivé aplikace. Podle čísla cílového portu operační systém pozná které aplikaci má TCP-segment doručit.

Porty připomínají poštovní schránky v paneláku, jak je znázorněno na obr. 9.1.



Obr. 9.1 Porty TCP a UDP

Základní jednotkou přenosu v protokolu TCP je TCP segment. Někdy se také říká TCP paket. Proč segment? Aplikace běžící na jednom počítači posílá protokolem TCP data aplikaci běžící na jiném počítači. Aplikace potřebuje přenést např. soubor velký 2 GB. Jelikož TCP segmenty jsou baleny do IP datagramů, který má pole délka dlouhé 16 bitů, tak TCP segment může být dlouhý maximálně 65535 minus délka TCP-záhlaví. Přenášené 2 GB dat musí být rozděleny na segmenty, které se vkládají do TCP paketu – přeneseně se pak místo TCP paket říká TCP segment.

TCP segment se vkládá do IP-datagramu. IP-datagram se vkládá do linkového rámce. Použije-li se příliš velký TCP-segment, který se celý vloží do velkého IP-datagramu, který je větší než maximální velikost přenášeného linkového rámce (MTU), pak IP protokol musí provést fragmentaci IP-datagramu (viz obr 9.2).

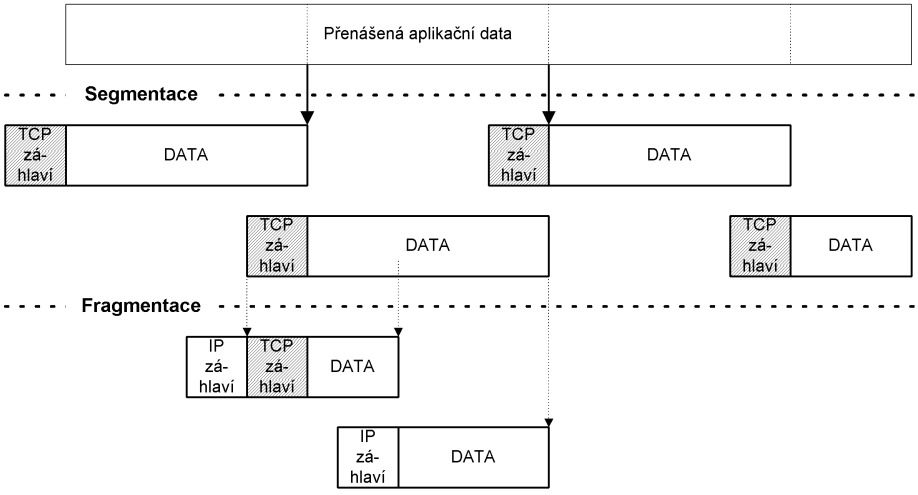
Fragmentace zvyšuje režii, proto je cílem vytvářet segmenty takové velikosti, aby fragmentace nebyla nutná.

9.1 TCP segment

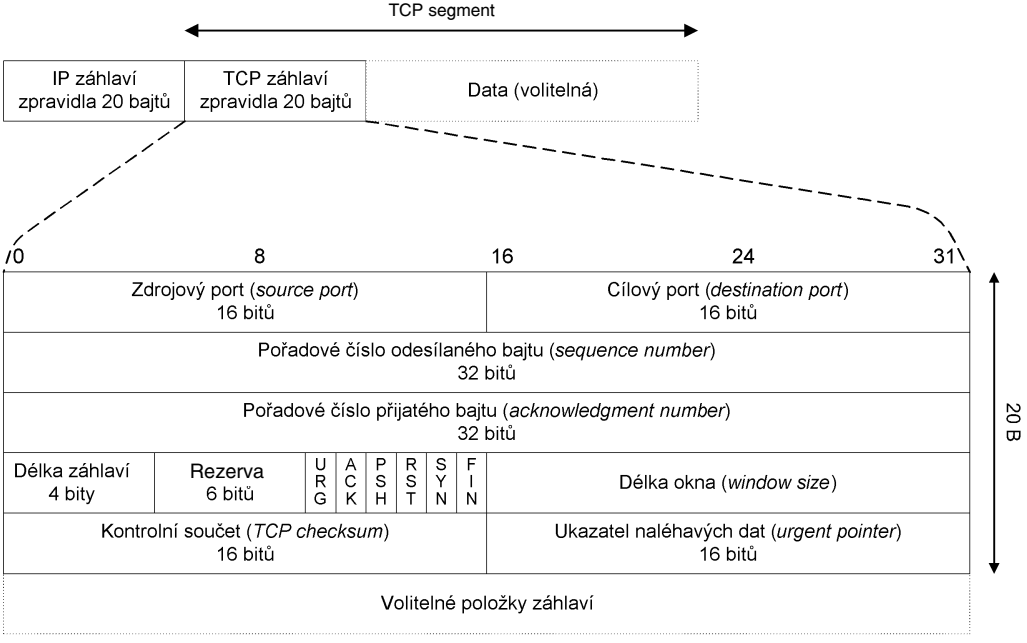
TCP segment má strukturu uvedenou na obr. 9.3.

Zdrojový port (*source port*) je port odesílatele TCP segmentu, **cílový port** (*destination port*) je portem adresáta TCP segmentu. Pětice: zdrojový port, cílový port, zdrojová IP-adresa, cílová IP-adresa a protokol (TCP) jednoznačně identifikuje v daném okamžiku spojení v Internetu.

TCP segment je část z toku dat tekoucích od odesílatele k příjemci. **Pořadové číslo odesílaného bajtu** je pořadové číslo prvního bajtu TCP segmentu v toku dat od odesílatele k příjemci (TCP segment nese bajty od **pořadového čísla odesílaného bajtu** až do délky segmentu). Tok dat v opačném směru má samostatné (jiné) číslování svých dat. Jelikož pořadové číslo odesílaného bajtu je 32 bitů dlouhé, tak po dosažení hodnoty 2^{32}-1 nabude cyklicky opět hodnoty 0. Číslování obecně nezačíná od nu-



Obr. 9.2 Segmentace a fragmentace



Obr. 9.3 TCP segment

ly (ani od nějaké určené konstanty), ale číslování by mělo začínat od náhodně zvoleného čísla. Vždy když je nastaven příznak SYN, tak operační systém odesílatele začíná znovu číslovat, tj. vygeneruje startovací pořadové číslo odesílaného bajtu, tzv. ISN (*Initial Sequence Number).*

Naopak **pořadové číslo přijatého bajtu** vyjadřuje číslo následujícího bajtu, který je příjemce připraven přijmout, tj. příjemce potvrzuje, že správně přijal vše až do pořadového čísla přijatého bajtu mínus jedna.

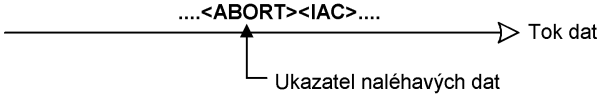
Délka záhlaví vyjadřuje délku záhlaví TCP segmentu v násobcích 32 bitů (4 bajtů) – podobně jako u IP-záhlaví.

Délka okna vyjadřuje přírůstek pořadového čísla přijatého bajtu, který bude příjemcem ještě akceptován (viz kapitola 9.7).

Ukazatel naléhavých dat může být nastaven pouze v případě, že je nastaven příznak URG. Přičte-li se tento ukazatel k pořadovému číslu odesílaného bajtu, pak ukazuje na konec úseku naléhavých dat. Odesílatel si přeje, aby příjemce tato naléhavá data přednostně zpracoval. Tento mechanismus používá např. protokol Telnet.

Aplikační protokol Telnet přenáší data mezi terminálem a serverem. Kromě běžných aplikačních dat, může tok dat obsahovat i příkaz IAC, tj. „následující data interpretuj jako příkaz protokolu Telnet" (*Interpret As Command*). Příkaz IAC začíná znakem IAC (desítkově 255) následovaný příslušným příkazem. Jako příkaz může být např. ABORT (zruš proces), který je reprezentován bajtem o desítkové hodnotě 238.

Příkazem ABORT signalizuje uživatel žádost o zrušení procesu. Uživatel například odeslal na server velké množství dat, která čekají ve vyrovnávací paměti serveru na zpracování. Pokud uživatel chce proces ukončit bez čekání na ukončení zpracování dat a nezpracovaná data zahodit, pak vyšle příkaz ABORT. Pokud by server zpracovával veškerá data sekvenčně, pak by se příkaz ABORT vykonal až po zpracování všech dat. Uživatel však chce proces ukončit okamžitě. V TCP segmentu nesoucím příkaz ABORT se nastaví příznak URG a vyplní se ukazatel naléhavých dat ukazující na příkaz ABORT.



Obr. 9.4 Naléhavá data

Server podle příznaku URG zjistí, že TCP segment obsahuje naléhavá data, přičtením ukazatele naléhavých dat k pořadovému číslu odeslaného bajtu zjistí konec naléhavých dat. Nyní začne procházet vyrovnávací paměť od konce naléhavých dat směrem k počátku vyrovnávací paměti až narazí na bajt obsahující <IAC>. Nyní již má zjištěna celá naléhavá data a může je začít interpretovat, tj. v našem případě zrušit proces.

Využití ukazatele naléhavých dat závisí na tvůrci aplikace. Tvůrce většinou vyvíjí jak software pro odesílatele, tak i software pro příjemce. Většina aplikací tento mechanismus nevyužívá – programy telnet a rlogin jsou spíše výjimky. Existují však i aplikace, jež ukazatel naléhavých dat využívají, ale ten ukazuje na počátek naléhavých dat (nikoliv na konec, jak určuje norma).

V poli příznaků mohou být nastaveny následující příznaky:

- **URG** – TCP segment nese naléhavá data.
- **ACK** – TCP segment má platné pole „Pořadové číslo přijatého bajtu" (nastaven ve všech segmentech kromě prvního segmentu, kterým klient navazuje spojení).
- **PSH** – Zpravidla se používá k signalizaci, že TCP segment nese aplikační data, příjemce má tato data předávat aplikaci. Použití tohoto příznaku není ustáleno.
- **RST** – Odmítnutí TCP spojení.
- **SYN** – Odesílatel začíná s novou sekvencí číslování, tj. TCP segment nese pořadové číslo prvního odesílaného bajtu (ISN).
- **FIN** – odesílatel ukončil odesílání dat. Pokud bychom použili přirovnání k práci se souborem, pak příznak FIN odpovídá konci souboru (EOF). Přijetí TCP segmentu s příznakem FIN neznamená, že v opačném směru není dále možný přenos dat. Jelikož protokol TCP vytváří plně duplexní spojení, tak příznak FIN způsobí jen uzavření přenosu dat v jednom směru. V tomto směru už dále nebudou odesílány TCP segmenty obsahující příznak PSH (nepočítaje v to případné opakování přenosu).

V dalším textu budeme kombinaci nastavených příznaků zapisovat podle prvních písmen z názvu příznaku. Pokud je příznak nenastaven, pak místo něj napíšeme tečku. Např. skutečnost, že TCP segment má nastaveny příznaky ACK a FIN a ostatní příznaky nenastaveny, zapíšeme: .A…F .

Kontrolní součet IP-záhlaví se počítá pouze ze samotného IP-záhlaví. Z hlediska zabezpečení integrity přenášených dat je důležitý kontrolní součet v záhlaví TCP-segmentu počítaný i z přenášených dat. Tento kontrolní součet se počítá nejen ze samotného TCP segmentu, ale i z některých položek IP-záhlaví. Kontrolní součet vyžaduje sudý počet bajtů, proto v případě lichého počtu se data fiktivně doplní jedním bajtem na konci.

Kontrolní součet se počítá z polí znázorněných na obrázku 9.5. Tato struktura slouží pouze pro výpočet kontrolního součtu – v každém případě se žádná takováto struktura nepřenáší Internetem! Někdy se tato struktura označuje jako pseudozáhlaví.

Na závěr si ještě na obr. 9.6 prohlédněte společně znázorněná IP i TCP záhlaví.

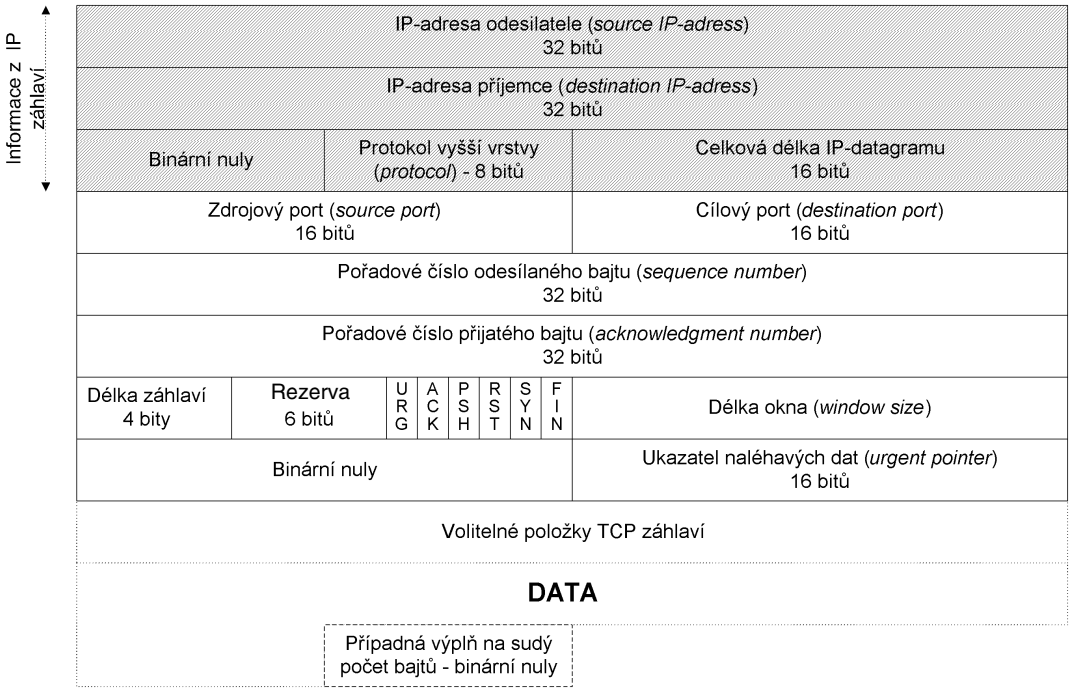
9.2 Volitelné položky TCP záhlaví

Povinné položky TCP záhlaví tvoří 20 B. Za povinnými položkami následují volitelné položky.

Volitelná položka se skládá z typu volitelné položky, délky volitelné položky a hodnoty. Délka TCP záhlaví musí být dělitelná čtyřmi. V případě, že délka záhlaví by nebyla dělitelná čtyřmi, pak se záhlaví doplňuje prázdnou volitelnou položkou – NOP.

Jelikož pole délka záhlaví je pouze 4 bity dlouhé, tak toto pole může nabývat maximálně hodnoty $1111_2 = 15_{10}$. Délka záhlaví se udává v násobcích čtyř, tudíž záhlaví může být dlouhé maximálně 15x4=60 bajtů. Povinné položky zaberou 20 bajtů, takže na volitelné zbývá nejvýše 40 bajtů.

Některé volby TCP záhlaví včetně jejich struktury jsou uvedeny na obrázku 9.7.



Obr. 9.5 Pole ze kterých se počítá kontrolní součet TCP záhlaví

9.3 Příklad výpisu TCP segmentu z Network Monitoru

Následující TCP segment má nastaven pouze příznak SYN. Segment má uvedenu jednu volitelnou položku záhlaví, kterou je maximální velkost přijímaného segmentu (MSS), tato položka je nastavena na hodnotu 1460.

```
+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
  IP: ID = 0x4030; Proto = TCP; Len: 44
      IP: Version = 4 (0x4)
      IP: Header Length = 20 (0x14)
    + IP: Service Type = 0 (0x0)
      IP: Total Length = 44 (0x2C)
      IP: Identification = 16432 (0x4030)
    + IP: Flags Summary = 2 (0x2)
      IP: Fragment Offset = 0 (0x0) bytes
      IP: Time to Live = 128 (0x80)
      IP: Protocol = TCP - Transmission Control
      IP: Checksum = 0x63DF
      IP: Source Address = 194.149.104.198
```

```
      IP: Destination Address = 194.149.104.203
      IP: Data: Number of data bytes remaining = 24 (0x0018)
  TCP: ....S., len:    4, seq: 145165778-145165781, ack: 0, win: 8192, src: 1458  dst:
4433
      TCP: Source Port = 0x05B2
      TCP: Destination Port = 0x1151
      TCP: Sequence Number = 145165778 (0x8A70DD2)
      TCP: Acknowledgement Number = 0 (0x0)
      TCP: Data Offset = 24 (0x18)
      TCP: Reserved = 0 (0x0000)
      TCP: Flags = 0x02 : ....S.
          TCP: ..0..... = No urgent data
          TCP: ...0.... = Acknowledgement field not significant
          TCP: ....0... = No Push function
          TCP: .....0.. = No Reset
          TCP: ......1. = Synchronize sequence numbers
          TCP: .......0 = No Fin
      TCP: Window = 8192 (0x2000)
      TCP: Checksum = 0xF3ED
      TCP: Urgent Pointer = 0 (0x0)
      TCP: Options
              TCP: Option Kind (Maximum Segment Size) = 2 (0x2)
              TCP: Option Length = 4 (0x4)
              TCP: Option Value = 1460 (0x5B4)
```

9.4 Navázání a ukončení spojení protokolem TCP

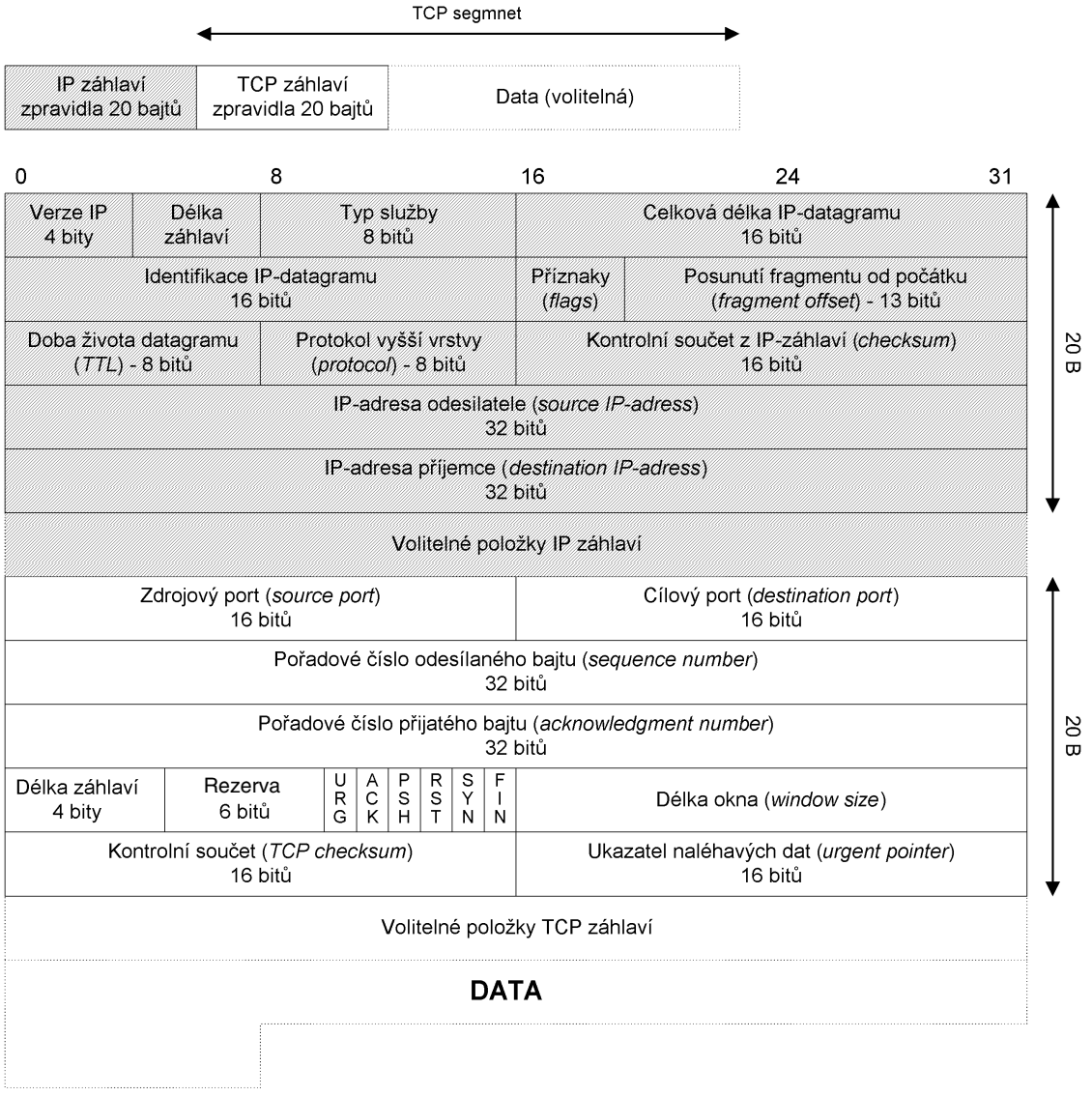
Jádrem protokolu IP byl zejména popis IP-datagramu. Jelikož je protokol IP datagramovou (nespojovanou službou), tak se protokol IP příliš nepotřeboval zatěžovat případy, kdy IP-datagram není doručen. IP protokol takové stavy může nanejvýš signalizovat protokolem ICMP. Signalizace protokolem ICMP je pouze dobrou vůlí protokolu IP. V praxi se setkáváme s mnoha případy, kdy signalizace pomocí protokolu ICMP je potlačována, protože je nežádoucí – např. z bezpečnostních důvodů.

Protokol TCP využívá k transportu dat Internetem protokol IP, avšak nad tímto protokolem zřizuje spojovanou službu. Musí řešit problémy navázání a ukončení spojení, potvrzování přijatých dat, vyžádání ztracených dat, ale také problémy průchodnosti přenosové cesty. Popis TCP segmentu je tedy jen malou částí TCP protokolu, podstatně rozsáhlejší částí je popis výměny TCP segmentů (*handshaking*) mezi oběma konci TCP spojení.

9.4.1 Navazování spojení

Protokol TCP umožňuje jedné straně navazovat spojení. Druhá stana spojení buď akceptuje, nebo odmítne. Z hlediska aplikační vrstvy bude stranou navazující spojení klient a server ta strana, která spojení očekává. Existují i jiné aplikační modely než klient/server, kterým protokol TCP zabezpečuje přenos dat, přesto budeme v zájmu usnadnění výkladu pojem klient používat pro stranu iniciující spojení a termín server pro stranu očekávající spojení.

Zajímavostí je, že protokol TCP umožňuje, aby obě strany navazovaly spojení i současně. V praxi jsme se však s využitím této možnosti nesetkali, proto nadále budeme popisovat jen případ, kdy jedna strana navazuje spojení, které druhá strana očekává.

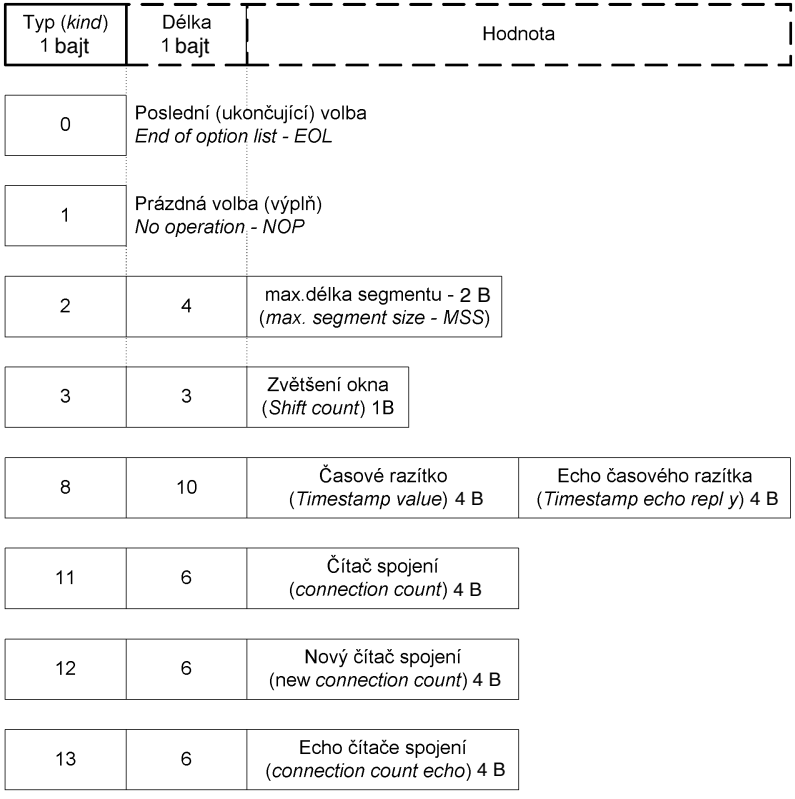


Obr. 9.6 IP a TCP záhlaví

Klient si přeje např. navázat spojení se serverem běžícím na počítači „Server" na portu 4433 (v praxi bychom napsali, že klient si přeje navázat spojení se serverem „Server:4433"). Klient pro spojení použije port 1458. Zatímco port serveru musí být předem všeobecně znám (aby o něm klienti věděli), tak s portem klienta je to zpravidla jinak. Je sice také možné aby klient vždy používal pevný port – v našem případě 1458. Běžnější však je, že klient na konkrétním čísle portu netrvá. Požádá správu volných portů svého operačního systému o přidělení nějakého volného portu na dobu spojení. Předpokládej-

me, že operační systém klienta klientovi přidělil port 1458. Operační systém takto přiděluje porty větší než 1023. Tyto porty se označují jako klientské či neprivilegované. Na rozdíl od portů menších než 1024, o jejichž použití může žádat pouze privilegovaný uživatel (např. v operačním systému UNIX uživatel root). Pro náš výklad je však jedno, zdali klient trval na přidělení konkrétního portu 1458 nebo chtěl nějaký port a operační systém mu přidělil zrovna port 1458.

Porty menší než 1024 se označují jako privilegované. Tyto porty používají nejčastěji servery, protože servery zpravidla spouští privilegovaný uživatel nebo jsou spouštěny při startu operačního systému jako by je spouštěl privilegovaný uživatel. Náš server nepoužívá privilegovaný port, používá port 4433 (4433>1023). Takový server může spustit běžný uživatel operačního systému. (Pochopitelně pokud je příslušný port volný, tj. nemá jej alokován jiný proces.)

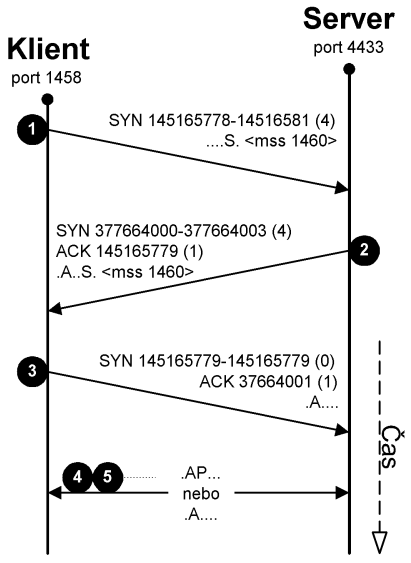


Servery používají všeobecně známá čísla portů. Tato čísla přiděluje organizace IANA tvůrcům serverů. Na http://www.iana.org lze získat přehled přidělených čísel portů.

klient začíná navazovat spojení odesláním prvního TCP segmentu ❶ (viz obr. 9.8), kde je port odesílatele 1458 a port příjemce 4433. Tento TCP segment klient vloží do IP-datagramu jehož IP-adresa odesílatele bude „Klient" a IP-adresa příjemce „Server". To je vcelku jasné. Jak však budou vypadat ostatní pole TCP segmentu?

Klient vygeneruje náhodné číslo v intervalu 0 až 2^{32}-1, které použije jako startovací pořadové číslo odesílaného bajtu (tzv. ISN). V našem případě bylo vytvořeno ISN=145165778. Skutečnost, že klient právě vytvořil startovací pořadové číslo odesílaného bajtu vyznačí v TCP segmentu nastavením příznaku SYN (….S.). Během spojení již pořadové číslo odesílaného bajtu bude vždy vyjadřovat číslo odesílaného bajtu, tj. nemůže být znovu vygenerováno. Tj. klient nemůže v žádném dalším TCP segmentu během tohoto spojení nastavit příznak SYN. Segment ❶ je prvním segmentem v TCP komunikaci, proto nemůže potvrzovat žádná přijatá data. Pole pořadové číslo přijatého bajtu nemá platný význam (bývá vyplněno binárními nulami) a nemůže být tedy ani nastaven příznak ACK (příznak ACK je nastaven ve všech dalších TCP segmentech až do ukončení spojení). TCP segment s nastaveným příznakem SYN a nenastaveným příznakem ACK je velice zvláštním segmentem. Tato kombinace nastaveného příznaku SYN a nenastaveného příznaku ACK je specifická pro první TCP segment spojení. Pokud se chce klientům zamezit v navázání spojení nějakým směrem, pak stačí v tomto směru odfiltrovat všechny TCP segmenty s nastaveným příznakem SYN a klient (útočník) nemá šanci. Tento mechanismus se též často využívá pro ochranu intranetů od Internetu.

Obr. 9.8
Navazování spojení



Součástí segmentů ❶ a ❷ byla volitelná položka záhlaví TCP segmentu MSS (*Maximum segment size*). Tato položka oznamuje druhé straně maximální délku datové části TCP segmentu jakou si přeje přijímat (aby se pokud možno zamezilo fragmentaci IP-datagramu). Tato volba se může vyskytovat pouze v TCP segmentech s nastaveným příznakem SYN (tj. v prvních dvou segmentech).

Implicitně se MSS používá 536 bajtů. Tato hodnota je používána pro spojení mimo lokální síť (přes WAN). Pro náš příklad jsme použili spojení v rámci lokální sítě protokolem Ethernet II. Pro linkový protokol Ethernet II je maximální délka datové části rámce rovna 1500. Pokud od 1500 odečteme 20 pro IP-záhlaví a dalších 20 pro TCP záhlaví, pak dojdeme k hodnotě z našeho příkladu (1460).

Pokud bychom na linkové vrstvě použili Ethernet ISO 8802-3, pak bychom museli ještě odečíst další 3 bajty na záhlaví dle ISO 8802-2 a dalších 5 bajtů na záhlaví protokolu SNAP, takže bychom mohli nabízet pouze MSS=1452.

Jelikož jsme pro výpis použili MS Network Monitor, tak ve výpisu segmentu ❶ se jeví jako bychom odesílali segmentem ❶ bajty o pořadovém čísle 145165778 až 14516581, tj. jako kdybychom odesílali čtyři bajty dat (pro přehlednost jsme na obrázku počet odesílaných bajtů spočítali do kulatých závorek). První tři segmenty přitom žádna data nenesou. Segment ❶ obsahuje pouze čtyři bajty dlouhou volitelnou položku TCP záhlaví specifikující maximální délku přijímaných segmentů <mss 1460>. MS Network Monitor přičítá délku volitelných položek záhlaví k délce dat. Volitelné položky záhlaví budeme často uvádět ve špičatých závorkách.

Výpis MS Network Monitoru (vypisujeme pouze sumární řádek protokolu TCP):

❶. segment Klient → Server

```
TCP: ....S., len: 4, seq: 145165778-145165781, ack:    0, win: 8192, src: 1458  dst: 4433
```

❷. segment Server → Klient

```
TCP: .A..S., len: 4, seq: 377664000-377664003, ack: 145165779, win:33580, src: 4433  dst: 1458
```

Druhý segment již potvrzuje přijatá data – má nastaven příznak ACK. Potvrzuje jeden bajt přijatých dat. Z výpisu MS Network Monitoru by se mohlo zdát, že potvrzuje jeden bajt volitelných položek záhlaví – to však není pravda. Potvrzuje příznak SYN. Příznak SYN (podobně i příznak FIN) se jeví jako by byly tvořeny jedním bajtem. Ve skutečnosti je to způsobeno tím, že pořadové číslo potvrzovaného bajtu vyjadřuje číslo bajtu, který může odesílatel odeslat – tudíž odesílatel může odeslat ISN+1.

Od druhého segmentu budou všechny další segmenty potvrzovat přijatá data, tj. budou mít nastaven příznak ACK.

❸. segment Klient → Server

```
TCP: .A...., len: 0, seq: 145165779-145165779, ack: 377664001, win: 8760, src: 1458  dst: 4433
```

Třetí segment rovněž potvrzuje příznak SYN, takže jakoby potvrzuje jeden bajt. Třetí a další segment již nemůže nést volitelnou položku záhlaví MSS (maximální délka segmentu).

Třetím segmentem končí navazování spojení. Někdy se proto také „česky" říká, že protokol TCP potřebuje pro navázání spojení „třífázový handshaking" (pro ukončení spojení pak „čtyřfázový").

❹. segment Klient → Server

```
TCP: .AP..., len:84, seq: 145165779-145165862, ack: 377664001, win: 8760, src: 1458  dst: 4433
```

Asi jste překvapeni, že čtvrtý segment posílá klient serveru – určitě jste očekávali, že čtvrtý segment pošle server klientovi. Ono je to jedno, jakmile kterákoliv strana obdrží první ACK, tak může začít s vysíláním. Vždyť TCP je plně duplexní spoj. A náš klient byl už netrpělivý z toho, kolik má dat k odeslání. Skutečnost, že TCP segment nese aplikační data je vyjádřena nastavením příznaku PSH.

❺. segment Server → Klient

```
TCP: .AP..., len:79, seq: 377664001-377664079, ack: 145165863, win:33580, src: 4433  dst: 1458
Atd.
```

Ne všechny TCP segmenty musí nutně nést aplikační data, tj. mít nastaven příznak PSH (.AP…). Může se stát, že jeden konec spojení odesílá data, avšak druhý konec nemá momentálně žádná data k odeslání. I když druhý konec nemá co posílat, tak musí potvrzovat přijatá data. Takové potvrzování provádí TCP segmenty s nenastaveným příznakem PSH (.A….), tj. segmenty bez dat.

V každém okamžiku spojení je spojení v tzv. stavu. Při navazování spojení může být:

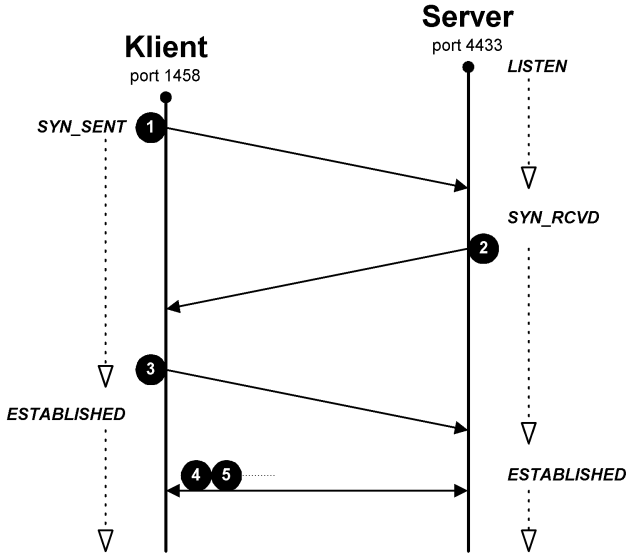
- Server ve stavu:
  - LISTEN – server je připraven na spojení s klienty.
  - SYN_RCVD – server přijal od klienta první TCP segment, tj. segment s příznakem SYN.
- Klient ve stavu:
  - SYN_SEND – klient odeslal první TCP segment, tj. segment s příznakem SYN.

Pokud se spojení naváže, pak klient i server přecházejí do stavu ESTABLISHED, tj. spojení navázáno. V tomto stavu si mohou oba konce současně předávat data.

Spojení a jejich stavy snadno vypíšete na vašem počítači (ať se jedná o UNIX či Windows) příkazem:

```
netstat -a
```

Obr. 9.9 Stavy při navazování spojení



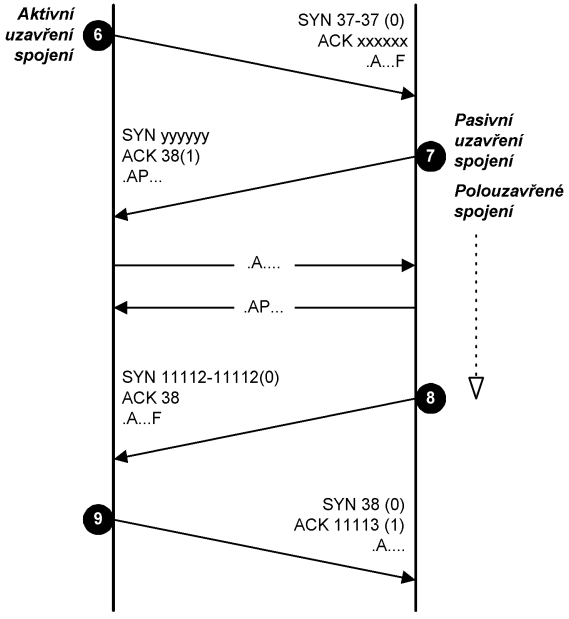
9.4.2 Ukončování spojení

Zatímco spojení navazoval v architektuře klient/server zpravidla klient, tak ukončit spojení může libovolná strana. Strana, která první odešle TCP segment s příznakem FIN (ukončení spojení) provádí tzv. aktivní ukončení spojení (*active close*), druhé straně nezbývá než provést pasivní ukončení spojení *(pasive close*). Teoreticky je možné i současné ukončení spojení.

Provede-li jedna strana aktivní ukončení spojení, pak již nemůže odesílat data (nemůže odeslat TCP segment s příznakem PSH). Druhá strana však může v odesílání dat pokračovat až do té doby, dokud neprovede sama ukončení spojení. Mezidobí od aktivního ukončení spojení do ukončení spojení nazýváme polouzavřeným spojením (*half close*). TCP segment s příznakem FIN je obdobou konce souboru (EOF).

Pro řádné uzavření spojení jsou nutné čtyři TCP segmenty. Příznak FIN se opět jako příznak SYN při navazování spojení potvrzuje jako by zabíral 1 B dat.

Obr. 9.10
Ukončování spojení



Segment ❻ startuje aktivní uzavření spojení nastavením příznaku FIN. Segment ❼ potvrzuje uzavření spojení druhou stranou, tj. provádí pasivní uzavření spojení. Většinou segment ❼ obsahuje též příznak FIN a dochází tím ke startu uzavření celého spojení. Náš obrázek však znázorňuje obecný případ (využívaný např. programem rsh), kdy segment ❼ příznak FIN neobsahuje, protože tato strana chce pokračovat ve spojení, tj. chce použít polouzavřené spojení pro přenos aplikačních dat. Strana, která spojení uzavřela již nemůže odesílat žádná data (jí odesílané segmenty nemohou obsahovat příznak PSH).

Stavy při ukončování spojení:

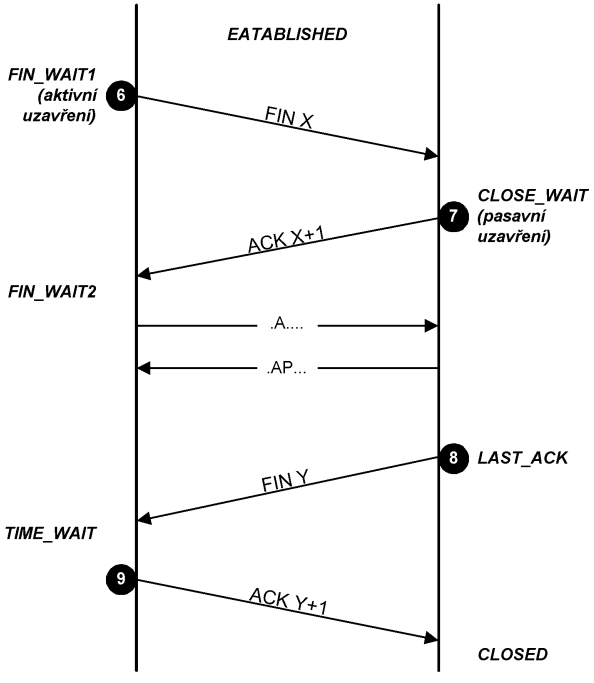
- FIN_WAIT1 – strana zjistila, že již všechna data odeslala (a potřebuje signalizovat konec souboru – EOF), tak v TCP segmentu nastaví příznak FIN, čímž signalizuje aktivní uzavření spojení segmentem ❻.
- CLOSE_WAIT – druhá strana obdržela aktivní uzavření spojení a nezbývá jí nic jiného než potvrdit segmentem ❼ přechod do pasivního uzavření spojení, kterému odpovídá stav CLOSE_WAIT.

- FIN_WAIT2 je stav poté, co strana obdrží potvrzení segmentem ❼ aktivního uzavření spojení od protějšku. Ve stavu FIN_WAIT2 strana zůstává do té doby, dokud protějšek nezašle TCP segment s příznakem FIN, tj. do přechodu do stavu TIME_WAIT. Pokud aplikace nepočítá s přenosem dat v polouzavřeném spojení, a spojení je nečinné po 11,25 minuty, pak operační systém automaticky změní stav spojení na CLOSED.
- LAST_ACK – druhá strana již odeslala všechna data a signalizuje úplné ukončení spojení segmentem ❽.
- TIME_WAIT – všechna data oběma směry již byla přenesena. Je nutné pouze potvrdit úplné uzavření spojení. Odesláním TCP segmentu ❾ je potvrzeno úplné ukončení spojení. Tento segment již není potvrzován, proto strana zůstává ve stavu TIME_WAIT po dobu 2 minut (některé implementace TCP/IP zkracují tuto dobu až na 30 s). Tato doba by totiž měla přibližně odpovídat dvojnásobku doby života TCP segmentu v síti. Důvod je prostý: segment ❾ se může v síti ztratit a druhá strana si může vyžádat jeho opakování. Pokud by však spojení bylo uzavřeno, pak opakování by již nebylo možné.
- CLOSED – druhá strana obdržela potvrzení úplného uzavření spojení a přechází do stavu CLOSED. Strana, která odeslala segment ❾ přechází do stavu CLOSED až po uplynutí zmíněného intervalu.

9.4.3 Odmítnutí spojení

Spojení se odmítá nastavením příznaku RST (*Reset*) v záhlaví TCP segmentu.

Obr. 9.11 Stavy při uzavírání spojení



Spojení je odmítáno v zásadě ve dvou případech:

- Klient požaduje spojení se serverem na portu, na kterém žádný server neběží. To je rozdíl oproti protokolu UDP. Pokud je zaslán UDP datagram na port, kde neběží žádný server, pak systém odpoví ICMP zprávou nedosažitelný port.
- Druhým případem je situace, kdy je odmítnuto dále pokračovat v již navázaném spojení. Zde lze rozlišit také dva případy:
  - Řádné ukončení spojení je poměrně dlouhou záležitostí (např. aplikace je nucena posečkávat ve stavu TIME_WAIT). Aplikace si po odeslání všech dat přeje ukončit spojení rychleji – použije odmítnutí spojení. V praxi se setkáváme s tím, že buď místo segmentu ❽ je odeslán segment s nastaveným příznakem RST. Nebo po segmentu ❾ následuje ještě potvrzení segmentu ❾ pomocí TCP segmentu s nastaveným příznakem RST. Takovýmto způsobem je však možné ukončit spojení až si obě strany vymění všechna data.
  - Jedna z komunikujících stran zjistí, že protějšek je nedůvěryhodný, pak okamžitě ukončuje spojení. To je případ například protokolu SSL. Protokol SSL zabezpečuje bezpečnou komunikaci (šifrovanou a autentizovanou). Pokud se nedokáže jedna strana autentizovat nebo použít tak silné kryptografické algoritmy jak vyžaduje druhá strana, tak je spojení okamžitě ukončeno nastavením příznaku RST.

9.5 Zjištění stavu spojení

Výpis všech spojení protokoly TCP a UDP lze získat pomocí příkazu netstat s parametrem –a.

```
$ netstat -an
```

Active Internet connections (including servers)

```
Proto Recv-Q Send-Q  Local Address          Foreign Address        (state)

tcp        0      0  194.149.105.18.22      194.149.103.204.24695  TIME_WAIT
tcp        0      0  194.149.105.18.3099    194.108.145.128.25     SYN_SENT
tcp        0  34472  194.149.105.18.3079    195.47.32.245.25       ESTABLISHED
tcp        0      0  *.22                   *.*                    LISTEN
tcp        0      0  *.25                   *.*                    LISTEN
tcp        0      0  *.53                   *.*                    LISTEN
udp        0      0  *.53                   *.*
udp        0      0  127.0.0.1.53           *.*
```

První dva řádky tvoří záhlaví výpisu. Význam jednotlivých sloupců:

- Sloupec `Proto` obsahuje název použitého protokolu (TCP nebo UDP).
- Sloupec `Recv-Q` vyjadřuje počet bajtů ve vstupní frontě spojení (čekajících na zpracování aplikací).
- Sloupec `Send-Q` vyjadřuje počet bajtů ve výstupní frontě (čekajících na odeslání).
- Sloupec `Local Address` obsahuje adresu lokálního síťového rozhraní tečkou odděleného od čísla lokálního portu. Servery čekající na spojení mohou mít na místo IP-adresy uvedenu hvězdičku. Hvězdička označuje, že server očekává spojení na všech svých síťových rozhraních.
- Sloupec `Foregin Address` obsahuje IP-adresu a port vzdáleného konce spojení. Hvězdičky vyznačují, že server očekává spojení z libovolné IP-adresy a libovolného portu.
- Sloupec `(state)` obsahuje stav spojení.

Pro servery mohou nastat následující kombinace:

V předchozím příkladu řádek

```
tcp        0      0  194.149.105.18.22      194.149.103.204.24695  TIME_WAIT
```

vyjadřuje, že server na lokálním počítači běžící na portu 22 (tj. program sshd) potvrdil úplné ukončení spojení (TIME_WAIT) s počítačem o IP-adrese 194.149.103.204. Klientovi byl pro toto spojení přiřazen port 24695.

Local Address	Foreign Address	(state)	
IP1.port1	IP2.port2	LISTEN	Server očekává na svém síťovém rozhraní IP1 spojení s konkrétním klientem o adrese IP2 a portu port2.
IP1.port1	IP2.port2	Kromě `LISTEN`	Server navazuje/má navázáno/ukončuje spojení s konkrétním klientem.
IP1.port1	*.*	`LISTEN`	Server očekává spojení na svém síťovém rozhraní IP1 (a žádném jiném) s libovolným klientem.
*.port1	*.*	`LISTEN`	Server očekává s libovolným klientem spojení na všech svých síťových rozhraních.

Řádek:

```
tcp        0      0  *.53                   *.*                    LISTEN
```

vyjadřuje, že na lokálním počítači na jeho všech síťových rozhraních čeká server na portu 53 (tj. program named) na spojení s libovolným klientem.

9.6 Technika zpoždění odpovědi

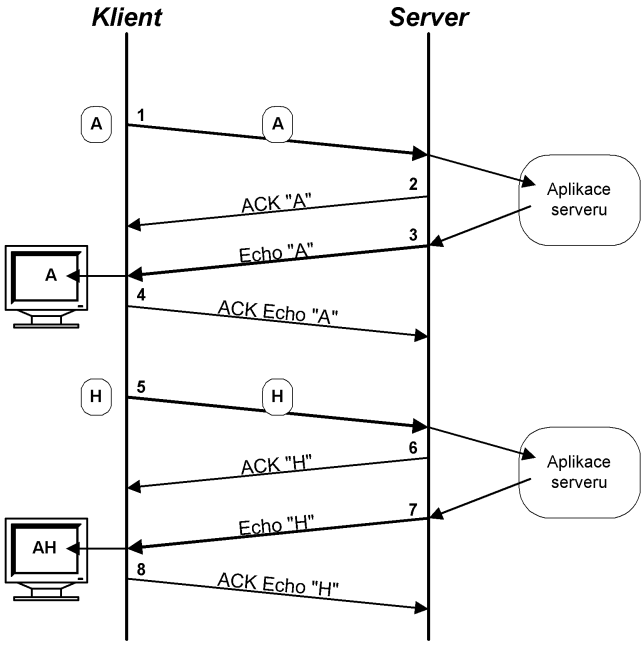
Interaktivní aplikace jako telnet či příkazový kanál FTP jsou komplikované tím, že jsou interaktivní. Tzn., že zmáčknete-li na klávesnici klávesu A, znak A se zabalí do TCP segmentu (20+1=21 B), TCP segment se vloží do IP-datagramu (20+21=41 B) a tento IP-datagram pak putuje Internetem jako segment (1) na obr. 9.12 až doputuje na server (vše se pochopitelně ještě vkládá do linkových rámců od směrovače ke směrovači). Server:

- Jednak potvrdí příjem znaku, tj. pokud nemá žádná data k odeslání, tak vyšle nedatový segment (40 B) – segment (2).
- Jednak předá znak A ke zpracování serveru, server musí vyslat znak A zpět – segment (3), aby klientův software zobrazil znak A na obrazovce klienta (provedl echo). Echo je nutné právě pro subjektivní pocit interaktivnosti aplikace. Klient přijme echo (znak A) a
  - zobrazí jej na obrazovce
  - potvrdí příjem echa serveru segmentem (4). Pokud nemá žádná další data k odeslání, pak potvrzení provede nedatovým TCP segmentem.

Pokud by takto aplikace pracovala, pak zmáčknutí jedné klávesy znamená přenést 81 B v každém směru (nezapočítávaje režii linkové vrstvy). 81 B se skládá z 41 B při odeslání znaku a 40 B při jejich potvrzení. Na následujícím obrázku je tato situace vyznačena pro stisk kláves A a H (začátek řetězce AHOJ).

Je vcelku pochopitelné, že je snaha objem přenášených dat zmenšit, čímž se snažíme zabránit ucpání přenosových cest. Myšlenka spočívá v tom, že potvrzování přijatých dat nebude probíhat okamžitě, ale se zpožděním. Během tohoto zpoždění se pak mohou objevit i další data k přenosu.

Obr. 9.12
Stisk kláves A a H



Princip spočívá v tom, že operační systém spustí pro tento účel hodiny zpravidla s tikem 200 ms (délka tiku musí být pod 500 ms). Po každém tiku systém zkontroluje, zdali není něco k odeslání (potvrzení přijatých dat či odeslání dat). Pokud je třeba něco odeslat, pak vše odešle najednou.

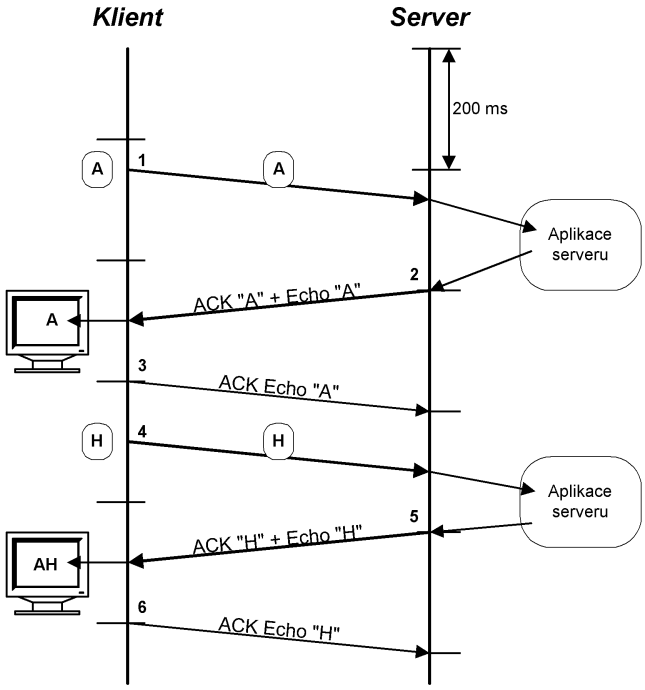
Na obrázku 9.13 server přijal znak A – segment (1), ale bude odpovídat až s dalším 200ms tikem. Do té doby aplikace stihne i připravit data k odeslání (echo A). Operační systém jedním TCP segmentem (segment (2)) provede potvrzení znaku A a zároveň odešle echo A.

Jenže je již málo pravděpodobné, že klient stiskne další klávesu H tak, aby software klienta byl schopen znak H odeslat společně s potvrzením přijetí echa klávesy A. Proto jak je z následujícího obrázku patrné klient provede potvrzení echa klávesy A pomocí segmentu (3) a stisk klávesy H způsobí vyslání dalšího TCP segmentu (4).

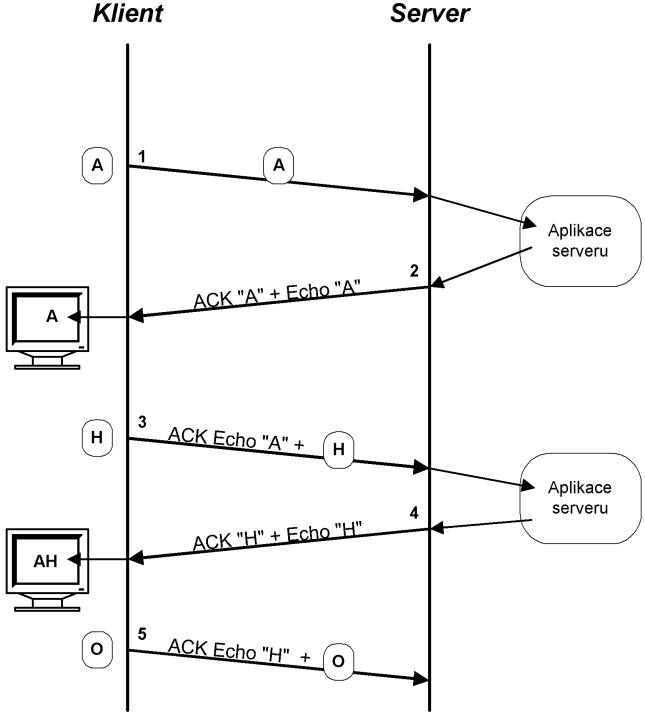
V ideálním případě dojde k redukci přenášených TCP segmentů ze čtyřech na tři. Ještě větší úspory přinese použití Nagleova algoritmu znázorněného na obrázku 9.14.

V tomto případě software klienta nečeká na další tik (200 ms), ale čeká až dojdou nějaká data od protější strany. Tento algoritmus vyrovnává dobu odezvy vůči kapacitě přenosové linky (řídí tok dat). Tj. je-li linka více zatížena, pak je na ní delší odezva a odpověď se také pozdrží.

Obr. 9.13
Zpoždění 200 ms



Obr. 9.14
Nagleův algoritmus



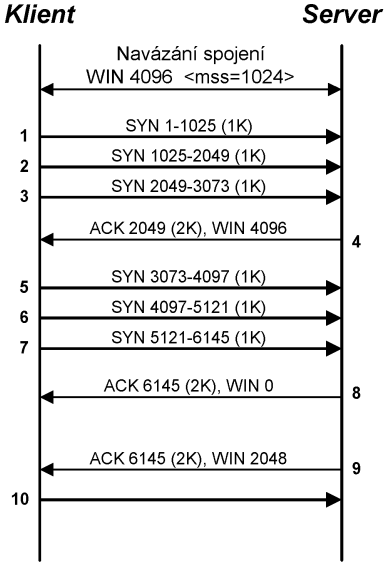
Technika zpoždění odpovědi je výhodná pro aplikace typu telnet, ale např. pro X-server je naopak nežádoucí. Pokud bychom ji použili pro X-server, pak pohyb myši by se nám na obrazovce mohl jevit trhaný. Pro aplikace přenášející velké objemy dat (např. HTTP, datový kanál FTP atd.) technika pozdržení odpovědi také ztrácí smysl.

9.7 Technika okna

Nyní je naším problémem případ, kdy klient potřebuje odeslat velké množství dat. Klient (resp. Server) může odesílat data druhé straně aniž by jejich příjem měl potvrzen až do tzv. okna (*Window* – zkratkou WIN).

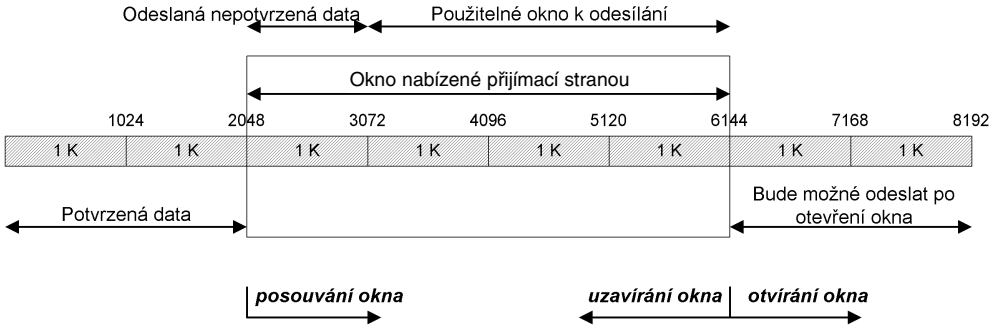
Představme si (obr. 9.15), že klient se serverem navázal spojení a vzájemně se dohodli na maximální velikosti segmentu (MSS) o velikosti 1 K (tj. 1024 B) a vzájemné velikosti okna 4 K (tj. 4096 B).

Obr. 9.15
Technika okna



Klient začne s odesíláním dat, odešle segmenty 1, 2 a 3. Poté obdrží od serveru potvrzení (4), které potvrzuje segmenty 1 a 2. Klient v zápětí odesílá segmenty 5, 6 a 7. Jenže server data mezitím nedokázal zpracovat a data mu zaplnila vyrovnávací paměť, proto segmentem 8 sice potvrdí příjem segmentů 3, 5, 6 a 7, ale zároveň klientovi uzavře okno, tj. klient nemůže s odesíláním dat pokračovat. Poté co server zpracuje část dat (2 KB), tak umožní klientovi pokračovat v odesílání, ale neotevře mu segmentem 9 okno celé – pouze 2 KB, protože všechna data ve vyrovnávací paměti ještě nezpracoval a pro více dat nemá místo.

Prozkoumejme na obr. 9.16 jak vidí klient okno po přijetí segmentu 4.



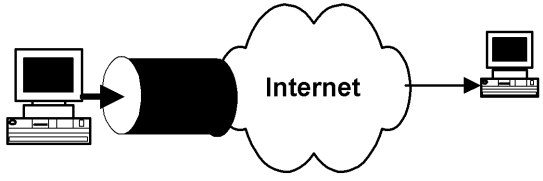
Obr. 9.16 Okno

První 2 KB jsou již potvrzeny, okno je tedy posunuto za bajt 2048. Tato potvrzená data již klient nemusí udržovat v paměti. Odeslaná, ale nepotvrzená data (segment 3) tvoří 1 KB. Klient může tedy odeslat bez dalšího potvrzení 3 KB dat.

9.8 Zahlcení sítě

Okno je množství dat, které je schopen přijmout příjemce. Okno je také příjemcem inzerováno. Problém je však také na straně odesílatele. Pokud je odesílatel na rychlé síti a příjemce na pomalé síti, tak by odesílatel mohl doslova nacpat do sítě data až do velikosti okna (předpokládáme, že velikost okna by byla shodou okolností značně velká). Jelikož síť by nebyla schopna toto velké množství dat přenést, tak by došlo k zahlcení sítě a data, která síť není schopna přenést, zahodí. Směrovače ukládají IP-datagramy do vyrovnávací paměti, ale i ta je omezená.

Obr. 9.17 Zahlcení



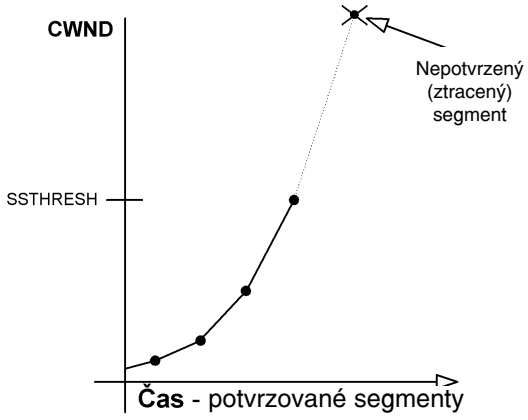
Ztráta dat je vždy nepříjemná. Snahou je se ztrátě dat vyhnout. Na straně odesílatele se proto také zavádí okno. Toto okno se snaží specifikovat, kolik může odesílatel odeslat nepotvrzených dat, aniž by došlo k zahlcení sítě. Toto okno na straně odesílatele se anglicky nazývá *Congestion Window* (zkratkou CWND). Odesílatel postupně zvyšuje CWND. CWND však nelze zvyšovat neomezeně. Hranice, za kterou už je větší pravděpodobnost zahlcení, se označuje SSTHRESH. Internet však budeme chtít využívat maximálně, tj. budeme chtít najít největší použitelné CWND (to bude někde nad SSTHRESH). SSHTRESH má smysl udržovat pouze v násobcích velikosti odesílaného segmentu (MSS).

Odesílatel odesílá vždy jen takové množství nepotvrzených dat, které nepřevyšuje okno inzerované příjemcem (WINDOW) ani nepřevyšuje CWND, tj. odesílá jen nejvýše min(WINDOW,CWND) nepotvrzených dat.

9.8.1 Pomalý start

Otázkou je jak určit maximální možné CWND. To odesílatel určuje dynamicky. Nejprve odešle jeden segment a vyčká na jeho potvrzení. Pokud potvrzení obdrží, pak vyšle dva segmenty. Pokud potvrzení obdrží, pak odešle čtyři atd. Jedná se o řadu 2^n (která je exponenciální).

Obr. 9.18
Pomalý start



Pochopitelně, že po několika krocích se odesílatel dostane do situace, kdy síť zahltí a potvrzení nedostane, tj. musí opakovat odesílání segmentů, protože se segment ztratil. V takovém okamžiku se CWND zmenší na polovinu a tato hodnota se také nastaví do veličiny SSTHRESH (pokud by SSTRESH mělo být menší než dva segmenty, pak se nastaví na velikost dvou segmentů).

Je třeba rozlišovat jakým způsobem byl zjištěn stav, že segment nebyl potvrzen. Nyní jsme předpokládali, že se segment po cestě k příjemci ztratil. Příjemce segment nedostal, a tak stále potvrzuje předchozí (přijatý) segment. Po třech zopakovaných potvrzeních předchozího přijatého segmentu se segment považuje za ztracený a odesílatel jej opakuje. Může však nastat situace, že odesílatel ve stanoveném časovém intervalu nedostane vůbec žádné potvrzení (ani žádného předchozího segmentu). V takovém případě se CWND nastaví na velikost jednoho segmentu (MSS) a SSTHRESH na dvojnásobek velikosti segmentu (2x MSS) a začne se s pomalým startem od počátku.

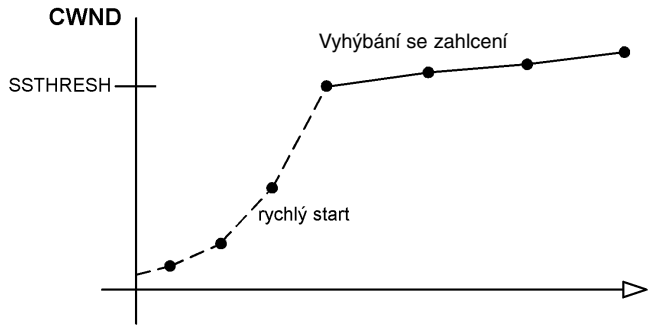
9.8.2 Vyhýbání se zahlcení

Odesílatel udržuje pro každé spojení aktuální hodnoty proměnných MSS, WINDOW, CWND i SSTHRESH. MSS je určeno příjemcem v okamžiku navazování spojení (segment s touto volbou má příznak SYN). WINDOW určuje příjemce dynamicky během spojení – specifikuje množství dat, která se mu vejdou do vyrovnávací paměti.

Odesílatel v okamžiku odesílání segmentu nemůže meditovat nad tím co má udělat, ale musí se rozhodnout rychle na základě udržovaných proměnných:

1. Když je CWND menší nebo rovno SSTHRESH, pak se jedná o pomalý start. Je tedy možné se pokusit o odeslání dvojnásobku dat.
2. V případě, že CWND je již větší než SSTHRESH, pak odeslání dvojnásobku by již pravděpodobně způsobilo zahlcení. V tomto případě se CWND zvyšuje pouze o (MSSxMSS/CWND + MSS/8) počítáno v celočíselné aritmetice. Toto „drobné zvětšování" CWND se nazývá algoritmus vyhýbání se zahlcení (*Congestion Avoidance Algorithm*).

Obr. 9.19
Vyhýbání se zahlcení



9.8.3 Ztráta segmentu

Ztrátě TCP segmentu nezamezí ani uvedený algoritmus. Ke ztrátě může dojít i změnou situace na přenosových cestách, poruchou přenosové cesty atd.

V případě, že CWND je značně velké (může být řádově i MB či dokonce GB), pak opakování celého nepotvrzeného okna v případě ztráty jednoho segmentu by bylo velice nepříjemné, protože by enormně zvyšovalo režii. Je proto snaha využít tzv. algoritmus rychlého zopakování.

Jak však odesílatel zjistí, že došlo ke ztrátě TCP segmentu? (Nyní se již nebudeme zmiňovat o případu, že odesílatel nedostává od příjemce potvrzováno vůbec nic.) Příjemce zjistí, že došlo ke ztrátě segmentu tím, že dostává další segmenty (a ztracený segment stále nedochází). Tj. chybí mu data v posloupnosti přijímaných dat. Přicházejí mu tedy segmenty mimo pořadí. Příjemce je povinen v případě přijetí segmentu mimo pořadí zopakovat (duplikovat) potvrzení posledního správně přijatého segmentu.

Jenže TCP segmenty jsou zabaleny do IP-datagramu. Každý IP-datagram putuje Internetem samostatně a teoreticky jinou cestou. Některé cesty jsou rychlejší a jiné pomalejší. Může se stát, že segment putoval pomalejší cestou a přirozeně dorazil později než následující segment. Mezitím však příjemce již provedl odeslání duplikovaného potvrzení.

Přijetí jedné duplikované odpovědi je proto bráno vcelku za běžnou záležitost. Jiná je situace, když se segment opravdu ztratil. Příjemce pak segment nedostal a dostal následující segment. Provedl duplikaci potvrzení posledního správně přijatého segmentu. Poté však dostane ještě následující segment, ten je však také mimo pořadí, protože příjemce ještě neobdržel ztracený segment. Opět provede duplikaci. Příjemce obdrží opět další segment, který je rovněž mimo pořadí – zopakuje duplikaci atd.

Odesílatel pak od příjemce postupně dostává první duplikát, druhý duplikát a stále si říká, že to je vcelku normální. Po obdržení třetího duplikátu si však řekne: „To už se opravdu muselo něco stát, zopakuji mu segment, o kterém se příjemce domnívá, že je ztracen" a provede zopakování ztraceného segmentu. Příjemce obdrží ztracený segment. Avšak jelikož příjemce nezahodil žádný z následujících segmentů (původně přijatých mimo pořadí), tak potvrdí přijetí všech přijatých segmentů.

Tento algoritmus rychlého zopakování umožňuje zopakovat pouze ztracený segment a nikoliv nutnost opakování všech nepotvrzených dat. Opakování všech nepotvrzených dat je nutné pouze v případě, že se algoritmus rychlého opakování nestihne v zadaném časovém intervalu.

Jak je to v případě rychlého zopakování s proměnnýcmi CWND a SSTHRESH?

1. Když příjemce obdrží třetí duplikát, pak:
   a) nastaví SSHTRESH na CWND/2,
   b) zopakuje odeslání TCP segmentu,
   c) nastaví CWND na SSHTRESH+3xMSS.
2. Po přijetí čtvrtého a dalšího duplikátu odesílatel vždy pokaždé zvětší CWND o velikost segmentu (MSS).
3. V případě, že ztracený segment je konečně potvrzen příjemcem, pak odesílatel nastaví CWND na hodnostu SSTHRESH.

Stabilní hodnotu proměnné SSTHRESH není snadné určit. Obzvláště na počátku spojení může kolísat. Některé operační systémy UNIX (např. DEC OSF/1 od verze 4) proto ve svých směrovacích tabulkách pro samostatně směrované počítače hodnotu SSTHRESH udržují i po ukončení spojení a u nových spojení tuto hodnotu pak použijí jako výchozí. Příkazem

```
netstat -rv
```

je možné i tyto rozšiřující položky směrovací tabulky vypsat. Sloupec s proměnnou SSTHRESH bývá nadepsán „Thresh".

9.9 Volba zvětšení okna

Okno inzerované příjemcem má v TCP záhlaví vyhrazeny 2 B. Příjemce proto může inzerovat okna pouze v rozmezí 0 až 65535. Taková okna jsou u gigabitových sítí příliš malá. Řešením je použití volitelné položky „zvětšení okna" v záhlaví TCP segmentu. Tato volba může být použita pouze v segmentech inicializujících spojení, tj. v segmentech s příznakem SYN.

Pomocí volitelné položky „zvětšení okna" se oba konce spojení dohodnou na zvětšení okna v intervalu 0 až 14. Označme toto zvětšení jako $\underline{n}$. V každém směru spojení může být dohoda jiná.

Hodnota zvětšení okna se použije zajímavým způsobem. V případě, že odesílatel nabízí okno o velikosti $\underline{o}$ a nabídl zvětšení okna $\underline{n}$. Pak příjemce chápe, že odesílatel nabízí okno $ox2^n$ (tj. posune šířku okna o n bitů).

Největší inzerovatelné okno je $65535x2^{14}$ (=1073725440=1G-16384). Oknem lze tedy maximálně inzerovat téměř 1GB.

Proč nemůže být okno ještě větší? Je to jednoduché. Přenášená data se číslují od 0 do 2^{32} (=4GB). V případě přetečení 2^{32} se opět začíná od nuly. Při zvětšení okna např. na 8 GB by odesílatel mohl odeslat 8 GB nepotvrzených dat. V případě, že by si příjemce přál nějaký segment z těchto 8 GB opakovat (např. segment začínající na 2 GB), pak by odesílatel nevěděl zdali mu opakovat segment začínající od 2 GB nebo od 6 GB, protože oba budou mít stejná pořadová čísla odeslaného bajtu (po 4 GB se začalo opět počítat od nuly).

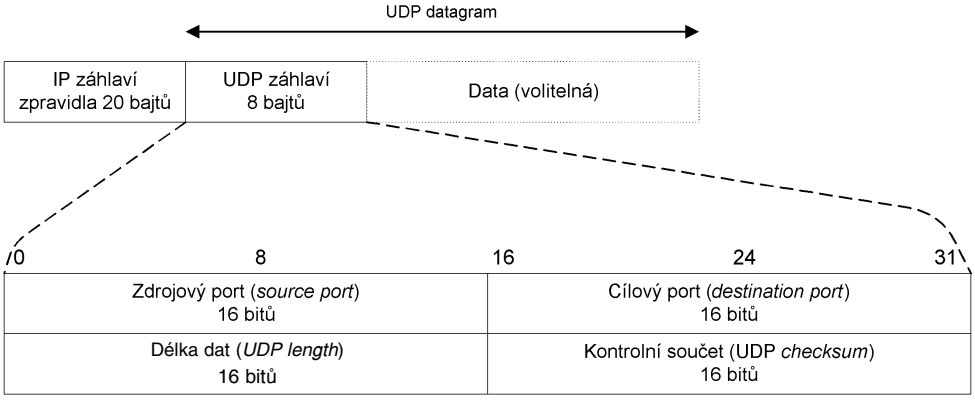
I při použití okna velkého stovky MB (což je přípustné) se může stát, že Internetem bude putovat segment z již potvrzeného okna, avšak s číslem používaným v aktuálním okně. Tento problém se řeší použitím další volitelné položky v záhlaví TCP segmentu „časové razítko" (tato položka může být v každém segmentu). Odesílatel do této volby vkládá jednoznačnou rostoucí posloupnost (čas). Příjemce pak do odpovědi vloží své časové razítko a zopakuje poslední přijaté časové razítko. Lze tak poznat, které segmenty jsou staré a které aktuální. Lze tak detekovat starý zatoulaný segment.

10

Protokol UDP (*User Datagram Protocol*)

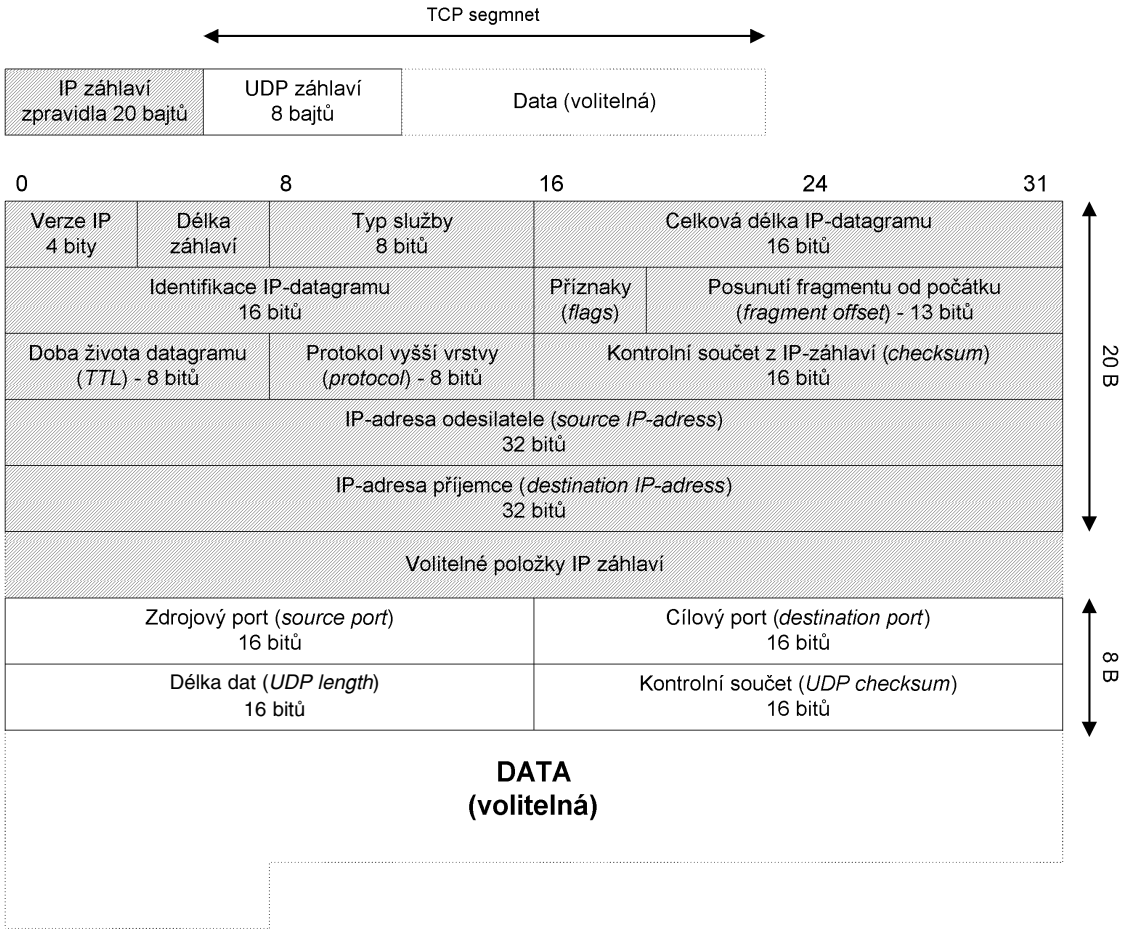
Protokol UDP je jednoduchou alternativou protokolu TCP. Protokol UDP je nespojovaná služba (na rozdíl od protokolu TCP), tj. nenavazuje spojení. Odesílatel odešle UDP datagram příjemci a už se nestará o to, zdali se datagram náhodou neztratil (o to se musí postarat aplikační protokol).

UDP datagramy jsou baleny do IP-datagramu, jak je znázorněno na obrázku 10.1.



Obr. 10.1 Záhlaví UDP datagramu

Celková podoba UDP datagramu včetně jeho IP-záhlaví je znázorněna na obrázku 10.2.



Obr. 10.2 UDP datagram

Z předchozího obrázku je patrné, že záhlaví UDP protokolu je velice jednoduché. Obsahuje čísla zdrojového a cílového portu – což je zcela analogické protokolu TCP. Opět je třeba dodat, že čísla portů protokolu UDP nesouvisí s čísly portů protokolu TCP. Protokol UDP má svou nezávislou sadu čísel portů.

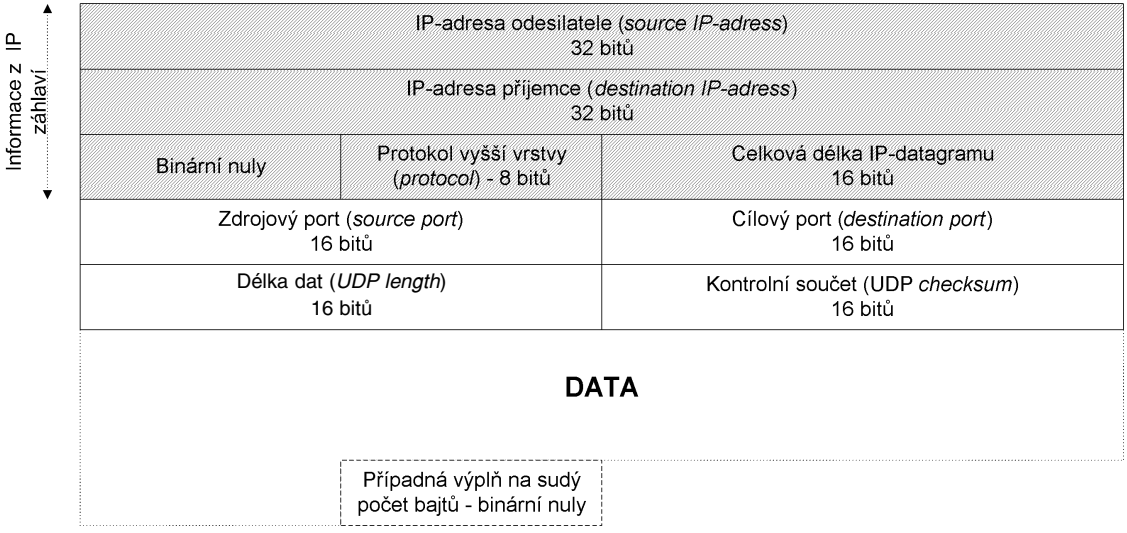
Pole délka dat obsahuje délku UDP datagramu (délku záhlaví + délku dat). Minimální délka je tedy 8, tj. UDP datagram obsahující pouze záhlaví a žádná data.

Zajímavé je že pole kontrolní součet nemusí být povinně vyplněné. Výpočet kontrolního součtu je tak v protokolu UDP nepovinný.

V minulosti bylo u některých počítačů zvykem výpočet kontrolního součtu vypínat – zejména se jednalo o počítače s instalovaným systémem NFS (*Network File System*). Důvodem bylo zrychlení odezvy počítače.

Zejména u důležitých serverů je třeba vždy zkontrolovat, zdali je opravdu výpočet kontrolního součtu zapnut. Nejnebezpečnější je to v případě DNS serveru, protože kontrolní součet pak je počítán jen na linkové vrstvě, ale např. linkový protokol SLIP výpočet kontrolního součtu také nepočítá, takže i technická porucha může způsobit poškození aplikačních dat, aniž by měl příjemce šanci to zjistit.

Pakliže se kontrolní součet počítá, pak se podobně jako pro protokol TCP počítá ze struktury (pseudozáhlaví) znázorněné na obrázku 10.3.



Obr. 10.3 Pseudozáhlaví pro výpočet kontrolního součtu v UDP datagramu

10.1. Fragmentace

I u UDP datagramů je možná fragmentace v IP-protokolu. Avšak u UDP protokolu se zásadně snažíme fragmentaci vyhýbat.

Typickým případem je DNS. DNS klient položí dotaz protokolem UDP. Pakliže odpověď serveru by přesáhla 512 B, pak server odešle jen tolik informací, aby nepřekročil hranici 512 B a navíc v aplikačních datech nastaví příznak TC (Truncation) specifikující, že odpověď byla zkrácena. Pakliže klientovi taková odpověď nestačí, pak ji zopakuje protokolem TCP, kterým mu server vrátí kompletní odpověď.

10.2. Příklad UDP datagramu

```
+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
  IP: ID = 0x9CCE; Proto = UDP; Len: 74
      IP: Version = 4 (0x4)
      IP: Header Length = 20 (0x14)
    + IP: Service Type = 0 (0x0)
      IP: Total Length = 74 (0x4A)
```

```
      IP: Identification = 40142 (0x9CCE)
    + IP: Flags Summary = 0 (0x0)
      IP: Fragment Offset = 0 (0x0) bytes
      IP: Time to Live = 30 (0x1E)
      IP: Protocol = UDP - User Datagram
      IP: Checksum = 0x803D
      IP: Source Address = 194.149.104.203
      IP: Destination Address = 192.36.148.18
      IP: Data: Number of data bytes remaining = 54 (0x0036)
  UDP: Src Port: DNS, (53); Dst Port: DNS (53); Length = 54 (0x36)
      UDP: Source Port = DNS
      UDP: Destination Port = DNS
      UDP: Total length = 54 (0x36) bytes
      UDP: UDP Checksum = 0x13A0
      UDP: Data: Number of data bytes remaining = 46 (0x002E)
+ DNS: 0x7E01:Std Qry for 130.204.212.195.in-addr.arpa. of type Dom. name ptr INET addr.

00000:  00 60 3E 1D 90 00 00 00 F8 21 71 A4 08 00 45 00   .`>......!q...E.
00010:  00 4A 9C CE 00 00 1E 11 80 3D C2 95 68 CB C0 24   .J.......=..h..$
00020:  94 12 00 35 00 35 00 36 13 A0 7E 01 00 00 00 01   ...5.5.6..~.....
00030:  00 00 00 00 00 00 03 31 33 30 03 32 30 34 03 32   .......130.204.2
00040:  31 32 03 31 39 35 07 69 6E 2D 61 64 64 72 04 61   12.195.in-addr.a
00050:  72 70 61 00 00 0C 00 01                           rpa.....
```

10.3. Oběžníky

Na první pohled by se zdálo, že protokol UDP je chudým příbuzným protokolu TCP. Může však existovat něco, co umí protokol UDP a nelze to udělat protokolem TCP? Právě zvláštností protokolu UDP je skutečnost, že adresátem UDP datagramu nemusí být pouze jednoznačná IP-adresa, tj. síťové rozhraní konkrétního počítače. Adresátem může být skupina stanic – adresovat lze i oběžník.

Adresovat lze všeobecné oběžníky (*broadcast*), ale podstatně zajímavějším případem je adresování adresných oběžníků (*multicast*). Např. u aplikací typu RealAudio navazuje každý klient spojení se serverem. Kdežto u ProgresiveRealAudio se šíří data pomocí adresných oběžníků, tj. dochází k ohromné úspoře kapacity přenosových cest. A právě to je příležitost pro UDP.

10.4. Na co je UDP krátký?

U ProgresiveRealAudio je Internetem transportován zvukový záznam. V případě ztráty UDP datagramu to v reproduktorech zapraská, ale není třeba si vyžádat opakování ztracených dat.

Problémem protokolu UDP je právě jak v případě adresných oběžníků dožádat nedoručená data. V jakém případě by bylo třeba vůbec vyžadovat ztracená data? Případem může být přenos souborů přes adresné oběžníky, tj. rozesílání souborů mnoha uživatelům současně (např. aplikačním protokolem MFTP – *Multicast File Transfer Protocol*). Může se totiž stát, že některý z adresátů neobdrží část souboru. Pak se adresát chce dožadovat zopakování ztracené části souboru. Jenže koho se má dožadovat – pokud by se dožadoval přímo odesílatele, tak v celosvětovém měřítku by mohlo takové zopakování znamenat i zahlcení celého Internetu. Adresát se bude dožadovat svého nejbližšího *mrouteru* (směrovače šířícího adresné oběžníky). To už ale protokol UDP opravdu nedokáže. Nastupují tak na scénu moderní protokoly pro šíření oběžníků, jakým se dnes jeví např. návrh protokolu PGM…

11

DNS

Všechny aplikace, které zajišťují komunikaci mezi počítači používají k identifikaci komunikujících uzlů IP-adresu. Pro člověka jako uživatele jsou však IP-adresy těžko zapamatovatelné. Proto se používá místo IP-adresy název síťového rozhraní. Pro každou IP-adresu máme zavedeno jméno síťového rozhraní (počítače), přesněji řečeno doménové jméno. Toto doménové jméno můžeme používat ve všech příkazech, kde je možné použít IP adresu. Výjimkou, kdy se musí použít IP-adresa, je identifikace samotného name serveru.

Jedna IP-adresa může mít přiřazeno i několik doménových jmen.

Vazba mezi jménem počítače a IP adresou je definována v DNS databázi. DNS (*Domain Name System*) je celosvětově distribuovaná databáze. Jednotlivé části této databáze jsou umístěny na tzv. name serverech.

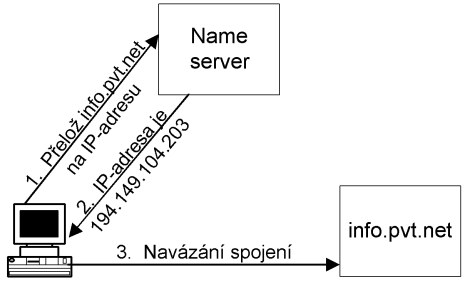
Příklad (obr. 11.1):

Chci-li se přihlásit na uzel info.pvt.net s IP adresou 194.149.104.203, použiji příkaz:

```
telnet info.pvt.net.
```

Ještě předtím, než se vlastní příkaz provede, přeloží se DNS jméno info.pvt.net na IP adresu a teprve poté se provede příkaz:

```
telnet 194.149.104.203
```



Obr. 11.1
Před navázáním spojení je nutné přeložit jméno na IP-adresu

Použití IP-adres místo doménových jmen je praktické vždy, když máme podezření, že DNS nám na počítači nepracuje korektně. Pak, ač to vypadá nezvykle, můžeme napsat např:

```
ping 194.149.104.203
http://194.149.104.203
```

či odeslat mail na: `dostalek@[194.149.104.203]`

Reakce však může být neočekávaná zejména pro poštu, protokol HTTP a protokol HTTPS. Poštovní server totiž nemusí podporovat dopravu na servery uvedené v hranatých závorkách. Protokol HTTP nám vrátí primární webovou stránku a protokol HTTPS se bude rozčilovat, že jméno serveru neodpovídá položce CN v jedinečném jméně certifikátu serveru.

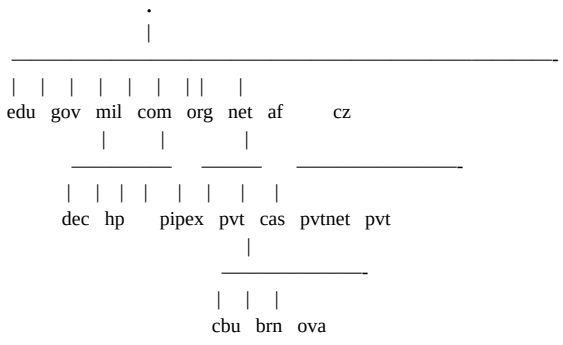
11.1 Domény a subdomény

Celý Internet je rozdělen do tzv. domén, tj. skupin jmen, která k sobě logicky patří. Domény specifikují, patří-li jména jedné firmě, jedné zemi apod. V rámci domény je možné vytvářet podskupiny, tzv. subdomény, např. doméně firmy lze vytvořit subdomény pro oddělení. Doménové jméno odráží příslušnost uzlu do skupiny a podskupiny. Každá skupina má přižazeno jméno. Z jednotlivých jmen skupin je pak složeno doménové jméno uzlu. Např. uzel se jménem jakub.firma.cz je uzel se jménem jakub v subdoméně firma domény cz.

Doménové jméno se skládá z řetězců vzájemně oddělených tečkou. Jméno se zkoumá zprava doleva. Nejvyšší instancí je tzv. root doména, která se vyjadřuje tečkou zcela vpravo (tato tečka bývá často vypouštěna). V root doméně jsou definované generické domény (*Top Level Domains – TLD*): edu, com, net, org, mil, int a arpa, které se používají převážně v USA, a dále podle normy ISO-3166 dvojznakové domény jednotlivých států. Pro Českou republiku je vyhrazena doména cz.

Doména cz se dělí na subdomény pro jednotlivé organizace: pvt.cz (pro PVT, a.s.), cas.cz (pro Českou Akademii Věd), cvut.cz (pro ČVUT) atd. Subdomény se mohou dělit na subdomény nižší úrovně. Např. entu.cas.cz (Entomologický ústav ČAV) atd. Subdomény obsluhují jako prvky počítače.

Jména tvoří stromovou strukturu:



Doména cz obsahuje doménu pvtnet. Doména pvtnet.cz obsahuje krajské subdomény: **pha, cbu, plz, unl, hrk, brn a ova**. Teoreticky by mohly být prvky těchto subdomén i subdomény ještě nižší úrovně atd.

11.2 Syntaxe jména

Jméno je uváděno v tečkové notaci. Např. abc.cbu.pipex.cz. Jméno má obecně syntaxi:

řetězec.řetězěc.řetězec....řetězec.

kde první řetězec je jméno počítače, další jméno nejnižší vnořené domény, další vyšší domény atd. Pro jednoznačnost se na konci uvádí také tečka, vyjadřující root doménu.

Celé jméno může mít maximálně 255 znaků, řetězec pak maximálně 63 znaky. Řetězec se může skládat z písmen, číslic a pomlčky. Pomlčka nesmí být na začátku ani na konci řetězce. Existují i rozšíření specifikující bohatší repertoár znaků použitelných pro tvorbu jmen. Zásadně se však těmto dalším znakům vyhýbáme, protože jen některé aplikace toto rozšíření podporují.

Mohou se použít velká i malá písmena, ale není to zase tak jednoduché. Z hlediska uložení a zpracování v databázi jmen (databázi DNS) se velká a malá písmena nerozlišují. Tj. jméno *newyork.com* bude uloženo v databázi na stejné místo jako *NewYork.com* nebo *NEWYORK.com* atp. Tedy při překladu jména na IP-adresu je jedno, kde uživatel zadá velká a kde malá písmena. Avšak v databázi je jméno uloženo s velkými a malými písmeny, tj. bylo-li tam uloženo např. *NewYork.com*, pak při dotazu databáze vrátí *NewYork.com*. Poslední tečka je součástí jména.

V některých případech se může část jména zprava vynechat. Téměř vždy můžeme koncovou část doménového jména vynechat v aplikačních programech. V databázích popisujících domény je však situace složitější.

Je možné vynechat:

- Poslední tečku téměř vždy.
- Na počítačích uvnitř domény se zpravidla může vynechat konec jména, který je shodný s názvem domény. Např. uvnitř domény pipex.cz, je možné psát místo počítač.abc.pipex.cz jen počítač.abc (nesmí se ale uvést tečka na konci!). Do kterých domén počítač patří se definuje příkazy domain a search v konfiguračním souboru resolveru.

Upozornění

Netvořte jména subdomén v kolizi se jmény Top Level Domén. Např. chcete-li rozdělit doménu *pipex.cz* na subdomény podle krajských měst a použijete dvojznakové řetězce, vznikne problém. Pro Liberec byste si vybrali např. *lb.pipex.cz*.

Uživatel z domény *cbu.pipex.cz* bude psát mail uživateli Alois v doméně *lb.pipex.cz* a napíše podle předchozího pravidla příkaz:

```
mail Alois@lb
```

(oba jsou přece v doméně pipex.cz). No a milý mail dojde klidně do Libanonu. Důvodem je to, že neexistuje přesná specifikace „místní domény“. To, co se doplňuje zprava v případě, že uživatel nezadal tečku na konci, je zcela v kompetenci místního správce.

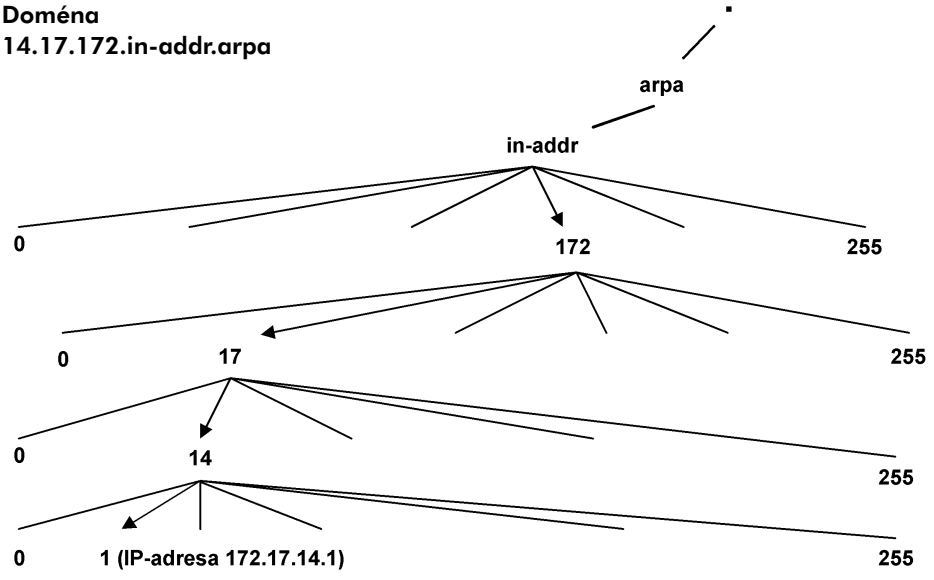
Uvedenému problému předejdete, zvolíte-li si pro Liberec např. doménu *lbc.pipex.cz*.

11.3 Reverzní domény

Již jsme uvedli, že komunikace mezi uzly probíhá na základě IP adres, nikoli doménových jmen. Některé aplikace naopak potřebují k IP-adrese nalézt jméno, tj. nalézt tzv. reverzní záznam. Jedná se tedy o překlad IP-adresy na doménové jméno. Tento překlad se často nazývá zpětným (reverzním) překladem.

Podobně jako domény tvoří i IP-adresy stromovou strukturu. Domény tvořené IP-adresami se pak často nazývají reverzní domény. Pro účely reverzního překladu byla definována pseudodoména „in-addr.arpa". Jméno této pseudo domény má historický původ, jde o zkratku „*inverse addresses in the Arpanet*".

Obr. 11.2
Doména
14.17.172.in-addr.arpa



Pod doménou in-addr.arpa jsou domény jmenující se jako první číslo z IP-adresy sítě. Např. síť 194.149.101.0 patří do domény 194.in-addr.arpa. Síť 172.17 patří do domény 172.in-addr.arpa. Dále doména 172.in-addr.arpa se dělí na subdomény, takže síť 172.17 tvoří subdoménu 17.172.in-addr.arpa. Je-li síť 172.17 rozdělena pomocí síťové masky na subsítě, pak každá subsíť tvoří ještě vlastní subdoménu. Všimněte si, že domény jsou zde tvořeny jakoby IP-adresami sítí psanými ale pozpátku.

Reverzní domény pro subsítě adres třídy C jsou tvořeny podle metodiky classless in-addr.arpa. Přesto že IP-adresa má pouze 4 bajty a klasická reverzní doména má tedy maximálně 3 čísla, jsou reverzní domény pro subsítě třídy C tvořeny 4 čísly.

Příklad:

Reverzní doména pro subsíť 194.149.150.16/28 je 16.150.149.194.in-addr.arpa

Opět i tyto reverzní subdomény sítí třídy C tvoří stromovou strukturu.

11.4 Doména 0.0.127.in-addr.arpa

Jistou komplikací (zvláštností) je adresa sítě 127.0.0.1. Síť 127 je totiž určena pro *loopback*, tj. softwarovou smyčku na každém počítači.

Zatímco ostatní IP-adresy jsou v Internetu jednoznačné, adresa 127.0.0.1 se vyskytuje na každém počítači.

Každý name server je autoritou nejen „obyčejných" domén, ale ještě autoritou (primárním name serverem) k doméně **0.0.127.in-adr.arpa**. V dalším textu budeme tento fakt považovat za samozřejmost a v tabulkách jej pro přehlednost nebudeme uvádět, ale nikdy na něj nesmíte zapomenout.

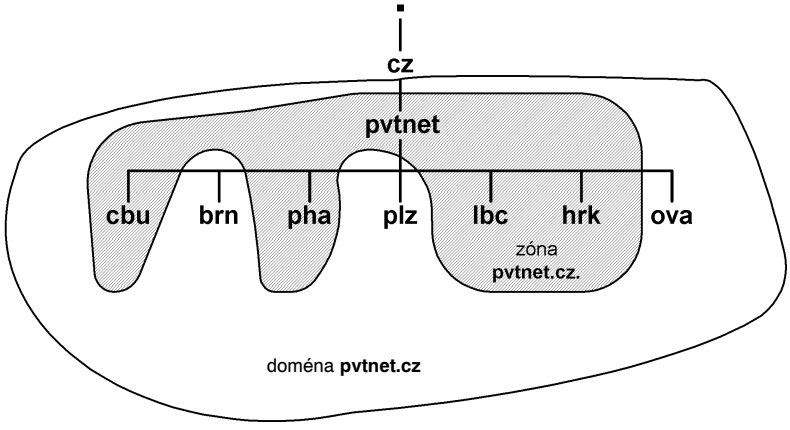
11.5 Zóna

Často se setkáváme s otázkou: „Co je to zóna?" „Jaký je vztah mezi doménou a zónou?". Vysvětleme si tedy vztah těchto pojmů na doméně cz.

Jak jsme již uvedli, doména je skupina počítačů, které mají společnou pravou část svého doménového jména. Doména je např. skupina počítačů, jejichž jméno končí cz. Doména cz je však velká. Dělí se dále na subdomény např. pvt.cz, eunet.cz a tisíce dalších. Každou z domén druhé úrovně si většinou spravuje na svých name serverech majitel domény nebo jeho poskytovatel Internetu. Data pro doménu druhé úrovně např. pvt.cz nejsou na stejném name serveru jako doména cz. Jsou rozložena na mnoho name serverů. Data o doméně uložená na name serveru jsou nazývána zónou. Zóna tedy obsahuje jen část domény.

Zóna je část prostoru jmen, kterou obhospodařuje jeden name server.

Obr. 11.3
Zóna pvtnet.cz

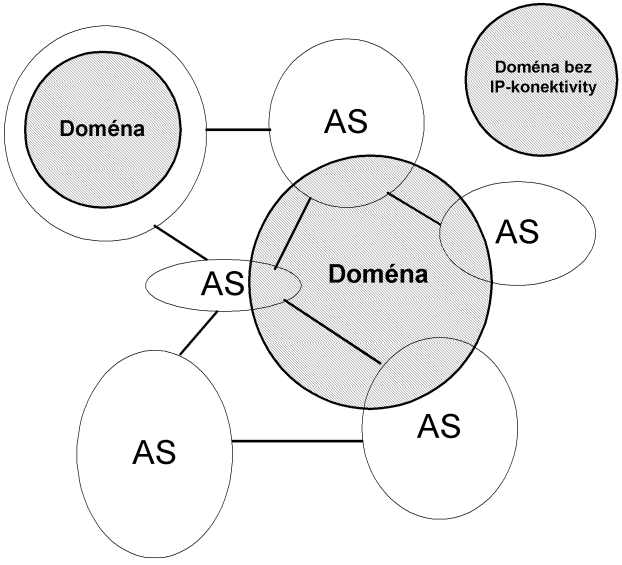


Na obrázku 11.3 je znázorněno, jak může být (hypoteticky) v doméně *pvtnet.cz* pomocí vět typu NS decentralizována kompetence na nižší správní celky. Takže doména *pvtnet.cz* obsahuje v sobě všechny subdomény, ale zóna *pvtnet.cz* delegovala na jiné name servery pravomoci na zóny *brn.pvtnet.cz*, *plz.pvtnet.cz* a *ova.pvtnet.cz*. Takže zóna *pvtnet.cz* obsahuje doménu pvtnet.cz až na tři uvedené výjimky.

11.6 Doména a autonomní systém

Na tomto místě musíme zdůraznit, že rozdělení sítě na autonomní systémy nesouvisí s rozdělením na domény (nebo snad na zóny). Tzn. je-li podniku přiděleno jméno domény a IP-adresy sítí jedním poskytovatelem, pak při přechodu k jinému poskytovateli zůstanou podniku jména domén, ale IP-adresy dostane od nového poskytovatele nové. Musí se tedy přečíslovat jednotlivé LAN, ale jména počítačů a adresy elektronické pošty zůstanou beze změn.

Obr. 11.4 Domény a autonomní systémy



Autonomní systémy dělí Internet z hlediska IP-adres (směrování), naproti tomu domény dělí Internet z hlediska jmen počítačů.

Jiná je situace u reverzních domén, které kopírují strukturu poskytovatelů Internetu.

11.7 Rezervované domény a pseudodomény

Později se ukázalo, že jako TLD je možné využít i jiné domény. Některé další TLD byly rezervovány RFC-2606:

- doména **.test** pro testování.
- doména **.expample** pro vytváření dokumentace a příkladů.
- doména **.invalid** pro navozování chybových stavů.
- doména **.localhost** pro softwarovou smyčku

Obdobně byla rezervována doména **.local** pro intranety. Význam této domény je obdobný jako význam sítě 10.0.0.0/8. V intranetu je tak možné využívat nejednoznačnou doménu, čímž si ulehčíme práci se dvěma různými doménami stejného jména *firma.cz* – jednou v Internetu a druhou v intranetu.

Z obrázku 11.4 je patrné, že mohou existovat i domény, které nejsou přímo připojeny k Internetu, tj. jejichž počítače ani nepoužívají síťový protokol TCP/IP – tedy nemají ani IP-adresu. Takovéto domény se někdy označují jako pseudodomény. Mají význam zejména pro elektronickou poštu.

Pomocí pseudodomény lze řešit problém posílání elektronické pošty do jiných sítí než Internet (např. DECnet či MS Exchange).

Firma ve své vnitřní síti používá jednak síťový protokol TCP/IP a jednak protokol DECnet. Z Internetu je adresován uživatel používající ve vnitřní síti protokol TCP/IP např. *Alois@počítač.firma.cz*. Ale jak adresovat uživatele na počítačích pracujících v protokolu DECnet?

Pro tento případ se vsune do adresy pseudodoména *dnet*. Takže uživatel je adresován *uživatel@počítač.dnet.firma.cz*. Pomocí DNS je veškerý mail adresovaný do domény *dnet.firma.cz* přesměrován na bránu do protokolu DECnet (brána domény firma.cz), která provede transformaci z protokolu TCP/IP (resp. SMTP) do protokolu DECnet (resp. Mail-11).

11.8 Dotazy (překlady)

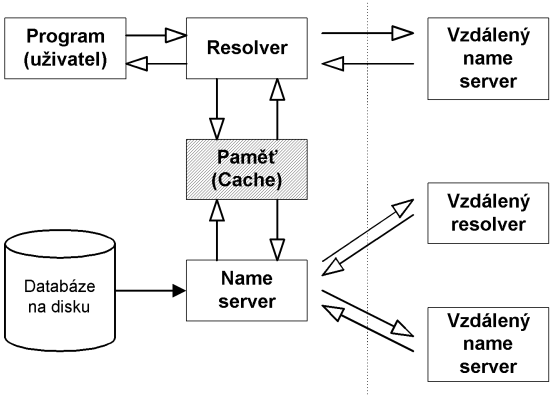
Přeložení jména na IP-adresu zprostředkovává tzv. resolver. Resolver je klient, který se dotazuje name serveru. Jelikož je databáze celosvětově distribuována, nemusí nejbližší name server znát odpověď, proto může tento name server požádat o pomoc další name servery. Získaný překlad pak name server vrátí jako odpověď resolveru. Veškerá komunikace se skládá z dotazů a odpovědí.

Name server po svém startu načte do paměti data pro zónu, kterou spravuje. Primární name server načte data z lokálního disku, sekundární name server dotazem zone transfer získá pro spravované zóny data z primárního name serveru a rovněž je uloží do paměti. Tato data primárního a sekundárního name serveru se označují jako autoritativní (nezvratná). Dále name server načte z lokálního disku do paměti data, která nejsou součástí dat jeho spravované zóny, ale umožní mu spojení s root name servery a případně s name servery, kterým delegoval pravomoc pro spravování subdomén. Tato data se označují jako neautoritativní.

Name server i resolver společně sdílejí paměť cache. Během práce do ní ukládají kladné odpovědi na dotazy, které provedly jiné name servery, tj. ke kterým jsou jiné name servery autority. Ale z hlediska našeho name serveru jsou tato data opět neautoritativní – pouze šetří čas při opětovných dotazech.

Představme si, že přijde dotaz (např. požadavek na překlad jména na IP-adresu) na name server. Je-li server pro daný dotaz autoritou (autoritativní name server) a nemá požadované jméno v databázi, odpoví negativně (jméno nelze přeložit na IP-adresu). Není-li name server pro daný dotaz autoritou, pak odpoví, že neví a doporučí name server, který by autoritou mohl být.

Obr. 11.5
DNS na serveru



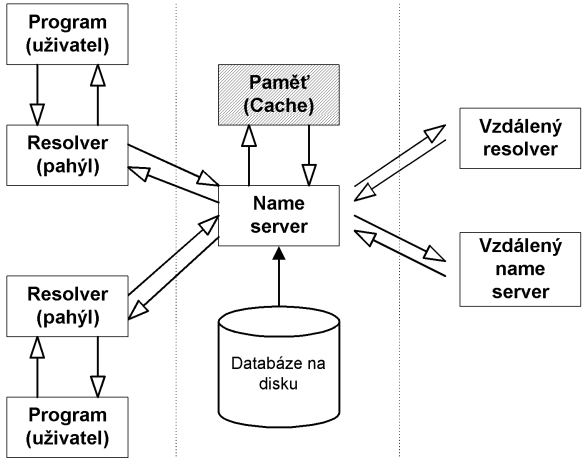
Do paměti se ukládají jen kladné odpovědi. Provoz by byl podstatně zrychlen, kdyby se tam ukládaly i negativní odpovědi (negativní caching), avšak to je podstatně složitější problém. Podpora negativního cachingu je záležitostí posledních několika let.

Takto pracuje DNS na serverech (např. s operačním systémem NT nebo UNIX). Avšak např. PC nemívají realizovány servery. V takovém případě se celý mechanismus redukuje na tzv. pahýlový resolver. Tj. z celého mechanismu zůstane pouze resolver.

Resolver předává všechny dotazy na lokální name server. Od name serveru pak očekává konečnou (rekurzivní) odpověď. Name server buď odpoví přímo, nebo sám kontaktuje další name servery, tj. name server rekurzivně řeší dotaz a klientovi zašle až výsledek.

Lokální name server udržuje společnou cache pro všechny lokální počítače.

Obr. 11.6
Pahýlový resolver



Z obrázku 1.6 je patrné, že lokální name server udržuje společnou cache pro všechny lokální počítače s pahýlovým resolverem.

DNS používá jak protokol UDP, tak i protokol TCP. Pro oba protokoly používají port 53 (tj. porty 53/udp a 53/tcp). Běžné dotazy, jako je překlad jména na IP-adresu a naopak, se provádějí přes protokol UDP. Délka přenášených dat protokolem UDP je implicitně omezena na 512 B (příznakem *truncation* může být signalizováno, že se odpověď nevešla do 512 B a pro přídanou kompletní odpověď je nutné použít protokol TCP). Délka UDP paketu je omezena na 512 B, protože u větších IP-datagramů by mohlo dojít k fragmentaci. Fragmentaci UDP datagramu DNS nepovažuje za rozumnou.

Dotazy, kterými se přenášejí data o zóně (*zone transfer*) např. mezi primárním a sekundárním name serverem, se přenáší protokolem TCP.

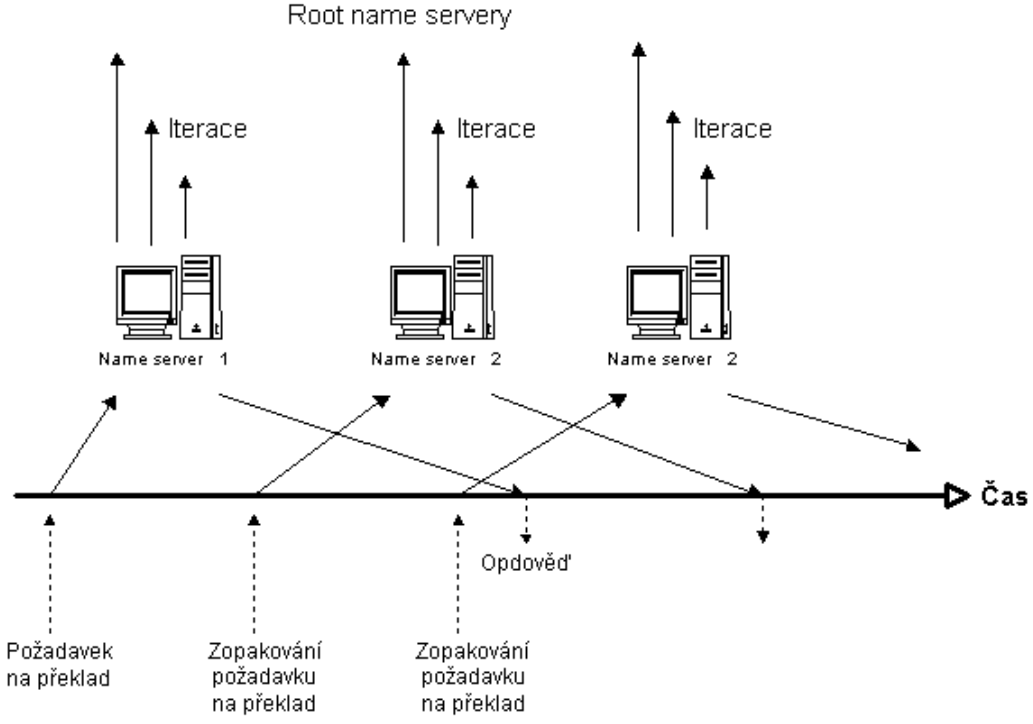
Běžné dotazy (např. překlad jména na IP-adresu a naopak) se provádí pomocí datagramů protokolu UDP. Překlad požaduje klient (resolver) na name serveru. Neví-li si name server rady, může požádat o překlad (o pomoc) jiný name server prostřednictvím root name serveru.

V Internetu platí pravidlo, že databáze s daty nutnými pro překlad jsou vždy uloženy alespoň na dvou nezávislých počítačích (nezávislých name serverech). Je-li jeden nedostupný, pak se překlad může provést na druhém počítači.

Obecně se nepředpokládá, že by byly všechny name servery dostupné. V případě, že by se pro překlad použil protokol TCP, pak by navazování spojení na nedostupný počítač znamenalo přečkat časové

intervaly protokolu TCP pro navázání spojení a teprve poté by bylo možno se pokusit navázat spojení s dalším name serverem.

Řešení pomocí protokolu UDP je elegantnější: Datagramem se vyšle žádost prvnímu serveru, nepřijde-li se odpověď do krátkého časového intervalu, pak se pošle datagramem žádost dalšímu (záložnímu name serveru), nepřijde-li se opět odpověď, pak se pošle dalšímu atd. V případě, že se vyčerpají všechny možné name servery, pak se opět začne prvním a celý kolotoč se zopakuje, dokud nepřijde odpověď nebo nevyprší stanovený časový interval.



Obr. 11.7 Řešení požadavku na překlad

11.9 Resolver

Resolver je komponenta systému zabývající se překladem IP-adresy. Resolver je klient. Resolver není konkrétní program. Je to soustava knihovních funkcí, která se sestavuje (linkuje) s aplikačními programy, požadujícími tyto služby (např. telnet, ftp, WWW-prohlížeč atd.). Tj. potřebuje-li např. telnet převést jméno počítače na jeho IP-adresu, pak zavolá příslušné knihovní funkce.

Klient (např. zmíněný telnet) zavolá knihovní funkce, které zformulují dotaz a vyšlou jej na server. Server je v UNIXu realizován programem *named*. Server buď překlad provede sám, nebo si sám vyžádá pomoc od dalších serverů, nebo zjistí, že překlad není možný.

Do hry ještě vstupují časová omezení. Může se totiž stát, že na položený dotaz nedostane resolver odpověď, ale další stejný dotaz již bude korektně zodpovězen (serveru se mezitím podařilo získat odpověď a první dotaz nebyl zodpovězen proto, že odpověď z jiného name serveru dlouho nepřicházela). Z hlediska uživatele se to jeví tak, že napoprvé se překlad nepovede a při dalším zadání téhož příkazu už ano.

Podobný efekt způsobuje i použití protokolu UDP. Může se totiž také stát, že server vůbec žádost o překlad neobdrží, protože je síť přetížená a UDP-datagram se prostě někde ztratil.

Klient může sice mít v konfiguračním souboru uvedeno více name serverů, ale použije se vždy jen odpověď, která přišla první. Tj. když jako první přijde negativní odpověď (např., že k danému jménu neexistuje IP-adresa), nepokusí se resolver kontaktovat další name server, který by jméno snad přeložil (jak si mnozí představují), ale oznámí, že překlad k danému jménu neexistuje.

Konfigurační soubor pro resolver se v operačním systému UNIX jmenuje /etc/resolv.conf. Zpravidla obsahuje dva typy řádků (druhý se může několikrát opakovat):

```
domain          jméno_místní_domény
nameserver      IP-adresa_name_serveru
```

V případě, že uživatel zadal jméno bez tečky na konci, pak resolver za zadané jméno přidá jméno domény z příkazu *domain* a pokusí se jméno předat name serveru k přeložení. V případě, že se překlad neprovede (negativní odpověď name serveru), pak se resolver pokusí ještě přeložit jméno samotné, tj. bez přípony z příkazu *domain*.

Některé resolvery umožňují zadat příkazem *search* více jmen místních domén.

Příkazem *nameserver* se specifikuje IP-adresa name serveru, který má resolver kontaktovat. Je možné uvést i další příkazy nameserver pro případ, že některé name servery jsou nedostupné. Musí se zde uvést IP-adresa name serveru – nikoliv doménové jméno name serveru!

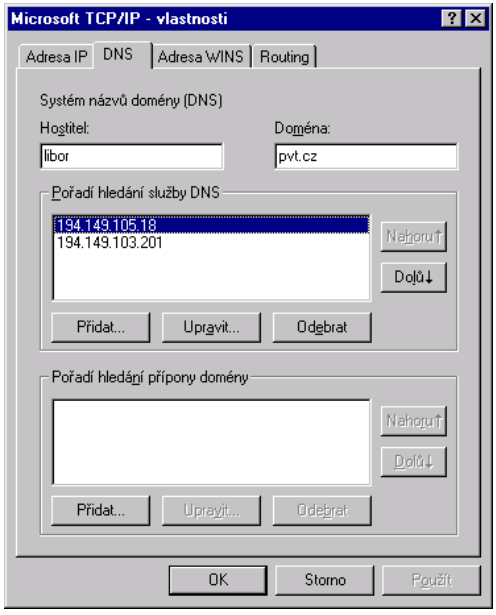
V případě konfigurace resolveru na name serveru může příkaz nameserver ukazovat na místní name server 127.0.0.1 (nemusí to však být pravidlem).

Další parametry resolveru (např. maximální počet příkazů nameserver) lze nastavit v konfiguračním souboru jádra. Tento soubor se často jmenuje */usr/include/resolv.h*. Musí pak pochopitelně následovat sestavení jádra operačního systému.

Obecně je možné konfigurovat všechny počítače též bez použití DNS. Pak se veškeré dotazy na překlady adres provádějí lokálně pomocí souboru */etc/hosts*. Je možné obě metody i kombinovat (nejčastější případ), pak však je třeba být opatrný na obsah databáze */etc/hosts*. Většinou je možné i nastavit v jakém pořadí se mají databáze prohlížet. Zpravidla se prohlíží nejprve soubor */etc/hosts* a posléze DNS. V DEC OSF/1 slouží pro konfiguraci pořadí prohledávání soubor */etc/svc.conf*.

V systému NT se resolver konfiguruje pomocí okna. Do pole doména vyplníme lokální doménu, která se bude doplňovat ke jménům v případě, že neuvedeme na konci tečku. Pakliže překlad s touto doplněnou doménou i bez ní selže, pak se systém pokusí ještě doplňovat domény z okna „Pořadí hledání přípony domény“.

Obr. 11.8
Konfigurace resolveru v NT



11.10 Name server

Name server udržuje informace pro překlad jmen počítačů na IP-adresy (resp. pro reverzní překlad). Name server obhospodařuje nějakou část z prostoru jmen všech počítačů. Tato část se nazývá zóna. Zóna je tvořena doménou nebo její částí. Name server totiž může pomocí věty typu NS ve své konfiguraci delegovat spravování subdomény na name server nižší úrovně.

Name server je program, který provádí na žádost resolveru překlad. V UNIXu je name server realizován programem *named*.

Podle uložení dat rozlišujeme následující typy name serverů:

- **Primární name server** udržuje data o své zóně v databázích na disku. Pouze na primárním name serveru má smysl editovat tyto databáze.
- **Sekundární name server** si kopíruje databáze v pravidelných časových intervalech z primárního name serveru. Tyto databáze nemá smysl na sekundárním name serveru editovat, neboť budou při dalším kopírování přepsány. Primární i sekundární name servery jsou tzv. autoritou pro své domény, tj. jejich data pro příslušnou zónu se považují za nezvratná (autoritativní).
- **Caching only server** není pro žádnou doménu ani primárním, ani sekundárním name serverem (není žádnou autoritou). Avšak využívá obecné vlastnosti name serveru, tj. data, která jím prochází, ukládá ve své paměti. Tato data se označují jako neautoritativní. Každý server je caching server, ale slovy caching only zdůrazňujeme, že pro žádnou zónu není ani primárním, ani sekundárním name serverem. (Pochopitelně i caching only server je primárním name serverem pro zónu 0.0.127.in-addr.arpa, ale to se nepočítá.)
- **Root name server** je name server obsluhující root doménu. Každý root name server je primárním serverem, což jej odlišuje od ostatních name serverů.

Jeden name server může být pro nějakou zónu primárním serverem, pro jiné sekundárním serverem.

Z hlediska klienta není žádný rozdíl mezi primárním a sekundárním name serverem. Oba mají data stejné důležitosti – oba jsou pro danou zónu autoritami. Klient nemusí ani vědět, který server pro zónu je primární a který sekundární. Naproti tomu caching server není autoritou, tj. nedokáže-li provést překlad, pak kontaktuje autoritativní server pro danou zónu.

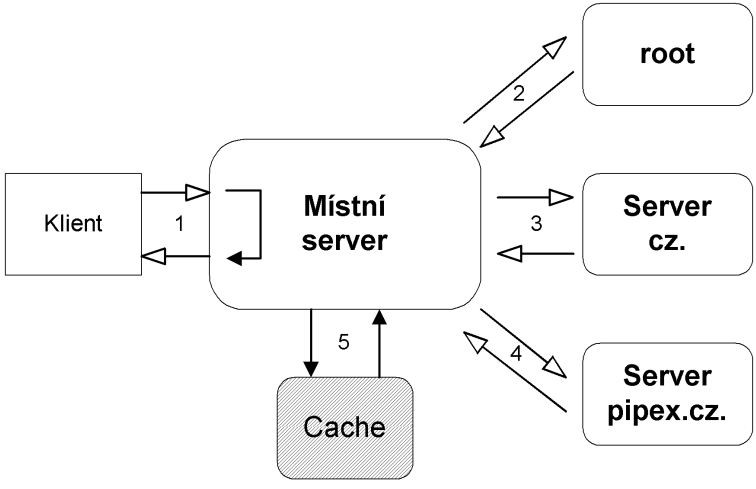
Takže přidá-li správce zóny (*hostmaster*) do databáze na primárním name serveru další počítač, pak po době stanovené parametrem ve větě SOA se tato databáze automaticky opraví i na sekundárních name serverech (opravil-li by ručně jen databázi na sekundárním name serveru, pak by po stejné době oprava zmizela!). Problém nastane v případě, že uživatel v době, kdy ještě není sekundární name server aktualizován, dostane první odpověď od sekundárního name serveru. Ta je negativní, tj. takový počítač v databázi není.

Klasickou chybou je, že primární name server pracuje korektně, ale na sekundárním name serveru z nějakého důvodu nejsou data pro zónu. Klienti náhodně dostávají autoritativní odpovědi ze sekundárního name serveru či z primárního name serveru. Odpovědi z primárního name serveru správně překládají, kdežto odpovědi ze sekundárního name serveru jsou negativní (uživatelé pak říkají: „jednou to jde a podruhé ne“).

Autoritativní data pocházejí z databází na disku. Je zde pouze jedna výjimka. Pro správnou činnost name serveru musí name server znát root name servery. Pro ty však není autoritou, přesto každý name server má na disku databázi informací o root serverech, kterou ale zavádí příkazem cache do sekundární paměti (není k nim autorita).

Na obr. 11.9 je překlad jména *abc.pipex.cz* na IP-adresu (nejedná se o forwarder nebo slave server):

Obr. 11.9
Překlad jména abc.pipex.cz na IP-adresu



1. Resolver zformuluje požadavek na name server a očekává jednoznačnou odpověď. Umí-li nameserver odpovědět, pak obratem zašle odpověď. Odpověď hledá ve své cache paměti (5). Tam jsou, jak autoritativní data z databází na disku, tak i neautoritativní data získaná při předešlých řešeních. Nezná-li server odpověď, pak kontaktuje další servery. Vždy začíná root name serverem.

Nezná-li name server přímo odpověď, pak kontaktuje root name server, proto každý name server musí znát IP-adresy root name serverů. Není-li však žádný root server dostupný (to je např. případ všech uzavřených sítí), pak po několika neúspěšných pokusech celý proces překladu zkolabuje.

Root name server zjistí, že informace o doméně cz delegoval větou typu NS name serveru nižší úrovně a zašle našemu name serveru IP-adresy serverů spravujících domény cz.

2. Name server se obrátí na server pro doménu cz, který však zjistí, že informace o doméně delegoval větou typu NS name serveru nižší úrovně a zašle našemu name serveru IP-adresy serverů spravujících domény pipex.cz.
3. Náš server se tedy obrátí na server spravující doménu pipex.cz, který mu požadavek vyřeší (nebo ne). Výsledek předá klientovi.
4. Informace které postupně získal si též uloží do cache.

Program *nslookup* je užitečný program pro správce name serveru. Chcete-li programem nslookup provádět dotazy jakoby name serverem, pak zakažte rekurence a přidávání doménových jmen příkazy:

```
$ nslookup
 set norecurse
 set nosearch
```

11.11 Forwarding a slave servery

Ještě existují dva typy serverů: forwarding a slave servery. Tato vlastnost serveru nesouvisí s tím, zda jsou primárními nebo sekundárními servery pro nějakou zónu, ale souvisí se způsobem jejich překladu.

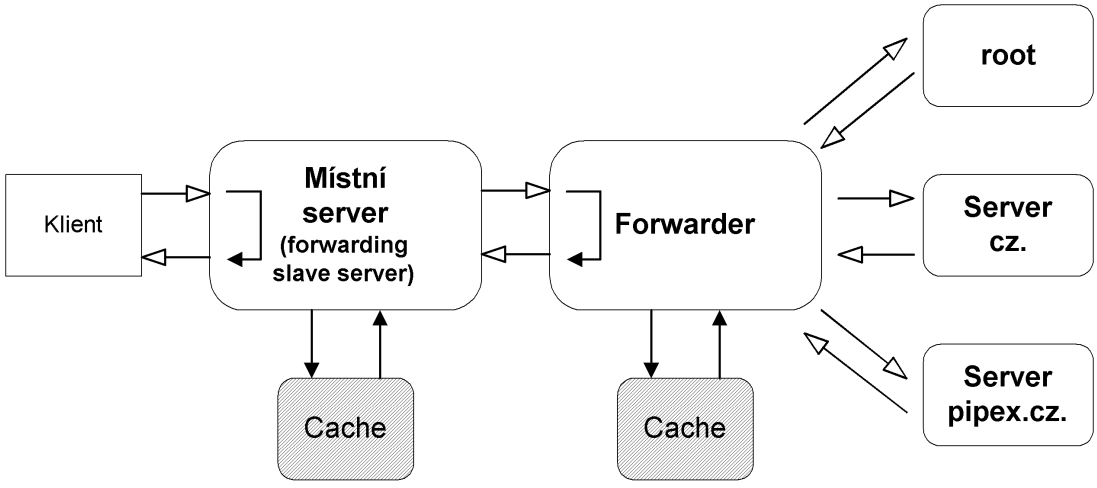
Zatím jsme si řekli, že resolver předává požadavek na překlad name serveru, tj. pošle name serveru dotaz a čeká na odpověď. Když name server neumí odpovědět sám, kontaktuje root name server, který mu poradí, pak z jeho rady kontaktuje další name server, pak ... . Name server posílá do Internetu mnoho paketů.

Je-li podniková síť připojena k Internetu pomalou linkou, pak místní name server zatěžuje linku svými překlady. V takovém případě je výhodné si name server konfigurovat jako **forwarding server** (viz obr. 11.10). Forwarding server vezme požadavek od klienta a předá jej forwarderovi na rychlé síti jako rekurzivní dotaz. Forwarder je server v Internetu, který je připojen rychlejšími linkami. Dotaz rekurzivně vyřeší a pošle mému forwarding serveru konečný výsledek. Jako forwarder je praktické použít name server posytovatele Internetu.

Forwarding server čeká na odpověď od forwardera. Nedočká-li se odpovědi v daném časovém intervalu, pak sám kontaktuje root name servery a pokouší se sám vyřešit překlad.

Nemá-li forwarding server kontaktovat root name servery, ale pouze čekat na odpověď od forwardera, pak je nutné označit takový server navíc jako **slave server**. Slave servery se používají v uzavřených podnikových sítích (za firewallem), kde není možný kontakt s root name servery. Slave server pak kontaktuje forwardera, který je součástí firewallu.

Slave server musí být forwarding server. Avšak jak forwarding server, tak slave server mohou být jak caching only servery, tak i primární nebo sekundární name servery pro určitou zónu.



Obr. 11.10 Forwarder server

11.12 Věty RR

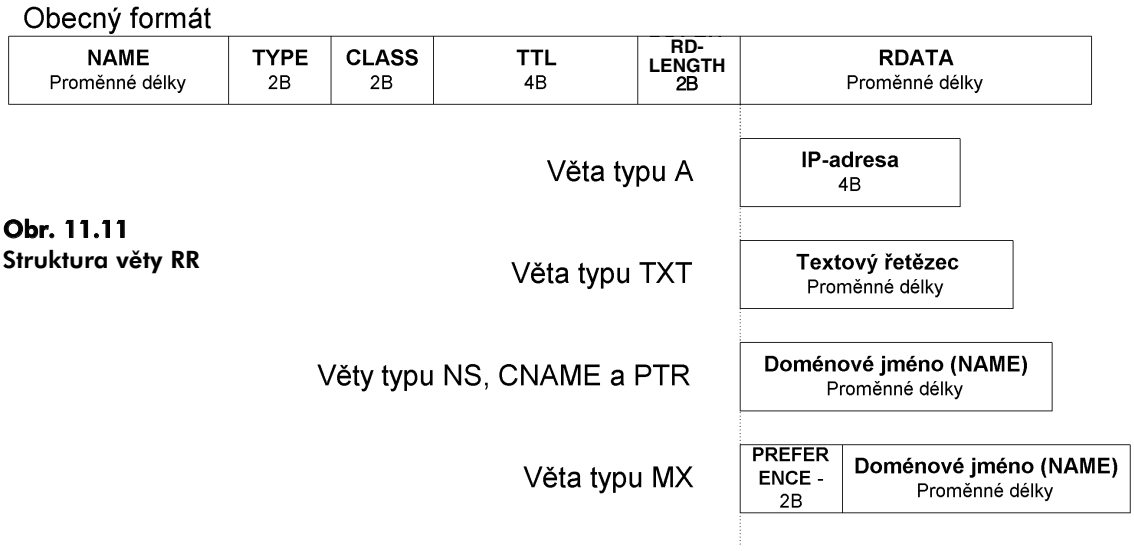
Informace o doménových jménech a jim příslušejících IP adresách, stejně tak jako všechny ostatní informace distrubuované pomocí DNS, jsou uloženy v paměti DNS serverů ve tvaru zdrojových vět (*Resource Records – RR*). Name server naplňuje svou paměť několika způsoby. Autoritativní data načte ze souborů na disku, nebo je získá pomocí dotazu *zone transfer* z paměti jiného serveru. Neautoritativní data získává postupně z paměti jiných serverů, tak jak vyřizuje jednotlivé DNS dotazy.

V případě, že DNS klient (resolver) potřebuje získat informace z DNS, pak požaduje po name serveru věty RR podle zadaných požadavků. Klient může např. požadovat po serveru věty RR typu A, které obsahují IP adresy pro dané doménové jméno apod.

Všechny věty RR mají stejnou strukturu. Struktura RR věty je znázorněna na obr. 11.11.

Jednotlivá pole vět RR obsahují:

- **NAME** – Doménové jméno.
- **TYPE** – Typ věty.
- **CLASS** – Třída věty.
- **TTL** – Time to live. 32bitové číslo udávající dobu, po kterou může být tento RR udržován v cache serveru jako platný. Po vypršení této doby musí být věta považována za neplatnou. Hodnota 0 zabraňuje neautoritativním serverům uložit RR větu do cache.
- **RDLENGTH** – 16bitové číslo specifikující délku pole RDATA.
- **RDATA** – Vlastní data ve tvaru řetězce proměnné délky. Formát tohoto pole závisí na typu a třídě RR.



Obr. 11.11
Struktura věty RR

Typ	Anglický název	Význam pole RDATA
A	A host address	32bitová IP adresa
NS	Authoritative name server	Doménové jméno name serveru, který je autoritou pro danou doménu.
CNAME	Canonical name for an alias	Doménové jméno specifikující synonymum k NAME.
SOA	Start Of Autority.	Právě jedna věta SOA uvozuje každou zónu. Obsahuje 7 polí. Přesný popis viz DNS databáze.
PTR	Domain name pointer	Doménové jméno. Věta se používá pro reverzní překlad.
HINFO	Host information	Obsahuje dva znakové řetězce. První obsahuje popis HW a druhý popis SW, který je používán na počítači NAME.
MX	Mail exchange	Obsahuje dvě pole. První 16bitové pole bez znaménka obsahuje preferenci a druhé obsahuje doménové jméno mailového serveru.
TXT	Text string	Textový řetězec s popisem.
AAAA	IP6 address	128bitová IP adresa (IP verze 6)
WKS	Well known service description	Popisuje obvyklé služby serveru v protokolech TCP a UDP. Obsahuje 3 části: 32bitovou adresu, číslo protokolu, porty služeb.
SIG	Security signature	Podpisová věta používaná při autentizaci v Secure DNS.
KEY	Security key	Veřejný klíč zóny používaný pro podepisování při autentizaci.
NXT	Next domain	Další doménové jméno. Autentikace neexistence doménového jména a typu.

Tab. 11.1 Nejčastější typy vět RR

11.13 Protokol DNS

Doménová služba je realizována jednoduchým protokolem. Tento protokol pracuje způsobem dotaz – odpověď. Klient pošle dotaz serveru a server na dotaz odpoví. Jistou komplikací je komprese jmen, která se provádí proto, aby byly DNS pakety co nejúspornější.

Protokol DNS je protokol aplikační vrstvy, neřeší tedy otázku vlastního přenosu paketů. Přenos svých paketů svěřuje transportnímu protokolu. Na rozdíl od drtivé většiny ostatních aplikačních protokolů využívá DNS jako transportní protokoly UDP i TCP. Dotaz i odpověď jsou přenášeny vždy stejným transportním protokolem.

U dotazů na překlad (tj. žádosti o RR record) je dávána přednost protokolu UDP. V případě, že je DNS odpověď delší než 512 B, vloží se do odpovědi pouze část informací nepřesahující 512 B a v záhlaví se nastaví bit TC, specifikující, že se jedná o neúplnou odpověď. Klient si může kompletní odpověď vyžádat protokolem TCP.

U přenosu zón např. mezi primárním a sekundárním name serverem se používá protokol TCP. Name server standardně očekává dotazy jak na portu 53/udp, tak na portu 53/tcp.

Poznámka:

U protokolu UDP je třeba upozornit, že některé implementace protokolu UDP využívají možnosti nevyplňovat pole pro kontrolní součet v záhlaví UDP paketu. Tato vlastnost může být užitečná např. pro NFS, ale pro DNS je nebezpečná. Porucha sítě tak může způsobit nesmyslnou odpověď, zejména je-li na cestě mezi serverem a klientem použit např. linkový protokol SLIP. Zkontrolujte si proto před instalací name serveru, zdali váš systém vyplňuje kontrolní součet v UDP paketu.

11.14 DNS query

DNS protokol používá několik typů operací. Standardní nejčastěji používanou operací je DNS QUERY. Tedy získání RR rekordu. S novými rozšířeními protokolu DNS přibývají i nové typy operací,. např. DNS NOTIFY nebo DNS UPDATE.

11.14.1 Formát DNS paketu

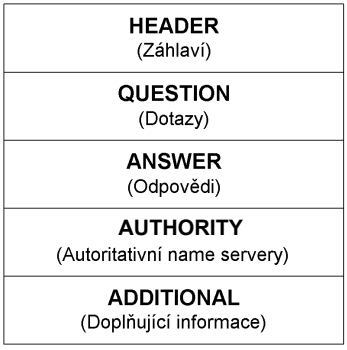
DNS používá stejný formát paketu pro dotaz i odpověď. Paket se může skládat až z pěti sekcí, vždy musí obsahovat sekci záhlaví (HEADER).

11.14.2 Záhlaví paketu

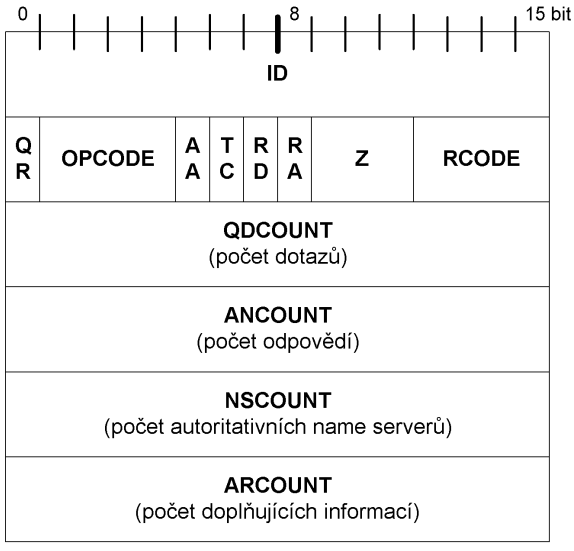
Záhlaví paketu je povinné, je obsaženo v dotazu i odpovědi.

První dva bajty (16 bitů) záhlaví obsahují identifikátor zprávy. Identifikaci zprávy generuje klient a server ji kopíruje do odpovědi. Identifikace slouží k párování dotazu a odpovědi. Jednoznačně určuje, ke kterému dotazu patří která odpověď. Identifikace umožňuje klientovi posílat více dotazů současně, aniž by musel čekat na odpověd.
Další dva bajty záhlaví obsahují řídící bity. Tabulka 11.2 uvádí význam jednotlivých hodnot těchto řídících bitů.

Obr. 11.12
Formát paketu protokolu DNS



Obr. 11.13
Pole záhlaví DNS paketu



Další čtyři dvojbajtová pole obsahují počet vět obsažených v sekcích, které následují za záhlavím.

- **QDCOUNT –** číslo určující, z kolika vět se skládá dotaz
- **ANCOUNT** – číslo určující, z kolika vět se skládá odpověď
- **NSCOUNT** – číslo určující, z kolika vět se skládá sekce obsahující odkazy na autoritativní name servery
- **ARCOUNT** – číslo určující, z kolika vět se skládá sekce doplňující informace

Příklad DNS paketu odchyceného na síti je uveden v tab. 11.3.

Pole	Počet bitů	Hodnota
QR	1	0 pokud je zpráva dotazem 1 pokud je zpráva odpovědí
Opcode	4	Typ otázky je stejný v dotazu i odpovědi: 0 – standardní otázka (QUERY) 1 – inverzní otázka (IQUERY) 2 – otázka na status (STATUS) 4 – notify otázka (NOTIFY) 5 – update otázka (UPDATE)
AA	1	0 – odpověď není autoritativní 1 – odpověď je autoritativní
TC	1	1 – odpověď byla zkrácena na 512 bajtů. Pokud má klient zájem o celou odpověď, pak musí dotaz zopakovat pomocí protokolu TCP.
RD	1	1 – pokud klient požaduje rekurzivní překlad (důležité pro dotaz)
RA	1	1 – pokud server umožňuje rekurzivní překlad (důležité pro odpověď)
Z	3	rezervováno pro budoucí použití
Rcode	4	Výsledkový kód odpovědi 0 – Bez chyby *(Noerror)* 1 – Chyba ve formátu dotazu, server jej neumí interpretovat *(FormErr)* 2 – Server neumí odpovědět *(ServFail)* 3 – Jméno z dotazu neexistuje (tj. negativní odpověď), tuto odpověď mohou vydat pouze autoritativní name servery *(NXDomain)* 4 – Server nepodporuje tento typ dotazu *(NotImp)* 5 – Server odmítá odpovědět, např. z bezpečnostních důvodů *(Refused)*

Tab. 11.2 Význam jednotlivých řídících bitů ze záhlaví DNS paketu

Tab. 11.3 Příklad DNS paketu odchyceného na síti

```
+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP: DOD Internet Protocol
+ IP: ID = 0xF126; Proto = UDP; Len: 253
+ UDP: Src Port: DNS, (53); Dst Port: Unknown (1143); Length = 233 (0xE9)
  DNS: 0x3:Std Qry Resp. for snmp0.pvt.net. of type Host Addr on class INET addr.
      DNS: Query Identifier = 3 (0x3)
      DNS: DNS Flags = Response, OpCode - Std Qry, AA RD RA Bits Set, RCode - No error
          DNS: 1............... = Response
          DNS: .0000........... = Standard Query
          DNS: .....1.......... = Server authority for domain
          DNS: ......0......... = Message complete
          DNS: .......1........ = Recursive query desired
```

```
          DNS: ........1....... = Recursive queries supported by server
          DNS: .........000.... = Reserved
          DNS: ............0000 = No error
      DNS: Question Entry Count = 1 (0x1)
      DNS: Answer Entry Count = 1 (0x1)
      DNS: Name Server Count = 5 (0x5)
      DNS: Additional Records Count = 5 (0x5)
      DNS: Question Section: snmp0.pvt.net. of type Host Addr on class INET addr.
          DNS: Question Name: snmp0.pvt.net.
          DNS: Question Type = Host Address
          DNS: Question Class = Internet address class
    + DNS: Answer section: snmp0.pvt.net. of type Host Addr on class INET addr.
    + DNS: Authority Section: pvt.net. of type Auth. NS on class INET addr.(5 records pre-
           sent)
      DNS: Additional Records Section: ns.pvt.net. of type Host Addr on class INET …
        + DNS: Resource Record: ns.pvt.net. of type Host Addr on class INET addr.
        + DNS: Resource Record: ns1.pvt.net. of type Host Addr on class INET addr.
        + DNS: Resource Record: snmp0.pvt.net. of type Host Addr on class INET addr.
        + DNS: Resource Record: ns0.pipex.net. of type Host Addr on class INET addr.
        + DNS: Resource Record: ns1.pipex.net. of type Host Addr on class INET addr.
 00000:  00 20 AF F9 2F A0 00 60 3E 1D 90 00 08 00 45 00   . ../..`.....E.
 00010:  00 FD F1 26 00 00 1D 11 54 C7 C2 95 69 12 C2 95   ...&....T...i...
 00020:  68 C5 00 35 04 77 00 E9 8E 41 00 03 85 80 00 01   h..5.w...A......
 00030:  00 01 00 05 00 05 05 73 6E 6D 70 30 03 70 76 74   .......snmp0.pvt
 00040:  03 6E 65 74 00 00 01 00 01 C0 0C 00 01 00 01 00   .net............
```

11.14.3 Sekce dotaz (Question section)

Pakety DNS dotazů obsahují většinou pouze jednu sekci, a to sekci dotazu (QDCOUNT=1). Sekce dotazu obsahuje tři pole:

QNAME – obsahuje doménové jméno. Protokol DNS nepoužívá pro vyjádření doménového jména tečkovou notaci. Každá část doménového jména (v běžném zápisu mezi tečkami) je uvozena bajtem obsahujícím délku řetězce. Na konci doménového jména je nula označující konec doménového jména (nulová délka řetězce). Příklad obsahu tohoto pole v dotazu na překlad doménového jména info.pvt.net: 4info3pvt3net0. Délky řetězce jsou v binárním tvaru.

QTYPE – specifikuje typ dotazu, tj. požadovaný typ věty v odpovědi.

Nejčastější typy dotazu uvádím v tabulce 11.4.

QCLASS – třída dotazu (viz tab. 11.5)

Tab. 11.4 Hodnoty typů dotazů

Typ	Hodnota (desítkově)	Význam
A	1	Požadavek na získání IP adresy verze 4
NS	2	Požadavek na získání autoritativních name serverů
CNAME	5	Požadavek na získání věty CNAME
SOA	6	Požadavek na získání věty SOA
WKS	11	Požadavek na získání věty WKS
PTR	12	Požadavek na získání PTR věty
HINFO	13	Požadavek na získání HINFO věty
MX	15	Požadavek na získání věty MX
TXT	16	Požadavek na získání TXT věty
SIG	24	Požadavek na získání věty SIG
KEY	25	Požadavek na získání věty KEY
NXT	30	Požadavek na získání věty NXT
AAAA	28	Požadavek na získání IP adresy verze 6
SRV	33	Požadavek na získání věty SRV
AXFR	252	Požadavek na získání transferu celé zony
IXFR		Požadavek na získání inkremetálního zóne transferu
*	255	Požadavek na získání všech vět

Tab. 11.5 Jednotlivé třídy

Číselná hodnota (desítkově)	Význam
1	IN – Internet
3	CH – Chaos
4	HS – Hesiod
255	* – všechny třídy (pouze jako QCLASS)

Příklad DNS paketu odchyceného na síti je uveden v tab. 11.6. Sekce dotazu je v příkladu zvýrazněna tučně.

Tab. 11.6 Příklad paketu se zvýrazněnou sekcí dotazu

```
+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP: DOD Internet Protocol
+ IP: ID = 0xF126; Proto = UDP; Len: 253
+ UDP: Src Port: DNS, (53); Dst Port: Unknown (1143); Length = 233 (0xE9)
  DNS: 0x3:Std Qry Resp. for snmp0.pvt.net. of type Host Addr on class INET addr.
      DNS: Query Identifier = 3 (0x3)
    + DNS: DNS Flags = Response, OpCode - Std Qry, AA RD RA Bits Set, RCode - No error
      DNS: Question Entry Count = 1 (0x1)
      DNS: Answer Entry Count = 1 (0x1)
      DNS: Name Server Count = 5 (0x5)
```

```
      DNS: Additional Records Count = 5 (0x5)
      DNS: Question Section: snmp0.pvt.net. of type Host Addr on class INET addr.
          DNS: Question Name: snmp0.pvt.net.
          DNS: Question Type = Host Address
          DNS: Question Class = Internet address class
    + DNS: Answer section: snmp0.pvt.net. of type Host Addr on class INET addr.
    + DNS: Authority Section: pvt.net. of type Auth. NS on class INET addr.(5 records pre-
          sent)
      DNS: Additional Records Section: ns.pvt.net. of type Host Addr on class INET ….
        + DNS: Resource Record: ns.pvt.net. of type Host Addr on class INET addr.
        + DNS: Resource Record: ns1.pvt.net. of type Host Addr on class INET addr.
        + DNS: Resource Record: snmp0.pvt.net. of type Host Addr on class INET addr.
        + DNS: Resource Record: ns0.pipex.net. of type Host Addr on class INET addr.
        + DNS: Resource Record: ns1.pipex.net. of type Host Addr on class INET addr.

00000:  00 20 AF F9 2F A0 00 60 3E 1D 90 00 08 00 45 00   . ../..`.....E.
00010:  00 FD F1 26 00 00 1D 11 54 C7 C2 95 69 12 C2 95   ...&....T...i...
00020:  68 C5 00 35 04 77 00 E9 8E 41 00 03 85 80 00 01   h..5.w...A......
00030:  00 01 00 05 00 05 05 73 6E 6D 70 30 03 70 76 74   .......snmp0.pvt
   00040:  03 6E 65 74 00 00 01 00 01 C0 0C 00 01 00 01 00   .net............
```

11.14.4 Sekce odpověď, autoritativní servery a doplňující informace

Pakety odpovědi obsahují obvykle vedle záhlaví a zopakované sekce dotazu ještě tři sekce: sekci odpovědi, sekci autoritativních serverů a sekci doplňujících informací. Sekce autoritativní name servery obsahuje jména name serverů uvedených ve větách NS. Sekce doplňkové údaje obsahuje obvykle IP adresy autoritativních name serverů. Věty v těchto sekcích jsou běžné resource recordy (RR) – obdobné větám v cache name serveru a mají společný formát:

NAME – doménové jméno, stejný formát jako v sekci dotazu QNAME
TYPE – typ věty, stejný formát jako v sekci dotazu QTYPE
CLASS – třída věty, stejný formát jako v sekci dotazu QCLASS
TTL – doba platnosti RR, tj. jak dlouho může být odpověď udržována jako platná v cache.
RDLENGTH – délka části RDATA
RDATA – pravá strana zdrojové věty (IP adresa nebo doménové jméno)

Příklad DNS paketu odchyceného na síti je v tab. 11.7.

Tab. 11.7 Příklad DNS paketu se sekcemi odpověď, autoritativní name servery a doplňující informace

```
+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
+ IP: ID = 0xF126; Proto = UDP; Len: 253
+ UDP: Src Port: DNS, (53); Dst Port: Unknown (1143); Length = 233 (0xE9)
  DNS: 0x3:Std Qry Resp. for snmp0.pvt.net. of type Host Addr on class INET addr.
      DNS: Query Identifier = 3 (0x3)
    + DNS: DNS Flags = Response, OpCode - Std Qry, AA RD RA Bits Set, RCode - No error
      DNS: Question Entry Count = 1 (0x1)
```

```
      DNS: Answer Entry Count = 1 (0x1)
      DNS: Name Server Count = 5 (0x5)
      DNS: Additional Records Count = 5 (0x5)
    + DNS: Question Section: snmp0.pvt.net. of type Host Addr on class INET addr.
      DNS: Answer section: snmp0.pvt.net. of type Host Addr on class INET addr.
          DNS: Resource Name: snmp0.pvt.net.
          DNS: Resource Type = Host Address
          DNS: Resource Class = Internet address class
          DNS: Time To Live = 86400 (0x15180)
          DNS: Resource Data Length = 4 (0x4)
          DNS: IP address = 194.149.103.34
      DNS: Authority Section: pvt.net. of type Auth. NS on class INET addr.(5 records pre-
           sent)
        + DNS: Resource Record: pvt.net. of type Auth. NS on class INET addr.
        + DNS: Resource Record: pvt.net. of type Auth. NS on class INET addr.
        + DNS: Resource Record: pvt.net. of type Auth. NS on class INET addr.
        + DNS: Resource Record: pvt.net. of type Auth. NS on class INET addr.
        + DNS: Resource Record: pvt.net. of type Auth. NS on class INET addr.
      DNS: Additional Records Section: ns.pvt.net. of type Host Addr on class INET …
        + DNS: Resource Record: ns.pvt.net. of type Host Addr on class INET addr.
        + DNS: Resource Record: ns1.pvt.net. of type Host Addr on class INET addr.
        + DNS: Resource Record: snmp0.pvt.net. of type Host Addr on class INET addr.
        + DNS: Resource Record: ns0.pipex.net. of type Host Addr on class INET addr.
        + DNS: Resource Record: ns1.pipex.net. of type Host Addr on class INET addr.

00000:  00 20 AF F9 2F A0 00 60 3E 1D 90 00 08 00 45 00   . ../..`.....E.
00010:  00 FD F1 26 00 00 1D 11 54 C7 C2 95 69 12 C2 95   ...&....T...i...
00020:  68 C5 00 35 04 77 00 E9 8E 41 00 03 85 80 00 01   h..5.w...A......
00030:  00 01 00 05 00 05 05 73 6E 6D 70 30 03 70 76 74   .......snmp0.pvt
00040:  03 6E 65 74 00 00 01 00 01 C0 0C 00 01 00 01 00   .net............
00050:  01 51 80 00 04 C2 95 67 22 03 70 76 74 03 6E 65   .Q.....g".pvt.ne
00060:  74 00 00 02 00 01 00 01 51 80 00 05 02 6E 73 C0   t.......Q....ns.
00070:  2F C0 2F 00 02 00 01 00 01 51 80 00 06 03 6E 73   /./......Q....ns
00080:  31 C0 2F C0 2F 00 02 00 01 00 01 51 80 00 02 C0   1././......Q....
00090:  0C C0 2F 00 02 00 01 00 01 51 80 00 0C 03 6E 73   ../......Q....ns
000A0:  30 05 70 69 70 65 78 C0 33 C0 2F 00 02 00 01 00   0.pipex.3./.....
000B0:  01 51 80 00 06 03 6E 73 31 C0 77 C0 42 00 01 00   .Q....ns1.w.B...
000C0:  01 00 01 51 80 00 04 C2 95 69 12 C0 53 00 01 00   ...Q.....i..S...
000D0:  01 00 01 51 80 00 04 C2 95 67 C9 C0 0C 00 01 00   ...Q.....g......
000E0:  01 00 01 51 80 00 04 C2 95 67 22 C0 73 00 01 00   ...Q.....g".s...
000F0:  01 00 00 5E 23 00 04 9E 2B 80 08 C0 8B 00 01 00   ...^#...+.......
00100:  01 00 00 5E 24 00 04 9E 2B C0 07                 ...^$...+..
```

Sekce odpovědi je v předchozím příkladu zvýrazněna tučně. Sekce dopňujících informací je vyznačena tučnou kurzívou.

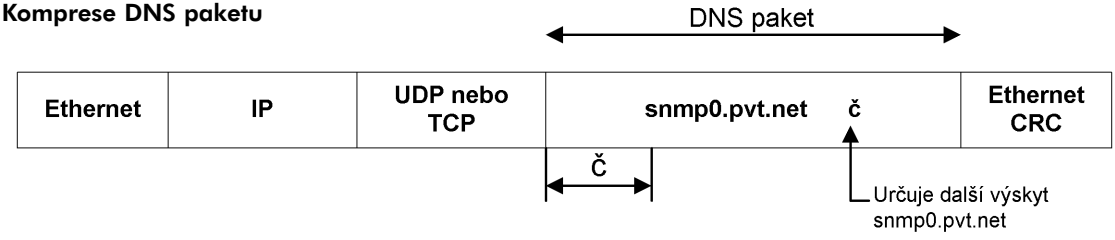
11.14.5 Komprese

Komprese slouží k redukci velikosti DNS paketu. V DNS paketu se může stejné doménové jméno nebo jeho část vyskytovat opakovaně. Komprese spočívá v tom, že je toto opakující se jméno uvedeno pouze jednou a každý další výskyt tohoto jména je nahrazen ukazovátkem na první výskyt.

Doménové jméno, jak jsem již uvedl, není v DNS paketu uvedeno v tečkové notaci, ale jako oddělovač jednotlivých částí doménového jména je použito číslo určující délku následující části. Tento oddělovač je uložen v jednou bajtu. Každá část doménového jména může být dlouhá maximálně 63 znaků, tedy oddělovací bajt bude mít maximální hodnotu 63 vyjádřenou dvojkově $00111111_2 = 63_{10}$. Pokud je v tomto bajtu číslo 192 nebo větší, pak je to příznak toho, že nebude uvedeno doménové jméno, ale pouze odkaz na jeho předcházející výskyt. Ukazatel je dlouhý 16 bitů. V prvních dvou bitech obsahuje jedničku, čímž se odlišuje od běžného odělovače. V dalších bitech pak obsahuje pořadové číslo bajtu od začátku DNS paketu, kde začíná doménové jméno, na které má odkaz ukazovat.

Posun 0 by ukazoval na první bajt, tj. pole ID v sekci záhlaví.

Obr. 11.14
Komprese DNS paketu



Příklad DNS paketu odchyceného v síti je uveden v tab. 11.8. DNS paket je vytištěn tučně. V paketu se opakuje doménové jméno snmp0.pvt.net. Jeho první výskyt v sekci dotaz je podtržen. Odkazy na tento výskyt jsou v dalších sekcích rovněž podtrženy.

Tab. 11.8 Příklad DNS paketu s kompresí záhlaví (DNS paket je uveden tučně)

```
+ FRAME: Base frame properties
 + ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
 + IP: ID = 0xF126; Proto = UDP; Len: 253
 + UDP: Src Port: DNS, (53); Dst Port: Unknown (1143); Length = 233 (0xE9)
   DNS: 0x3:Std Qry Resp. for snmp0.pvt.net. of type Host Addr on class INET addr.
       DNS: Query Identifier = 3 (0x3)
     + DNS: DNS Flags = Response, OpCode - Std Qry, AA RD RA Bits Set, RCode - No error
       DNS: Question Entry Count = 1 (0x1)
       DNS: Answer Entry Count = 1 (0x1)
       DNS: Name Server Count = 5 (0x5)
       DNS: Additional Records Count = 5 (0x5)
       DNS: Question Section: snmp0.pvt.net. of type Host Addr on class INET addr.
           DNS: Question Name: snmp0.pvt.net.
           DNS: Question Type = Host Address
           DNS: Question Class = Internet address class
```

```
       DNS: Answer section: snmp0.pvt.net. of type Host Addr on class INET addr.
           DNS: Resource Name: snmp0.pvt.net.
           DNS: Resource Type = Host Address
           DNS: Resource Class = Internet address class
          DNS: Time To Live = 86400 (0x15180)
           DNS: Resource Data Length = 4 (0x4)
           DNS: IP address = 194.149.103.34
     + DNS: Authority Section: pvt.net. of type Auth. NS on class INET addr.(5 records pre-
sent)
       DNS: Additional Records Section: ns.pvt.net. of type Host Addr on class INET …
         + DNS: Resource Record: ns.pvt.net. of type Host Addr on class INET addr.
         + DNS: Resource Record: ns1.pvt.net. of type Host Addr on class INET addr.
         + DNS: Resource Record: snmp0.pvt.net. of type Host Addr on class INET addr.
         + DNS: Resource Record: ns0.pipex.net. of type Host Addr on class INET addr.
         + DNS: Resource Record: ns1.pipex.net. of type Host Addr on class INET addr.

 00000:  00 20 AF F9 2F A0 00 60 3E 1D 90 00 08 00 45 00   . ../..`.....E.
 00010:  00 FD F1 26 00 00 1D 11 54 C7 C2 95 69 12 C2 95   ...&....T...i...
 00020:  68 C5 00 35 04 77 00 E9 8E 41 00 03 85 80 00 01   h..5.w...A......
 00030:  00 01 00 05 00 05 05 73 6E 6D 70 30 03 70 76 74   .......snmp0.pvt
 00040:  03 6E 65 74 00 00 01 00 01 C0 0C 00 01 00 01 00   .net............
 00050:  01 51 80 00 04 C2 95 67 22 03 70 76 74 03 6E 65   .Q.....g".pvt.ne
 00060:  74 00 00 02 00 01 00 01 51 80 00 05 02 6E 73 C0   t.......Q....ns.
 00070:  2F C0 2F 00 02 00 01 00 01 51 80 00 06 03 6E 73   /./......Q....ns
 00080:  31 C0 2F C0 2F 00 02 00 01 00 01 51 80 00 02 C0   1././......Q....
 00090:  0C C0 2F 00 02 00 01 00 01 51 80 00 0C 03 6E 73   ../......Q....ns
 000A0:  30 05 70 69 70 65 78 C0 33 C0 2F 00 02 00 01 00   0.pipex.3./.....
 000B0:  01 51 80 00 06 03 6E 73 31 C0 77 C0 42 00 01 00   .Q....ns1.w.B...
 000C0:  01 00 01 51 80 00 04 C2 95 69 12 C0 53 00 01 00   ...Q.....i..S...
 000D0:  01 00 01 51 80 00 04 C2 95 67 C9 C0 0C 00 01 00   ...Q.....g......
 000E0:  01 00 01 51 80 00 04 C2 95 67 22 C0 73 00 01 00   ...Q.....g".s...
 000F0:  01 00 00 5E 23 00 04 9E 2B 80 08 C0 8B 00 01 00   ...^#...+.......
 00100:  01 00 00 5E 24 00 04 9E 2B C0 07                  ...^$...+..
```

Obsah ukazatele na doménové jméno je hexadecimálně $C00C_{16} = 1100000000000111_2$. Pořadové číslo bajtu v paketu, v němž začíná první výskyt doménového jména, je tedy $00000000000111_2 = 12_{10}$. Připomeňme, že první bajt má pořadové číslo 0, tedy doménové jméno najdeme od 13. bajtu v DNS paketu. Je třeba si uvědomit, že v našem příkladu jsem nezachytil pouze DNS paket, ale celý rámec, který byl poslán sítí. Vlastní DNS paket začíná v příkladu jedenáctým bajtem na 3. řádku (00 03 85 80 ...).

Sami se můžete pokusit vyhledat další případ komprese v tomto paketu. Napovím, že jde o odkaz na řetězec pvt.net.

11.14.6 Inverzní dotaz

Inverzní dotaz se nesmí zaměňovat s reverzním dotazem. Při inverzním dotazu se také např. překládá IP adresa opět na jméno, ale k vyhledání se použijí věty typu A. Při reverzním překladu se používají věty typu PTR.

Inverzní dotazy nemusí být name serverem podporovány. Jejich specifikace je uvedena v RFC 1035.

11.14.7 Příklady komunikace

Pro ilustraci uvedu několik příkladů komunikace DNS klienta a DNS serveru. V jednotlivých paketech vynechám hexadecimální vyjádření, aby byly příklady přehlednější. V jednotlivých paketech si všimněte hlavně jejich záhlaví.

1. Příklad dotazu a odpovědi na neexistující RR větu:

K dotazu na překlad jména aaa.abc.cz jsem použil program nslookup, požadoval jsem konečnou (rekurzivní) odpověď. Použití programu nslookup pro získání překladu způsobilo poslání dvou paketů – dotazu a odpovědi.

```
# nslookup
Default Server:  localhost
Address:  127.0.0.1

> aaa.abc.cz
Server:  localhost
Address:  127.0.0.1


*** localhost can't find aaa.abc.cz: Non-existent host/domain
>

DNS dotaz:
 + FRAME: Base frame properties
 + ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
 + IP: ID = 0x3186; Proto = UDP; Len: 56
 + UDP: Src Port: Unknown, (1258); Dst Port: DNS (53); Length = 36 (0x24)
   DNS: 0x14:Std Qry for aaa.abc.cz. of type Host Addr on class INET addr.
       DNS: Query Identifier = 20 (0x14)
       DNS: DNS Flags = Query, OpCode - Std Qry, RD Bits Set, RCode - No error
           DNS: 0.............. = Query
           DNS: .0000.......... = Standard Query
           DNS: .....0..... = Server not authority for domain
           DNS: ......0..... = Message complete
           DNS: .......1..... = Recursive query desired
           DNS: ........0..... = No recursive queries
           DNS: .........000.... = Reserved
           DNS: ............0000 = No error
       DNS: Question Entry Count = 1 (0x1)
       DNS: Answer Entry Count = 0 (0x0)
       DNS: Name Server Count = 0 (0x0)
       DNS: Additional Records Count = 0 (0x0)
       DNS: Question Section: aaa.abc.cz. of type Host Addr on class INET addr.
```

```
          DNS: Question Name: aaa.abc.cz.

          DNS: Question Type = Host Address
          DNS: Question Class = Internet address class

DNS odpověď:
 + FRAME: Base frame properties
 + ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
 + IP: ID = 0x9D43; Proto = UDP; Len: 56
 + UDP: Src Port: DNS, (53); Dst Port: Unknown (1258); Length = 36 (0x24)
   DNS: 0x14:Std Qry Resp. : Name does not exist
       DNS: Query Identifier = 20 (0x14)
       DNS: DNS Flags = Response, OpCode - Std Qry, AA RD RA Bits Set, RCode - Name does not
            exist
           DNS: 1............... = Response
           DNS: .0000........... = Standard Query
           DNS: .....1.......... = Server authority for domain
           DNS: ......0......... = Message complete
           DNS: .......1........ = Recursive query desired
           DNS: ........1....... = Recursive queries supported by server
           DNS: .........000.... = Reserved
           DNS: ............0011 = Name does not exist
       DNS: Question Entry Count = 1 (0x1)
       DNS: Answer Entry Count = 0 (0x0)
       DNS: Name Server Count = 0 (0x0)
       DNS: Additional Records Count = 0 (0x0)
       DNS: Question Section: aaa.abc.cz. of type Host Addr on class INET addr.
           DNS: Question Name: aaa.abc.cz.
           DNS: Question Type = Host Address
           DNS: Question Class = Internet address class
```

2. Příklad komunikace s root serverem:

Pomocí programu nslookup požaduji po jednom z root serverů rekurzivní překlad jména www.lackfix.cz. Root servery jsou nakonfigurovány tak, aby rekurzivní překlady neprováděly. Výsledkem tedy je pouze získání jmen a IP autoritativních serverů pro TLD cz. V příkladu je podrobně rozepsán i obsah záhlaví IP paketu, ze kterého je zřejmá IP adresa root serveru.

```
# nslookup
Default Server:  localhost
Address:  127.0.0.1

> server a.root-servers.net
Default Server:  a.root-servers.net
Address:  198.41.0.4

> set recurse
> www.lackfix.cz
Server:  a.root-servers.net
Address:  198.41.0.4

Name:    www.lackfix.cz
```

```
Served by:
- NS.CESNET.CZ
          192.108.150.10
          CZ
- NS.EU.NET
          192.16.202.11
          CZ
- SUNIC.SUNET.SE
          192.36.125.2, 192.36.148.18
          CZ
- NS.UU.NET
          137.39.1.3
          CZ
- SPARKY.ARL.MIL
          128.63.58.18
          CZ
- NS.EUNET.CZ
          193.85.1.12
          CZ

>
```

DNS dotaz:

```
 + FRAME: Base frame properties
 + ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
   IP: ID = 0x1E88; Proto = UDP; Len: 60       IP: Version = 4 (0x4)
       IP: Header Length = 20 (0x14)     + IP: Service Type = 0 (0x0)
       IP: Total Length = 60 (0x3C)       IP: Identification = 7816 (0x1E88)
     + IP: Flags Summary = 0 (0x0)       IP: Fragment Offset = 0 (0x0) bytes
       IP: Time to Live = 128 (0x80)       IP: Protocol = UDP - User Datagram
       IP: Checksum = 0x2AA1       IP: Source Address = 194.149.104.197
       IP: Destination Address = 198.41.0.4
       IP: Data: Number of data bytes remaining = 40 (0x0028)twork Monitor
 + UDP: Src Port: Unknown, (1264); Dst Port: DNS (53); Length = 40 (0x28)
   DNS: 0x1A:Std Qry for www.lackfix.cz. of type Host Addr on class INET addr.
       DNS: Query Identifier = 26 (0x1A)
       DNS: DNS Flags = Query, OpCode - Std Qry, RD Bits Set, RCode - No error
           DNS: 0.............. = Query
           DNS: .0000.......... = Standard Query
           DNS: .....0..........  = Server not authority for domain
           DNS: ......0.........  = Message complete
           DNS: .......1........  = Recursive query desired
           DNS: ........0.......  = No recursive queries
           DNS: .........000....  = Reserved
           DNS: ............0000 = No error
       DNS: Question Entry Count = 1 (0x1)
       DNS: Answer Entry Count = 0 (0x0)
       DNS: Name Server Count = 0 (0x0)
       DNS: Additional Records Count = 0 (0x0)
```

```
      DNS: Question Section: www.lackfix.cz. of type Host Addr on class INET addr.
          DNS: Question Name: www.lackfix.cz.

          DNS: Question Type = Host Address
          DNS: Question Class = Internet address class
```

DNS odpověď:

```
+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
  IP: ID = 0x8DE8; Proto = UDP; Len: 320
      IP: Version = 4 (0x4)
      IP: Header Length = 20 (0x14)
    + IP: Service Type = 0 (0x0)
      IP: Total Length = 320 (0x140)
      IP: Identification = 36328 (0x8DE8)
    + IP: Flags Summary = 2 (0x2)
      IP: Fragment Offset = 0 (0x0) bytes
      IP: Time to Live = 245 (0xF5)
      IP: Protocol = UDP - User Datagram
      IP: Checksum = 0x053C
      IP: Source Address = 198.41.0.4
      IP: Destination Address = 194.149.104.197
      IP: Data: Number of data bytes remaining = 300 (0x012C)
+ UDP: Src Port: DNS, (53); Dst Port: Unknown (1264); Length = 300 (0x12C)
  DNS: 0x1A:Std Qry Resp. Auth. NS is CZ. of type Auth. NS on class INET addr.
      DNS: Query Identifier = 26 (0x1A)
      DNS: DNS Flags = Response, OpCode - Std Qry, RD Bits Set, RCode - No error
          DNS: 1............... = Response
          DNS: .0000........... = Standard Query
          DNS: .....0.......... = Server not authority for domain
          DNS: ......0......... = Message complete
          DNS: .......1........ = Recursive query desired
          DNS: ........0....... = No recursive queries
          DNS: .........000.... = Reserved
          DNS: ............0000 = No error
      DNS: Question Entry Count = 1 (0x1)
      DNS: Answer Entry Count = 0 (0x0)
      DNS: Name Server Count = 6 (0x6)
      DNS: Additional Records Count = 7 (0x7)
          DNS: Question Section: www.lackfix.cz. of type Host Addr on class INET addr.
          DNS: Question Name: www.lackfix.cz.
          DNS: Question Type = Host Address
          DNS: Question Class = Internet address class
      DNS: Authority Section: CZ. of type Auth. NS on class INET addr.(6 records present)
        + DNS: Resource Record: CZ. of type Auth. NS on class INET addr.
        + DNS: Resource Record: CZ. of type Auth. NS on class INET addr.
        + DNS: Resource Record: CZ. of type Auth. NS on class INET addr.
        + DNS: Resource Record: CZ. of type Auth. NS on class INET addr.
        + DNS: Resource Record: CZ. of type Auth. NS on class INET addr.
        + DNS: Resource Record: CZ. of type Auth. NS on class INET addr.
```

```
DNS: Additional Records Section: NS.CESNET.CZ. of type Host Addr on class INET …
  + DNS: Resource Record: NS.CESNET.CZ. of type Host Addr on class INET addr.
  + DNS: Resource Record: NS.EU.NET. of type Host Addr on class INET addr.
  + DNS: Resource Record: SUNIC.SUNET.SE. of type Host Addr on class INET addr.
  + DNS: Resource Record: SUNIC.SUNET.SE. of type Host Addr on class INET addr.
  + DNS: Resource Record: NS.UU.NET. of type Host Addr on class INET addr.
  + DNS: Resource Record: SPARKY.ARL.MIL. of type Host Addr on class INET addr.
  + DNS: Resource Record: NS.EUNET.CZ. of type Host Addr on class INET addr.
```

3. Příklad komunikace s DNS serverem smudla.svamberk.cz:

Pro srovnání s předchozím příkladem položíme stejný dotaz na jeden z běžných name serverů (nikoli root server).

Po serveru smudla.svamberk.cz požaduji rekurzivní překlad jména www.lackfix.cz. Dotaz jsem zadal pomocí programu nslookup. Pro ilustraci jsem v tomto příkladu navíc nastavil úroveň debug. Můžete porovnat výpis programu nslookup s obsahem DNS paketu.

V příkladu je podrobně rozvedeno i záhlaví IP paketu, odkud je zřejmá IP adresa dotazovaného serveru.
Výsledkem je získání konkrétní IP pro uzel www.lackfix.cz.

```
> server smudla.svamberk.cz
Default Server:  smudla.svamberk.cz
Address:  194.196.119.33

> set debug
> www.lackfix.cz
Server:  smudla.svamberk.cz
Address:  194.196.119.33

------------
Got answer:
    HEADER:
        opcode = QUERY, id = 4, rcode = NXDOMAIN
        header flags:  response, want recursion, recursion avail.
        questions = 1,  answers = 0,  authority records = 0,  additional = 0

    QUESTIONS:
        www.lackfix.cz.pvtnet.cz, type = A, class = IN

------------
------------
Got answer:
    HEADER:
        opcode = QUERY, id = 5, rcode = NOERROR
        header flags:  response, want recursion, recursion avail.
        questions = 1,  answers = 2,  authority records = 0,  additional = 0

    QUESTIONS:
        www.lackfix.cz, type = A, class = IN
    ANSWERS:
```

```
    ->  www.lackfix.cz
        canonical name = wwwvirt.pvtnet.cz
        ttl = 85847 (23 hours 50 mins 47 secs)
    ->  wwwvirt.pvtnet.cz
        internet address = 194.149.101.35
        ttl = 85847 (23 hours 50 mins 47 secs)

_____
Non-authoritative answer:
Name:    wwwvirt.pvtnet.cz
Address:  194.149.101.35
Aliases:  www.lackfix.cz

>
```

Všimněte si, že klient obdržel dvě odpovědi. První odpověď je záporná (rcode=NXDOMAIN). Důvod naleznete v sekci QUESTION. První dotaz byl totiž položen nikoli na jméno www.lackfix.cz, ale na jméno www.lackfix.cz.pvtnet.cz. To bylo způsobeno tím, že v příkazu nslookup nebyla uvedena tečka na konci DNS jména www.lackfix.cz, takže lokální resolver doplnil zprava v prvním dotazu DNS jméno doménou nastavenou v konfiguraci místního resolveru, což je právě pvtnet.cz.

DNS dotaz:

```
+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
  IP: ID = 0x7B8D; Proto = UDP; Len: 60
      IP: Version = 4 (0x4)
      IP: Header Length = 20 (0x14)
    + IP: Service Type = 0 (0x0)
      IP: Total Length = 60 (0x3C)
      IP: Identification = 31629 (0x7B8D)
    + IP: Flags Summary = 0 (0x0)
      IP: Fragment Offset = 0 (0x0) bytes
      IP: Time to Live = 128 (0x80)
      IP: Protocol = UDP - User Datagram
      IP: Checksum = 0x59E3
      IP: Source Address = 194.149.104.197
      IP: Destination Address = 194.196.119.33
      IP: Data: Number of data bytes remaining = 40 (0x0028)
+ UDP: Src Port: Unknown, (1270); Dst Port: DNS (53); Length = 40 (0x28)
  DNS: 0x5:Std Qry for www.lackfix.cz. of type Host Addr on class INET addr.
      DNS: Query Identifier = 5 (0x5)
      DNS: DNS Flags = Query, OpCode - Std Qry, RD Bits Set, RCode - No error
          DNS: 0.............. = Query
          DNS: .0000.......... = Standard Query
          DNS: .....0......... = Server not authority for domain
          DNS: ......0........ = Message complete
          DNS: .......1....... = Recursive query desired
          DNS: ........0...... = No recursive queries
          DNS: .........000.... = Reserved
          DNS: ............0000 = No error
      DNS: Question Entry Count = 1 (0x1)
      DNS: Answer Entry Count = 0 (0x0)
```

```
      DNS: Name Server Count = 0 (0x0)
      DNS: Additional Records Count = 0 (0x0)

      DNS: Question Section: www.lackfix.cz. of type Host Addr on class INET addr.
          DNS: Question Name: www.lackfix.cz.
          DNS: Question Type = Host Address
          DNS: Question Class = Internet address class
```

DNS odpověď (pouze druhá odpověď):

```
+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
  IP: ID = 0xA90D; Proto = UDP; Len: 105
      IP: Version = 4 (0x4)
      IP: Header Length = 20 (0x14)
    + IP: Service Type = 0 (0x0)
      IP: Total Length = 105 (0x69)
      IP: Identification = 43277 (0xA90D)
    + IP: Flags Summary = 0 (0x0)
      IP: Fragment Offset = 0 (0x0) bytes
      IP: Time to Live = 122 (0x7A)
      IP: Protocol = UDP - User Datagram
      IP: Checksum = 0x3236
      IP: Source Address = 194.196.119.33
      IP: Destination Address = 194.149.104.197
      IP: Data: Number of data bytes remaining = 85 (0x0055)
+ UDP: Src Port: DNS, (53); Dst Port: Unknown (1270); Length = 85 (0x55)
  DNS: 0x5:Std Qry Resp. for www.lackfix.cz. of type Canonical name on class INET addr.
      DNS: Query Identifier = 5 (0x5)
      DNS: DNS Flags = Response, OpCode - Std Qry, RD RA Bits Set, RCode - No error
          DNS: 1............... = Response
          DNS: .0000........... = Standard Query
          DNS: .....0.......... = Server not authority for domain
          DNS: ......0......... = Message complete
          DNS: .......1........ = Recursive query desired
          DNS: ........1....... = Recursive queries supported by server
          DNS: .........000.... = Reserved
          DNS: ............0000 = No error
      DNS: Question Entry Count = 1 (0x1)
      DNS: Answer Entry Count = 2 (0x2)
      DNS: Name Server Count = 0 (0x0)
      DNS: Additional Records Count = 0 (0x0)
      DNS: Question Section: www.lackfix.cz. of type Host Addr on class INET addr.
          DNS: Question Name: www.lackfix.cz.
          DNS: Question Type = Host Address
          DNS: Question Class = Internet address class
      DNS: Answer section: www.lackfix.cz. of type Canonical name on class INET addr….
        + DNS: Resource Record: www.lackfix.cz. of type Canonical name on class INET addr.
        + DNS: Resource Record: wwwvirt.pvtnet.cz. of type Host Addr on class INET addr.
```

4. Příklad použití protokolu TCP:

Pomocí programu nslookup se snažím získat všechny RR věty vztahující se ke jménu aaa.pvtnet.cz.

```
# nslookup
Default Server:  localhost
Address:  127.0.0.1

> set q=any
> aaa.pvtnet.cz
Server:  localhost
Address:  127.0.0.1

aaa.pvtnet.cz   text = „Lokalita Budejovice“
aaa.pvtnet.cz   text = „postovni server“
aaa.pvtnet.cz   text = „32 MB operacni pameti“
aaa.pvtnet.cz   text = „brzy bude proveden upgrade na 64 MB“
aaa.pvtnet.cz   CPU = PC        OS = Linux 1.3.20
aaa.pvtnet.cz   text = „e-mail: alena@pvt.net“
aaa.pvtnet.cz   text = „zkusebni uzel“
aaa.pvtnet.cz   text = „posta pro aaa.pvtnet.cz“
aaa.pvtnet.cz   text = „zatim nefunguje“
aaa.pvtnet.cz   preference = 10, mail exchanger = info.pvt.net
aaa.pvtnet.cz   preference = 20, mail exchanger = cbu.pvtnet.cz
aaa.pvtnet.cz   preference = 100, mail exchanger = mail.pvtnet.cz
aaa.pvtnet.cz   preference = 200, mail exchanger = mail2.pvtnet.cz
aaa.pvtnet.cz   internet address = 195.47.55.55
pvtnet.cz       nameserver = ns.pvt.net
pvtnet.cz       nameserver = ns1.pvt.net
pvtnet.cz       nameserver = snmp0.pvt.net
pvtnet.cz       nameserver = ns0.pipex.net
pvtnet.cz       nameserver = ns1.pipex.net
info.pvt.net    internet address = 194.149.104.203
cbu.pvtnet.cz   internet address = 194.149.105.18
ns.pvt.net      internet address = 194.149.105.18
ns1.pvt.net     internet address = 194.149.103.201
snmp0.pvt.net   internet address = 194.149.103.34
ns0.pipex.net   internet address = 158.43.128.8
ns1.pipex.net   internet address = 158.43.192.7
>
```

DNS dotaz poslaný přenosovým protokolem UDP:

```
+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
+ IP: ID = 0x5BA9; Proto = UDP; Len: 59
+ UDP: Src Port: Unknown, (1284); Dst Port: DNS (53); Length = 39 (0x27)
  DNS: 0xC:Std Qry for aaa.pvtnet.cz. of type Req. for all on class INET addr.
      DNS: Query Identifier = 12 (0xC)
      DNS: DNS Flags = Query, OpCode - Std Qry, RD Bits Set, RCode - No error
          DNS: 0.............. = Query
          DNS: .0000.......... = Standard Query
```

```
          DNS: .....0..... = Server not authority for domain
          DNS: ......0..... = Message complete
          DNS: .......1..... = Recursive query desired
          DNS: ........0..... = No recursive queries
          DNS: .........000.... = Reserved
          DNS: ............0000 = No error
      DNS: Question Entry Count = 1 (0x1)
      DNS: Answer Entry Count = 0 (0x0)
      DNS: Name Server Count = 0 (0x0)
      DNS: Additional Records Count = 0 (0x0)
    + DNS: Question Section: aaa.pvtnet.cz. of type Req. for all on class INET addr.
```

DNS odpověď: kompletní odpověď je delší než 512 bajtů, resolver tedy obdržel UDP protokolem pouze zkrácenou odpověď s indikací zkrácení v bitu TC (truncated).

```
+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
+ IP: ID = 0x6970; Proto = UDP; Len: 524
+ UDP: Src Port: DNS, (53); Dst Port: Unknown (1284); Length = 504 (0x1F8)
  DNS: 0xC:Std Qry Resp. for aaa.pvtnet.cz. of type Host Addr on class INET addr.
      DNS: Query Identifier = 12 (0xC)
      DNS: DNS Flags = Response, OpCode - Std Qry, AA TC RD RA Bits Set, RCode - No error
          DNS: 1............... = Response
          DNS: .0000........... = Standard Query
          DNS: .....1..... = Server authority for domain
          DNS: ......1..... = Message truncated
          DNS: .......1..... = Recursive query desired
          DNS: ........1..... = Recursive queries supported by server
          DNS: .........000.... = Reserved
          DNS: ............0000 = No error
      DNS: Question Entry Count = 1 (0x1)
      DNS: Answer Entry Count = 14 (0xE)
      DNS: Name Server Count = 5 (0x5)
      DNS: Additional Records Count = 0 (0x0)
    + DNS: Question Section: aaa.pvtnet.cz. of type Req. for all on class INET addr.
    + DNS: Answer section: aaa.pvtnet.cz. of type Host Addr on class INET addr.(14 records
present)
    + DNS: Authority Section = N/A
```

Následuje DNS dotaz v protokolu TCP.

```
+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
+ IP: ID = 0x5FA9; Proto = TCP; Len: 71
+ TCP: .AP..., len:   31, seq:  31853005-31853035, ack: 320256001, win: 8760, src: 1285  dst:
53
  DNS: 0x100:Std Qry for ¤ö_ of type Unknown Type on class Unknown Class
      DNS: TCP Length = 12 (0xC)
      DNS: Query Identifier = 256 (0x100)
      DNS: DNS Flags = Query, OpCode - Std Qry, RCode - Server unable to interpret query
```

```
          DNS: 0.............. = Query
          DNS: .0000.......... = Standard Query
          DNS: .....0......... = Server not authority for domain
          DNS: ......0........ = Message complete
          DNS: .......0........ = Iterative query desired
          DNS: ........0....... = No recursive queries
          DNS: .........000.... = Reserved
          DNS: ............0001 = Server unable to interpret query
      DNS: Question Entry Count = 0 (0x0)
      DNS: Answer Entry Count = 0 (0x0)
      DNS: Name Server Count = 0 (0x0)
      DNS: Additional Records Count = 865 (0x361)
    + DNS: Additional Records Section:  of type Unknown Type on class Unknown Class(865
records present)
```

DNS odpověď doručená protokolem TCP v plné délce 650 B.

```
+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
+ IP: ID = 0x697C; Proto = TCP; Len: 692
+ TCP: .AP..., len:  652, seq: 320256001-320256652, ack:  31853036, win:33580, src:   53  dst:
1285
  DNS: 0xC:Std Qry Resp. for aaa.pvtnet.cz. of type Host Addr on class INET addr.
      DNS: TCP Length = 650 (0x28A)
      DNS: Query Identifier = 12 (0xC)
      DNS: DNS Flags = Response, OpCode - Std Qry, AA RD RA Bits Set, RCode - No error
          DNS: 1.............. = Response
          DNS: .0000.......... = Standard Query
          DNS: .....1......... = Server authority for domain
          DNS: ......0........ = Message complete
          DNS: .......1........ = Recursive query desired
          DNS: ........1....... = Recursive queries supported by server
          DNS: .........000.... = Reserved
          DNS: ............0000 = No error
      DNS: Question Entry Count = 1 (0x1)
      DNS: Answer Entry Count = 14 (0xE)
      DNS: Name Server Count = 5 (0x5)
      DNS: Additional Records Count = 7 (0x7)
    + DNS: Question Section: aaa.pvtnet.cz. of type Req. for all on class INET addr.
    + DNS: Answer section: ._aaa_pv. of type Unknown Type on class Unknown Class(14 records
present)
    + DNS: Authority Section = N/A
    + DNS: Additional Records Section = N/A
```

5. Příklad použití programu nslookup ke zjištění obsahu komunikace.

Správce DNS serveru obvykle nepoužívá ke sledování komunikace mezi klientem a serverem Microsoft network monitor, ale vystačí si s programem nslookup. Ladicí úrovně *debug* a *d2* vypisují obsah DNS paketu ve srozumitelné podobě.

Porovnejte výpis získaný programem nslookup po nastavení ladicí úrovně debug a ladicí úrovně d2. V obou případech byl položen dotaz na stejný RR record.

Program nslookup nastaven na ladicí úroveň debug:

```
> set debug
>www.lackfix.cz.
Server:  localhost
Address:  127.0.0.1


------
Got answer (263 bytes):
    HEADER:
        opcode = QUERY, id = 4, rcode = NOERROR
        header flags:  response, auth. answer, want recursion, recursion avail.
        questions = 1,  answers = 2,  authority records = 5,  additional = 5

    QUESTIONS:
        www.lackfix.cz, type = A, class = IN
    ANSWERS:
    ->  www.lackfix.cz
        type = CNAME, class = IN, dlen = 19
        canonical name = wwwvirt.pvtnet.cz
        ttl = 86400 (1 day)
    ->  wwwvirt.pvtnet.cz
        type = A, class = IN, dlen = 4
        internet address = 194.149.101.35
        ttl = 86400 (1 day)
    AUTHORITY RECORDS:
    ->  pvtnet.cz
        type = NS, class = IN, dlen = 12
        nameserver = ns.pvt.net
        ttl = 86400 (1 day)
    ->  pvtnet.cz
        type = NS, class = IN, dlen = 6
        nameserver = ns1.pvt.net
        ttl = 86400 (1 day)
    ->  pvtnet.cz
        type = NS, class = IN, dlen = 8
        nameserver = snmp0.pvt.net
        ttl = 86400 (1 day)
    ->  pvtnet.cz
        type = NS, class = IN, dlen = 12
        nameserver = ns0.pipex.net
        ttl = 86400 (1 day)
    ->  pvtnet.cz
        type = NS, class = IN, dlen = 6
        nameserver = ns1.pipex.net
        ttl = 86400 (1 day)
    ADDITIONAL RECORDS:
    ->  ns.pvt.net
```

```
        type = A, class = IN, dlen = 4
        internet address = 194.149.105.18
        ttl = 86400 (1 day)
    ->  ns1.pvt.net
        type = A, class = IN, dlen = 4
        internet address = 194.149.103.201
        ttl = 86400 (1 day)
    ->  snmp0.pvt.net
        type = A, class = IN, dlen = 4
        internet address = 194.149.103.34
        ttl = 86400 (1 day)
    ->  ns0.pipex.net
        type = A, class = IN, dlen = 4
        internet address = 158.43.128.8
        ttl = 62655 (17 hours 24 mins 15 secs)
    ->  ns1.pipex.net
        type = A, class = IN, dlen = 4
        internet address = 158.43.192.7
        ttl = 62655 (17 hours 24 mins 15 secs)

------
Name:    wwwvirt.pvtnet.cz
Address:  194.149.101.35
Aliases:  www.lackfix.cz
```

Program nslookup nastaven na ladicí úroveň d2:

```
#nslookup
> set d2
> www.lackfix.cz.
Server:  localhost
Address:  127.0.0.1

------
SendRequest(), len 32
    HEADER:
        opcode = QUERY, id = 3, rcode = NOERROR
        header flags:  query, want recursion
        questions = 1,  answers = 0,  authority records = 0,  additional = 0

    QUESTIONS:
        www.lackfix.cz, type = A, class = IN

------
------
Got answer (263 bytes):
    HEADER:
        opcode = QUERY, id = 3, rcode = NOERROR
        header flags:  response, auth. answer, want recursion, recursion avail.
        questions = 1,  answers = 2,  authority records = 5,  additional = 5
```

```
QUESTIONS:
    www.lackfix.cz, type = A, class = IN
ANSWERS:
->  www.lackfix.cz
    type = CNAME, class = IN, dlen = 19
    canonical name = wwwvirt.pvtnet.cz
    ttl = 86400 (1 day)
->  wwwvirt.pvtnet.cz
    type = A, class = IN, dlen = 4
    internet address = 194.149.101.35
    ttl = 86400 (1 day)
AUTHORITY RECORDS:
->  pvtnet.cz
    type = NS, class = IN, dlen = 12
    nameserver = ns.pvt.net
    ttl = 86400 (1 day)
->  pvtnet.cz
    type = NS, class = IN, dlen = 6
    nameserver = ns1.pvt.net
    ttl = 86400 (1 day)
->  pvtnet.cz
    type = NS, class = IN, dlen = 8
    nameserver = snmp0.pvt.net
    ttl = 86400 (1 day)
->  pvtnet.cz
    type = NS, class = IN, dlen = 12
    nameserver = ns0.pipex.net
    ttl = 86400 (1 day)
->  pvtnet.cz
    type = NS, class = IN, dlen = 6
    nameserver = ns1.pipex.net
    ttl = 86400 (1 day)
ADDITIONAL RECORDS:
->  ns.pvt.net
    type = A, class = IN, dlen = 4
    internet address = 194.149.105.18
    ttl = 86400 (1 day)
->  ns1.pvt.net
    type = A, class = IN, dlen = 4
    internet address = 194.149.103.201
    ttl = 86400 (1 day)
->  snmp0.pvt.net
    type = A, class = IN, dlen = 4
    internet address = 194.149.103.34
    ttl = 86400 (1 day)
->  ns0.pipex.net
    type = A, class = IN, dlen = 4
    internet address = 158.43.128.8
    ttl = 62851 (17 hours 27 mins 31 secs)
-> ns1.pipex.net
    type = A, class = IN, dlen = 4
    internet address = 158.43.192.7
```

```
        ttl = 62851 (17 hours 27 mins 31 secs)

_______
Name:    wwwvirt.pvtnet.cz
Address:  194.149.101.35


Aliases:  www.lackfix.cz

>
```

11.15 DNS UPDATE

DNS UPDATE je specifikován v RFC 2136.

DNS UPDATE umožnuje dynamicky opravovat záznamy v DNS databázi. Umožnuje přidat jednu nebo několik vět do zónového souboru, nebo jednu případně několik vět ze zónového souboru zrušit. Záznamy v DNS databázi tedy nemusí již staticky opravovat správce serveru, ale je možné záznamy v DNS databázi opravit dynamicky pomocí příslušného klienta na dálku použitím protokolu DNS.

S DNS UPDATE se setkáváme v BIND 8, proto budeme používat i terminologii BIND 8, tj. master/slave name server.

Data v zóně lze pomocí DNS update opravovat pouze na primary master serveru. Pokud DNS Update dotaz obdrží slave server, pak tento dotaz forwarduje primary master serveru.

DNS UPDATE neumožňuje vytváření nových zón, umožňuje pouze opravovat zóny již existující. Pomocí DNS Update tedy není možné přidávat novou SOA větu nebo SOA větu rušit, SOA větu je možné pouze opravit.

Jedním DNS Update dotazem je možné opravit jednu nebo několik vět v jedné zóně.

Opravu zóny pomocí DNS Update je možné provést za specifikovaných podmínek. Podmínkou je existence nebo neexistence daných RR vět v master zóně před opravou. Např. požaduji-li zrušení věty v zóně, musí tato věta v zóně před opravou existovat. Podmínek tohoto typu může být pro provedení opravy specifikováno několik. Z hlediska splnění jsou podmínky posuzovány jako celek. Tedy není-li splněna jedna z uvedených podmínek, nejsou splněny podmínky jako celek a žádná z požadovaných oprav se neprovede.

V DNS update paketu jsou odděleně specifikovány podmínky pro provedení opravy a odděleně jsou uvedeny RR věty, které se mají do zónového souboru přidat nebo z něho zrušit.

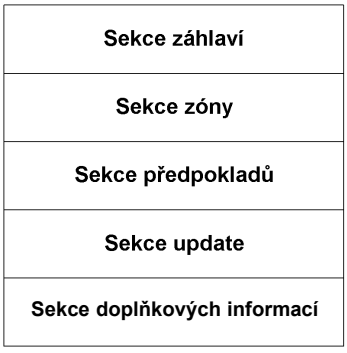
DNS UPDATE využívá specifikaci DNS protokolu tak, jak jej definuje RFC 1035 a jak jsme si jej vysvětlili v kapitole 11.14. Norma RFC2136 však definuje některá rozšíření tohoto protokolu, např. nový typ zprávy, nové výsledkové kódy. Formát DNS paketu pro UPDATE tedy zůstává stejný, skládá se opět z pěti částí. Jednotlivé části však mají v tomto případě specifický obsah i pojmenování.

Paket DNS UPDATE se skládá ze sekcí (viz obr. 11.15):

- **záhlaví**
- **sekce zóny** – definuje zónu, ke které se vztahují opravy
- **sekce předpokladu** – sada RR vět, které musí v zóně existovat
- **sekce update** – sada RR vět, které se mají opravit nebo zrušit

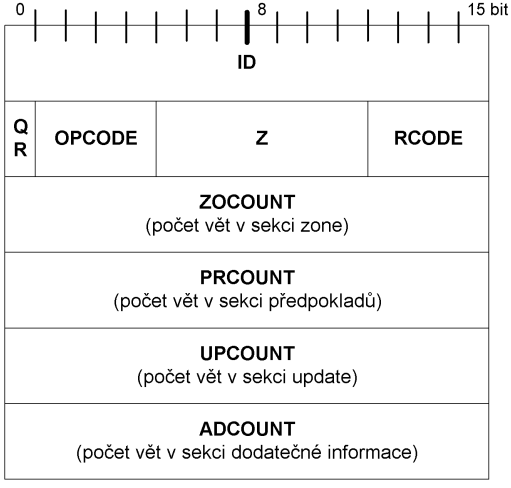
- **sekce doplňkových informací** – informace jež nejsou součástí update, ale jsou třeba k provedení update

Obr. 11.15
Formát paketu DNS UPDATE



11.15.1 Sekce záhlaví

Obr. 11.16
Sekce záhlaví paketu DNS UPDATE



Sekce záhlaví podobně jako u DNS QUERY obsahuje v prvních dvou bajtech identifikaci (pole ID), dále následují dva bajty řídících polí a délky jednotlivých sekcí (každá délka je 2 B):

ZOCOUNT – POČET VĚT V SEKCI ZONE
PRCOUNT – POČET VĚT V SEKCI PŘEDPOKLADU
UPCOUNT – POČET VĚT V SEKCI UPDATE
ADCOUNT – POČET VĚT V SEKCI DODATEČNÉ INFORMACE

Pole	Počet bitů	Hodnota
ID	16	Identifikátor zprávy, je kopírován do odpovědi.
QR	1	0 pokud je zpráva dotazem, 1 pokud je zpráva odpovědí. Ostatní kontrolní bity DNS dotazu Update nepoužívá. Za QR bezrostředně následuje pole Opcode.
Opcode	4	5(UPDATE), kopíruje se z dotazu do odpovědi
Z	7	Rezerva, musí být naplněna binárními nulami
Rcode	4	V odpovědi výsledkový kód, v dotazu nedefinované.

Tab. 11.9 Význam jednotlivých řídících polí

Kód chyby	číselná hodnota	popis chyby
NOERROR	0	Bez chyby
FORMERR	1	Chybný formát zprávy, name server ji neumí interpretovat
SERVFAIL	2	Při zpracování zprávy došlo k interní chybě serveru (např. chybě OS nebo vypršel timeout při forwardingu)
NXDOMAIN	3	Jméno, které by mělo existovat, neexistuje
NOTIMP	4	Name server nepodporuje daný typ zprávy (Opcode)
REFUSED	5	Name server odmítá provést zprávu např. z bezpečnostních důvodů
YXDOMAIN	6	Jméno, které by nemělo existovat, existuje.
YXRRSET	7	Sada RR vět, která by neměla existovat, existuje.
NXRRSET	8	Sada RR vět, která by měla existovat, neexistuje.
NOAUTH	9	Server není autoritou pro danou zónu.
NOTZONE	10	Jméno uvedené v sekci předpokladů nebo v sekci update není v dané zóně.

Tab. 11.10 Výsledkové kódy odpovědí (pole Rcode)

11.15.2 Sekce zóny

Sekce zóny určuje zónu, kterou bude opravována. Jedním DNS UPDATE dotazem je možné opravovat pouze jednu zónu, tj. sekce zone povoluje pouze jednu větu.

Sekce je tvořena třemi částmi:

ZNAME – jméno zóny
ZTYPE – musí být řetězec SOA
ZCLASS – třída zóny – IN (Internet)

11.15.3 Sekce předpokladu

Sekce předpokladu obsahuje skupinu RR vět, které musí v dané zóně existovat na primary master serveru v okamžiku doručení UPDATE paketu. V sekci předpokladu je možné použít pět variant:

1. **V zóně musí existovat minimálně jedna RR věta s daným NAME a TYPE** (*RR set exists, value independent*).

   Sekce předpokladu obsahuje jednu větu s daným NAME a TYPE, kterou očekává v zóně. Ostatní položky jsou nevýznamné, proto se použije RDLENGTH=0, RDATA je prázdné, CLASS=ANY, TTL=0.

   Např. požaduji, aby v doméně existovala věta typu A s doménovým jménem aaa.firma.cz s libovolným RDATA.

2. **V zóně musí existovat sada vět RR daného NAME a TYPE** a jejich pravá strana se musí shodovat s pravou stranou vět v UPDATE paketu (*RR set exists, value dependent*). Na pořadí vět nezáleží.

   V sekci je skupina RR vět s daným NAME a TYPE a RDATA, TTL=0, CLASS je určená v sekci zone.

   Např. požaduji, aby v zóně existovaly věty:

   ```
   mail.firma.cz.    A       195.47.11.11
   www.firma.cz.     A       195.47.11.12
   firma.cz. MX    10        mail.firma.cz.
   ```

3. **V zóně neexistuje žádná RR věta s daným NAME a TYPE** (*RR set does not exist*). Sekce obsahuje jednu RR větu s daným NAME a TYPE, RDLEGTH=0, RDATA je prázdné, CLASS=NONE, TTL=0.

   Např. požaduji, aby v zóně neexistovala žádná věta typu A s doménovým jménem mail.firma.cz.

4. **V zóně musí existovat alespoň jedna věta s daným NAME a typem definovaným v sekci ZONE** (*Name is in use*). V sekci je jedna věta RR s daným NAME, RDLENGTH=0, RDATA je prázdné, CLASS=ANY, TYPE=ANY, TTL=0.

   Např. požaduji, aby v zóně existovala alespoň jedna věta, jejíž doménové jméno obsahuje řetězec firma.cz. Tj. nejde o prázdnou zónu.

5. **V zóně neexistuje žádná RR věta jakéhokoli typu s daným NAME** (*Name is not in use*). Sekce obsahuje jednu RR větu s daným NAME, RDLENGTH=0, RDATA je prázdné, CLASS=NONE, TYPE=ANY, TTL=0.

   Např. požaduji, aby v zóně neexistovala žádná věta, jejíž doménové jméno obsahuje řetězec firma.cz. Tj. jde o prázdnou zónu.

11.15.4 Sekce update

Sekce update obsahuje RR věty, které mají být do zóny přidány nebo z ní zrušeny. Je možné provést čtyři druhy změn:

1. **Přidat věty RR**

   Sekce update obsahuje několik vět. Do souboru se přidají věty s daným NAME, TYPE, TTL, RDLENGTH a RDATA. CLASS se převezme ze sekce zone. Duplicitní věty v tomto seznamu se ignorují.

   Např. požaduji přidat věty:

   ```
   firma.cz. MX      20      mh.firma.cz.
   mh.firma.cz.      A       195.47.13.12
   ```

2. **Zrušit sadu RR vět daného typu.**

   Sekce obsahuje jednu větu, uvedené NAME a TYPE určuje, které věty mají být zrušeny. TTL=0, CLASS=ANY, RDLENGTH=0, RDATA je prázdné.

   Např. požaduji zrušit všechny věty s doménovým jménem mail.firma.cz.

3. **Zrušit všechny RR věty daného jména.**

   Sekce obsahuje jednu RR větu, dané NAME určuje, které věty mají být zrušeny. TYPE=ANY, TTL=0, CLASS=ANY, RDLENGTH=0, RDATA je prázdné.

   Např. požaduji zrušit všechny věty typu MX s doménovým jménem firma.cz.

4. **Zrušit jednu RR větu**

   Sekce obsahuje větu, která má být zrušena (NAME, TYPE, RDLENGTH, RDATA). TTL=0, CLASS=NONE. *Např. požaduji zrušit větu:*

```
firma.cz. IN      MX  10   mail.firma.cz.
```

11.15.5 Sekce doplňujících informací

Sekce doplňujících informací obsahuje RR věty, které se vztahují k vlastnímu update nebo k novým větám přidaným pomocí update. Např. může zde být uvedena glue věta pro zónu, pokud se přidává nová NS věta pro zónu.

11.15.6 Poznámky na závěr

Server si musí nová data získaná pomocí DNS update uložit do databáze na disk. Teprve potom může poslat odpověď na DNS update. Záleží na serveru, zda uloží pouze data získaná pomocí update, nebo vytvoří nový soubor pro celou zónu.

Je doporučováno používat DNS update spolu s nějakým systémem zabezpečení. Jednou z možností je Secure dynamic update specifikovaný v RFC 2137. Bez použití Secure DNS Update je vhodné alespoň zabezpečit, že bude server akceptovat pouze Update dotazy z dané IP adresy. Tato IP se určí v konfiguraci serveru.

11.16 DNS notify

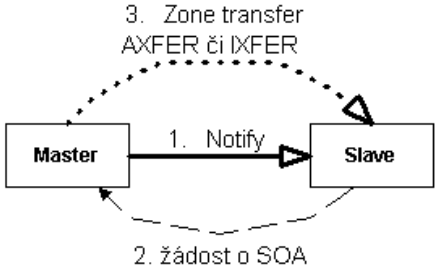
Mechanismus DNS notify popisuje RFC1996.

Mechanismus DNS notify umožňuje informovat slave servery o změně dat v zóně. Při použití DNS notify může slave server získat aktuální data k zóně dříve než teprve v okamžiku vypršení intervalu v poli *refresh* uvedeném ve větě typu SOA zóny.

Komunikaci o zóně mezi master a slave serverem při použití DNS notify iniciuje master server. Master server posílá v případě změny zóny zprávu slave serverům „Požádej mě o *zone transfer*“. Slave server po obdržení zprávy notify může okamžitě požádat o zone transfer.

Zprávu o změně zóny (DNS notify) obdrží všechny servery, které jsou uvedeny v NS větách pro zónu. Není informován server uvedený ve větě SOA, protože se předpokládá, že právě tento server zprávy generuje. Některé implementace umožňují správci master serveru sadu name serverů doplnit o další IP adresy dalších name serverů. Tyto další servery se nazývají stealth servery. Množina serverů, pro které se DNS notify generuje, se nazývá Notify set.

Obr. 11.17
Notify



11.16.1 Zpráva Notify

Zpráva notify používá formát DNS paketu definovaný v RFC 1035. DNS Notify používá pouze podmnožinu polí v paketu, nepoužitá pole musí být vyplněna binárními nulami. Typ zprávy (Opcode) je nastaven na hodnotu 4 (NOTIFY). Master server může jako součást notify zprávy poslat NAME, CLASS, TYPE i RDATA změněných vět v zóně. Notify zprávy nevyužívají sekci autoritativních serverů ani sekci doplňkových informací.

Příklad DNS notify: byla opravena master zóna abcde.cz:

```
+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
+ IP: ID = 0xD4; Proto = UDP; Len: 54
+ UDP: Src Port: Unknown, (1049); Dst Port: DNS (53); Length = 34 (0x22)
  DNS: 0x54C6:Std Qry for abcde.cz. of type SOA on class INET addr.
      DNS: Query Identifier = 21702 (0x54C6)
      DNS: DNS Flags = Query, OpCode - Rsrvd, RCode - No error
          DNS: 0............... = Query
          DNS: .0100........... = Reserved
          DNS: .....0.......... = Server not authority for domain
          DNS: ......0......... = Message complete
          DNS: .......0........ = Iterative query desired
          DNS: ........0....... = No recursive queries
          DNS: .........000.... = Reserved
          DNS: ............0000 = No error
      DNS: Question Entry Count = 1 (0x1)
      DNS: Answer Entry Count = 0 (0x0)
      DNS: Name Server Count = 0 (0x0)
      DNS: Additional Records Count = 0 (0x0)
      DNS: Question Section: abcde.cz. of type SOA on class INET addr.
          DNS: Question Name: abcde.cz.
          DNS: Question Type = Start of zone of authority
          DNS: Question Class = Internet address class

00000:  00 60 3E 1D 90 00 00 20 AF F9 2F A0 08 00 45 00   .`.... ../...E.
00010:  00 36 00 D4 00 00 40 11 22 E1 C2 95 68 C5 C2 95   .6....@.“...h...
00020:  69 12 04 19 00 35 00 22 E6 FD 54 C6 20 00 00 01   i....5.“..T. ...
00030:  00 00 00 00 00 00 05 61 62 63 64 65 02 63 7A 00   .......abcde.cz.
00040:  00 06 00 01                                       ....
```

V DNS paketu je pole Opcode nastaveno na hodnotu 4. Software použitý pro odchycení paketu na síti – Microsoft Network Monitor V4 interpretuje však tuto hodnotu jako Rsrvd (Reserved), neboť tato verze MNM ještě nepodporuje DNS notify zprávy.

Příklad DNS notify odpovědi:

```
+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
+ IP: ID = 0x84C9; Proto = UDP; Len: 40
+ UDP: Src Port: DNS, (53); Dst Port: Unknown (1049); Length = 20 (0x14)
  DNS: 0x54C6:Std Qry Resp. : This query not supported by name server
       DNS: Query Identifier = 21702 (0x54C6)
       DNS: DNS Flags = Response, OpCode - Rsrvd, RA Bits Set,
            RCode - This query not supported by name server
           DNS: 1............... = Response
           DNS: .0100........... = Reserved
           DNS: .....0.......... = Server not authority for domain
           DNS: ......0......... = Message complete
           DNS: .......0........ = Iterative query desired
           DNS: ........1....... = Recursive queries supported by server
           DNS: .........000.... = Reserved
           DNS: ............0100 = This query not supported by name server
       DNS: Question Entry Count = 0 (0x0)
       DNS: Answer Entry Count = 0 (0x0)
       DNS: Name Server Count = 0 (0x0)
       DNS: Additional Records Count = 0 (0x0)
       DNS: Frame Padding

00000:  00 20 AF F9 2F A0 00 60 3E 1D 90 00 08 00 45 00   . ../..`.....E.
00010:  00 28 84 C9 00 00 1D 11 C1 F9 C2 95 69 12 C2 95   .(..........i...
00020:  68 C5 00 35 04 19 00 14 AF 2A 54 C6 A0 84 00 00   h..5.....*T.....
00030:  00 00 00 00 00 00 00 00 00 00 00 00               ............
```

K přenosu Notify paketu se používá přenosový protokol UDP nebo TCP. Pokud je použit protokol TCP, je zpráva notify vyslána pouze jednou. Na master serveru je nastaven časový interval, po který master server čeká na odpověď. Při použití TCP slave server ani master server nesmí přerušit poskytování služeb během transakce.

Pokud je použit protokol UDP, master server periodicky posílá notify zprávy na slave server. Vysílání notify zpráv master server zastaví po obdržení odpovědi. Pokud master server neobdrží odpověď, pak vysílání těchto zpráv zastaví po vyčerpání nastaveného počtu opakování zpráv nebo poté co ICMP oznámí, že port je nedostupný. Interval mezi vysíláním jednotlivých zpráv může být specifikován jako parametr v konfiguraci master serveru (zpravidla 60 s). Podobně může být nastaven i počet povolených opakování (zpravidla 5).

Jediná událost, která aktivuje vyslání notify zprávy je změna SOA věty. Po obdržení zprávy Notify by se měl slave server chovat stejně jako když zóně uvedené v QNAME vyprší interval uvedený v poli refresh věty SOA. Slave server by tedy měl požádat master server o SOA pro zónu a zkontrolovat pole serial number. Pokud došlo ke zvýšení sériového čísla, pak iniciovat AXFR nebo IXFR.

Ve zprávě *zone transfer* by se měl slave dělat z masteru, který poslal na slave server notify zprávu.

V notify otázce může master poslat i změněné RR věty (změněné jméno, třídu, typ a nepovinně i RDATA). V žádném případě však nemohou být tyto informace (změněné RR věty v answer sekci notify otázky) použity pro opravu dat na slave serveru nebo jako indikace toho, že je třeba provést zone transfer nebo změnit refresh time u zóny. Jedná se pouze o informaci, ze které může slave server např. zjistit, že má již aktuální data a nepotřebuje iniciovat zone transfer.

Notify odpověď neobsahuje žádné podstatné informace, rozhodující je pouze fakt, že master server notify odpověď obdrží.

Pokud slave obdrží zprávu Notify obsahující QNAME od uzlu, který není master pro zónu, měl by tuto zprávu ignorovat a generovat chybovou hlášku do logu.

Při startu by server měl poslat Notify pro každou autoritativní zónu. Při restartu serveru je poslání notify zprávy volitelné.

Každý slave server pravděpodobně obdrží více stejných zpráv Notify. Notify protokol musí tuto multiplicitu podporovat.

Master server se snaží vyhnout velkému množství současných zone trasferů. Může tedy zprávu notify posílat s určitým zpožděním. Toto zpoždění bude vybíráno náhodně, takže každý slave server začne svůj zone transfer v jiném čase. Toto zpoždění nesmí být větší než pole refresh. Zpoždění může být nastavitelné parametrem master zony (30-60 s). Slave server, který obdrží notify musí nejprve tuto iniciovanou transakci dokončit a teprve potom posílat další notify zprávy níže.

V Bind 8.1 je mechanismus DNS Notify standardně implementován.

11.17 Inkrementální zone transfer

Inkrementální zone transfer specifikuje RFC 1995.

Inkrementální zone transfer (IXFR) umožňuje přenášet při změně dat v zóně z master serveru na slave server pouze změněná data, tedy pouze část zóny. Naproti tomu klasický zone transfer (AXFR) přenáší při sebemenší změně zóny celou zónu.

K tomu aby mohl master server poskytovat slave serveru pouze změněné věty, musí udržovat jednotlivé stavy databáze po jednotlivých změnách. Master server tedy musí udržovat nejnovější verzi zóny a rozdíly mezi nejnovější verzí a několika staršími verzemi. Verze souborů se liší sériovým číslem v SOA větě. Pokud slave server zjistí, že potřebuje nová data k zóně a podporuje IXRF, pošle dotaz na master server, kde uvede jakou verzi zóny má jako poslední např.: 98052001 (*serial*). Master server jako odpověď pošle slave serveru pouze změněné věty, tj. věty, které mají být zrušeny a věty nové. Alternativně může server poslat v odpovědi celou zónu. Celou zónu posílá server i tehdy, když má klient tak starou větu SOA, že již server není schopen poslat IXFR.

Po opravě zóny v paměti musí slave server uložit změny do souboru, teprve potom může sám poskytovat odpověď na IXFR dotaz. Pokud master server obdrží dotaz s vyšším číslem SOA, než má sám, pak jako odpověď vrací pouze svou aktuální SOA větu.

Pro přenos IXFR dotazů je možné použít TCP i UDP. Pokud klient zašle dotaz UDP, měl by server zaslat UDP odpověď. Pokud je odpověď delší než 512 B, pošle server v UDP odpovědi pouze větu SOA a klient musí navázat spojení TCP.

Slave server, který požaduje inkrementální zone transfer bývá označován jako IXFR klient. Master nebo slave server, který poskytuje inkrementální zone transfer bývá označován jako IXFR server.

IXFR používá formát DNS paketu tak, jak je definovaný v RFC 1035.

11.17.1 Formát dotazu

Jako typ otázky (Opcode) je uvedeno IXFR a sekce autoritativních name serverů obsahuje SOA větu zóny uložené na slave serveru.

11.17.2 Formát odpovědi

Jako typ otázky (Opcode) je v odpovědi opět uvedeno IXFR. První a poslední RR v sekci odpovědi je SOA věta zóny, která se má aktualizovat.

V IXFR je možné poslat jako odpověď jednu nebo několik změn (poslední nebo několik posledních verzí zóny) v rámci jedné zóny. Seznam všech změn v rámci jedné verze je v sekci odpovědi uzavřen z obou stran větami SOA.

Za změnu se považuje přidání nebo zrušení RR. Zrušené věty jsou předcházeny starou SOA a přidané věty jsou předcházeny novou SOA RR. Oprava věty je chápána jako zrušení věty původní a přidání věty nové.

Změny jsou v sekci odpovědi uspořádány od nejstarší po nejnovější.

IXFR klient může změnit starou verzi souboru na novou pouze po úspěšném provedení všech obdržených změn.

Inkrementální odpověď se liší od neinkrementální odpovědi tím, že začíná dvěma SOA větami.

V IXFR není možné vracet v odpovědi celou zónu. Pokud se zjistí, že změn v zóně je příliš a nevyplatí se použít IXFR, pak musí klient vyslat dotaz znovu a požádat o AXFR přenos.

11.17.3 Strategie čištění (purging)

IXFR server nemusí udržovat všechny předcházející verze zón, může staré verze kdykoli zrušit. U velkých a často se měnících zón můžeme narazit na rozpor mezi velikostí obsazeného diskového prostoru a možností dělat IXFR. Informace ze starších verzí souborů mohou být zahozeny v okamžiku, kdy by byl IXFR přenos delší než eventuální AXFR.

11.17.4 Příklady z RFC 1995

Předpokládejme tři verze dat pro zónu, aktuální je verze 3.

```
JAIN.AD.JP.         IN SOA NS.JAIN.AD.JP. mohta.jain.ad.jp. (
                                1 600 600 3600000 604800)
                    IN NS  NS.JAIN.AD.JP.
NS.JAIN.AD.JP.      IN A   133.69.136.1
NEZU.JAIN.AD.JP.    IN A   133.69.136.5
```

NEZU.JAIN.AD.JP. je zrušena a JAIN-BB.JAIN.AD.JP. je přidána.

```
jain.ad.jp.         IN SOA ns.jain.ad.jp. mohta.jain.ad.jp. (
                                2 600 600 3600000 604800)
                    IN NS  NS.JAIN.AD.JP.
```

```
NS.JAIN.AD.JP.      IN A   133.69.136.1
JAIN-BB.JAIN.AD.JP. IN A   133.69.136.4
                    IN A   192.41.197.2
```

Jedna z IP adres pro JAIN-BB.JAIN.AD.JP. je změněna.

```
JAIN.AD.JP.         IN SOA ns.jain.ad.jp. mohta.jain.ad.jp. (
                                3 600 600 3600000 604800)
                    IN NS  NS.JAIN.AD.JP.
NS.JAIN.AD.JP.      IN A   133.69.136.1
JAIN-BB.JAIN.AD.JP. IN A   133.69.136.3
                    IN A   192.41.197.2
```

IXFR dotaz

```
           +-------------------------------------------+
Header     | OPCODE=SQUERY                             |
           +-------------------------------------------+
Question   | QNAME=JAIN.AD.JP., QCLASS=IN, QTYPE=IXFR  |
           +-------------------------------------------+
Answer     |                                           |
           +-------------------------------------------+
Authority  | JAIN.AD.JP.         IN SOA serial=1       |
           +-------------------------------------------+
Additional |                                           |
           +-------------------------------------------+
```

může být zodpovězen pomocí předání celé zóny:

```
           +-------------------------------------------+
Header     | OPCODE=SQUERY, RESPONSE                   |
           +-------------------------------------------+
Question   | QNAME=JAIN.AD.JP., QCLASS=IN, QTYPE=IXFR  |
           +-------------------------------------------+
Answer     | JAIN.AD.JP.         IN SOA serial=3       |
           | JAIN.AD.JP.         IN NS  NS.JAIN.AD.JP. |
           | NS.JAIN.AD.JP.      IN A   133.69.136.1   |
           | JAIN-BB.JAIN.AD.JP. IN A   133.69.136.3   |
           | JAIN-BB.JAIN.AD.JP. IN A   192.41.197.2   |
           | JAIN.AD.JP.         IN SOA serial=3       |
           +-------------------------------------------+
Authority  |                                           |
           +-------------------------------------------+
Additional |                                           |
           +-------------------------------------------+
```

nebo posláním změněných dat:

```
            +-------------------------------------------------+
Header      | OPCODE=SQUERY, RESPONSE                         |
            +-------------------------------------------------+
Question    | QNAME=JAIN.AD.JP., QCLASS=IN, QTYPE=IXFR        |
            +-------------------------------------------------+
Answer      | JAIN.AD.JP.          IN SOA serial=3            |
            | JAIN.AD.JP.          IN SOA serial=1            |
            | NEZU.JAIN.AD.JP.     IN A   133.69.136.5        |
            | JAIN.AD.JP.          IN SOA serial=2            |
            | JAIN-BB.JAIN.AD.JP.  IN A   133.69.136.4        |
            | JAIN-BB.JAIN.AD.JP.  IN A   192.41.197.2        |
            | JAIN.AD.JP.          IN SOA serial=2            |
            | JAIN-BB.JAIN.AD.JP.  IN A   133.69.136.4        |
            | JAIN.AD.JP.          IN SOA serial=3            |
            | JAIN-BB.JAIN.AD.JP.  IN A   133.69.136.3        |
            | JAIN.AD.JP.          IN SOA serial=3            |
            +-------------------------------------------------+
Authority   |                                                 |
            +-------------------------------------------------+
Additional  |                                                 |
            +-------------------------------------------------+
```

nebo posláním kondenzovaných dat:

```
            +-------------------------------------------------+
Header      | OPCODE=SQUERY, RESPONSE                         |
            +-------------------------------------------------+
Question    | QNAME=JAIN.AD.JP., QCLASS=IN, QTYPE=IXFR        |
            +-------------------------------------------------+
Answer      | JAIN.AD.JP.          IN SOA serial=3            |
            | JAIN.AD.JP.          IN SOA serial=1            |
            | NEZU.JAIN.AD.JP.     IN A   133.69.136.5        |
            | JAIN.AD.JP.          IN SOA serial=3            |
            | JAIN-BB.JAIN.AD.JP.  IN A   133.69.136.3        |
            | JAIN-BB.JAIN.AD.JP.  IN A   192.41.197.2        |
            | JAIN.AD.JP.          IN SOA serial=3            |
            +-------------------------------------------------+
Authority   |                                                 |
            +-------------------------------------------------+
Additional  |                                                 |
            +-------------------------------------------------+
```

V budoucnu je možné očekávat, že se IXFR uplatní hlavně u velkých domén (com, cz, ...)

11.18 Negativní caching (DNS NCACHE)

Udržování negativních odpovědí na DNS dotazy definuje RFC1034 a RFC2308.

Pod pojmem **negativní caching** chápeme uložení informací v paměti serveru o tom, že neexistuje v DNS požadovaná RR věta nebo doménové jméno.

Dnes používané resolvery negenerují stejné negativní odpovědi na stejný dotaz. Aby bylo možné negativní odpovědi korektně používat, je třeba přesně definovat obsah negativní odpovědi a dobu, po kterou má být negativní odpověď udržována v paměti.

RFC1034 definovalo negativní caching jako nepovinný. Některé implementace BINDu negativní caching podporují. Negativní caching je např. implementován v BINDu 4.9.2. RFC2308 definuje negativní caching jako povinnou vlastnost resolveru a definuje obsah negativní odpovědi.

11.18.1 Jaké negativní odpovědi ukládat do paměti?

RFC2308 definuje jako povinné ukládání negativních odpovědí s RCODE nastaveným na NXDOMAIN a NOERROR_NODATA.

Chyba	Popis chyby
Name error (NXDOMAIN)	Doménové jméno uvedené v QNAME dotazu neexistuje. RCODE je nastaveno na NXDOMAIN. Autoritativní sekce může obsahovat SOA a NS věty.
NOERROR_NODATA	RCODE nastaveno na NOERROR, ale sekce odpovědi neobsahuje žádný RR record. Autoritativní sekce může obsahovat SOA větu a NS věty.

Tab. 11.11 Povinně ukládané negativní odpovědi

Zbylé typy negativních odpovědí jsou ponechány jako volitelné. Může se jednat např. o negativní odpovědi způsobené chybou name serveru (viz tab. 11.12).

Server failure	Server neposkytuje data o zóně z důvodu chybné konfigurace zóny. Server neposkytuje data o zóně z důvodu nedostupnosti master serveru pro zónu.
Dead / Unreachable server	Server neexistuje, nefunguje nebo není dostupný.

Tab. 11.12 Volitelně ukládané negativní odpovědi

Pokud server podporuje ukládání odpovědí jiných typů než NXDOMAIN a NOERROR_NODATA, nesmí tyto odpovědi udržovat v paměti déle než 5 minut. Jako součást ukládání informace musí udržovat i IP adresu serveru z odpovědi.

11.18.2 Jak dlouho udržovat negativní odpovědi v paměti?

Všechny RR věty uložené v paměti jsou považovány za platné, pokud mají TTL větší než 0. TTL je tedy rozhodující položka vzhledem k paměti. I negativní odpovědi, pokud mají být udržovány v paměti, musí mít definováno TTL. Jak však TTL negativní odpovědi určit, když negativní odpověď obvykle neobsahuje žádnou RR větu v sekci odpověď, jak je patrné z příkladu č. 1 z kapitoly 11.1.7?

Určení TTL pro negativní odpověď se řeší vložením SOA věty pro zónu do autoritativní sekce odpovědi.

11.18.3 Pole MINIMUM ve větě SOA

Pole MINIMUM ve větě SOA mělo postupně tři interpretace.

1. Minimum ttl pro všechny věty RR v zóně. (Nikdy se v praxi takto nepoužívalo.)
2. Implicitní ttl pro všechny věty RR v zóně, které neobsahují ttl pole. (Pouze na primary name serveru). Po provedení zone transferu mají všechny RR věty ttl pole naplněno.
3. ttl negativní odpovědi pro zónu.

Nadále se bude uplatňovat pole MINIMUM ve větě SOA ve významu 3. TTL pro jednotlivé RR věty musí být definována přímo v RR větách, nebo pomocí nového příkazu $TTL v zóně souboru.

Syntaxe příkazu:

$TTL ttl komentář

Všechny RR věty uvedené v souboru za příkazem $TTL, které nemají explicitně uvedeno ttl, přebírají ttl z příkazu $TTL.

Reálné TTL se určuje jako minimum z pole TTL obsaženého v SOA větě a z pole MINIMUM. TTL se u negativních odpovědí v paměti snižuje stejným způsobem jako u kladných odpovědí. Pokud je TTL u negativní odpovědi 0, je informace v paměti neplatná.

11.18.4 Pravidla ukládání negativních odpovědí

- Ukládání negativních odpovědí je povinné. Pokud resolver ukládá odpovědi do paměti, musí ukládat i negativní odpovědi.
- Neautorizované negativní odpovědi nemohou být ukládány.
- V paměti musí být uložena i věta SOA z autoritativní sekce odpovědi.
- Negativní odpověď bez věty SOA nesmí být ukládána.
- Věta SOA z paměti musí být přidávána do odpovědi.
- Odpověď NXDOMAIN musí být ukládána spolu s QNAME a QCLASS.
- Odpověď NOERROR_NODATA musí být ukládána spolu s QNAME, QTYPE a QCLASS.
- V master souboru musí být vložen příkaz $TTL.

11.19 Věta typu SRV

Věta typu SRV byla experimentálně zavedena v RFC-2052 a v RFC-2782 pak byla zavedena definitivně. Zásadní rozdíl mezi oběma normami spočívá zejména v tom, že RFC-2782 předřazuje znak podtržítko (_) před název služby a název protokolu. Věty typu SRV využívají Windows 2000.

Cílem věty SRV je v databázi DNS udržovat nejen jména počítačů, ale i jména služeb. Již jsme se s takovým případem setkali – větami MX kde službou byla elektronická pošta. Věty MX specifikují poštovní servery, na které se má pošta zasílat. Pomocí priority je vyjádřeno, který poštovní server má být kontaktován jako první a které poštovní servery následně.

Zastavme se u případu www-serveru. V praxi, pokud chceme získat nějaké informace o české firmě, tak do prohlížeče napíšeme:

```
http://www.firma.cz
```

tj. chceme protokolem HTTP kontaktovat www-server z domény firma.cz. Protokol HTTP využívá protokol TCP.

Věta typu SRV zavádí do DNS informace k nalezení příslušeného weboveho serveru. V našem případě budeme v DNS hledat jméno:

```
http.tcp.www.firma.cz.
```

Syntaxe věty SRV je (kód v protokolu DNS má pro větu typu SRV hodnotu 33):

```
_Služba._Protokol.doménové-jméno [TTL] IN SRV Priorita  Váha  Port Počítač.
```

Kde:

Služba specifikuje symbolické jméno služby (serveru). Např. ldap, http, smtp atd.

Protokol specifikuje protokol, např. tcp či udp.

Priorita určuje prioritu. Firma může provozovat několik www-serverů, aby v případě výpadku byl vždy nějaký server dostupný. Firma bude chtít zavést prioritu, který server se má klient pokoušet kontaktovat jako první a který jako další.

Váha – firma může mít web-server extrémně zatížen, proto zřídí několik paralelně běžících webových serverů o stejné prioritě. Každý bude spuštěn na počítači o jiném výkonu. Zavádí se proto váha (váha ve smyslu váženého průměru). V DNS jsou např. dva záznamy:

```
_http._tcp.www.firma.cz.  IN   SRV 10  1  80  server1.firma.cz.
                          IN   SRV 10  3  88  server2.firma.cz.
```

Jelikož oba záznamy mají stejnou prioritu (10), tak v případě, že oba servery jsou dostupné, může klient náhodně kontaktovat libovolný z nich. Avšak server2.firma.cz je výkonnější než server1.firma.cz. Váha proto sděluje, že servery mají být kontaktovány sice náhodně, ale při velkém počtu navazovaných spojení má být 25 % spojení navázáno se server1.firma.cz a 75 % spojení se server2.firma.cz, tj. server2 je třikrát výkonnější než server1.

Váha nula je určena pro administrátory, tj. neprovádí se žádné vyrovnávání výkonu mezi počítači.

Port specifikuje port, na kterém server běží.

Počítač specifikuje jméno počítače (odkaz na větu typu A), na kterém je služba poskytována (na kterém běží server). Pokud je jako počítač uvedena pouze tečka, pak služba není poskytována.

Příklad:

```
$ORIGIN         firma.cz.
@               IN      SOA  …
                IN      NS  …
…
; následující řádky specifikují, že protokolem telnet má být kontaktován buď server1
; nebo server2. Server2 je třikrát výkonější.
_telnet._tcp    IN      SRV     0       1       23      server1.firma.cz.
                IN      SRV     0       3       23      server2.firma.cz.
; v případě, že není dostupný ani server1 ani server2, pak administátor má kontaktovat
; server3:
                IN      SRV     10      0       23      server3.firma.cz.
; Jsou provozovány dva www-servery. Klient má kontaktovat server1 a v případě
; nedostupnosti počítače server1 má kontaktovat server2, ale na portu 88:
_http._tcp      IN      SRV     0       0       80      server1.firma.cz.
                IN      SRV     5       0       88      server2.firma.cz.
; Jelikož je zvykem v případě protokolu HTTP psát www před jméno domény,
; tak přidáme ještě:
_http._tcp.www  IN      SRV     0       0       80      server1.firma.cz.
                IN      SRV     5       0       88      server2.firma.cz.
; Nesmíme zapomenout na věty typu A. Raději znakem @ specifikujeme aktuální
; doménu (aby se jméno nevzalo z předchozí věty):
@               IN      A       10.1.1.2
                IN      A       10.1.1.2
; Pochopitelně uvedeme i věty typu A jednotlivých serverů:
server1         IN      A       10.1.1.1
server2         IN      A       10.1.1.2
server3         IN      A       10.1.1.3
; Ostatní služby nejsou podporovány:
*._tcp          IN      SRV     0       0       0       .
*._tcp          IN      SRV     0       0       0       .
```

Poznámka: V celé publikaci jsme se důsledně snažili vyhnout používání hvězdiček v doménovém jméně. V praxi jsme si totiž mnohokrát prověřili, že hvězdičky jsou ve jméně zdrojem neočekávaných chyb. Používají se proto pouze u vět typu MX a snad v budoucnu u vět typu SRV.

Jak taková hvězdička pracuje? Představme si případ věty typu A:

```
*.firma.cz      IN      A       10.1.1.10
```

Pak na jakýkoliv dotaz na prvek domény firma.cz explicitně neuvedený v DNS bude DNS odpovídat, že má adresu 10.1.1.10. Tj. pocitac1.firma.cz má adresu 10.1.1.10, pocitac2.firma.cz má také adresu 10.1.1.10 atd. I kdybychom si přáli, aby tomu tak bylo, pak v případě, že se spleteme a místo pocitac1.firma.cz napíseme poctac1.firma.cz, pak nam DNS nevrátí, že jsme se spletli, ale vrátí adresu, ale s největší pravděpodobností jinou, než čekáme.

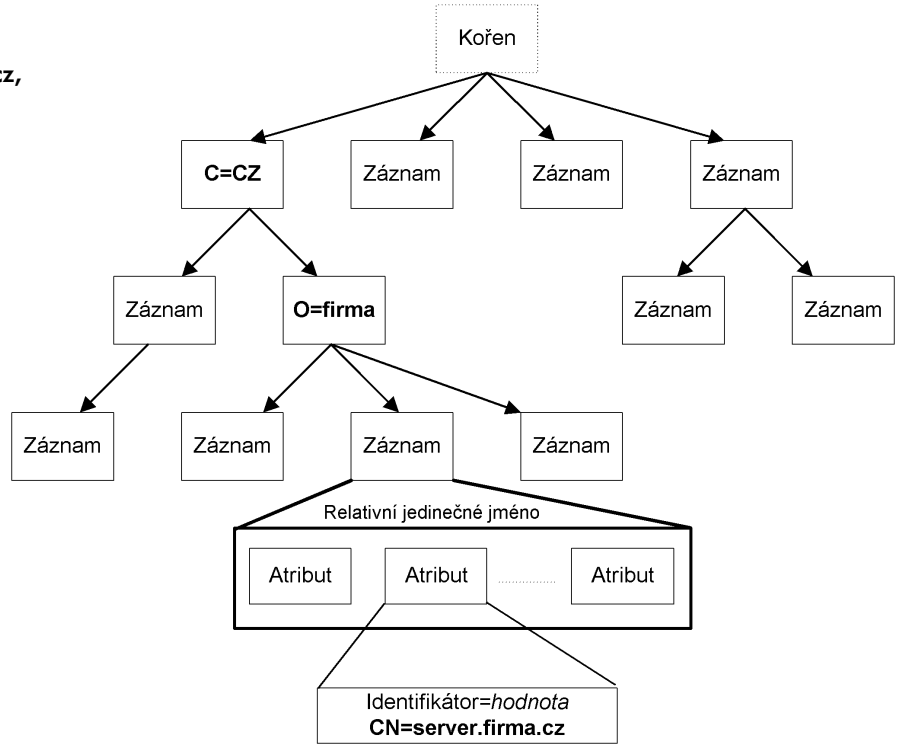
11.20 X.500 a LDAP

DNS tvoří celosvětovou databázi. Jiným konkurenčním systémem jsou adresářové služby podle normy X.500, tj. normy vydané ITU. Jedná se o celou řadu norem X.5xx.

Aby nedošlo k nedorozumění, musím v úvodu zdůraznit, že slovo adresář zde není míněno jako adresář souborů, ale v původním významu. Potřebujete-li nějakou představu, pak si představte jako adresář „bílé stránky“ telefonního seznamu.

Cílem X.500 je zavést celosvětové „bílé stránky“ všech objektů světa. Představme si, že bychom měli bílé stránky všech telefonních seznamů světa. Pokud bychom v nich chtěli něco nalézt, tak bychom si je museli nejprve roztřídit podle zemí, pak podle měst a podle telefonních operátorů. V konkrétním telefonním seznamu bychom pak vyhledávali konkrétní záznam. Záznam se častěji označuje jako relativní jedinečné jméno (*Relative Distinguished Name*).
Každý záznam adresářové struktury je složen z atributů. Záznam je zpravidla tvořen jedním adributem, ale může být tvořen i více atributy.
Jednotlivé záznamy tvoří stromovou strukturu – viz obr. 11.18.

Jedinečné jméno (*Distinguished Name*) konkrétního objektu světa je pak posloupnost všech relativních rozlišitelných jmen (*Distinguished Name*) od kořene adresáře.



Obr. 11.18
Jedinečné jméno CN= server.firma.cz, O=firma,C=CZ

11.20.1 Rozlišitelná jména

Na obr. 11.18 je absolutní rozlišitelné jméno CN= server.firma.cz,O=firma,C=CZ tvořeno jednotlivými záznamy (relativními rozlišitelnými jmény):

C=CZ

O=firma

CN= server.firma.cz

Každý záznam má tvar:

```
identifikátor atributu = hodnota
```

Identifikátory jednotlivých atributů záznamů vznikly zkrácením:

C – Country

O – Organization

CN – Common Name atd.

V počítačové komunikaci se nepoužívají slovní identifikátory atributů, ale každý identifikátor je určen svou hodnotou, tj. jednoznačnou řadou čísel. Tyto řady tvoří celosvětově jednoznačnou nomenklaturu identifikátorů objeků, která má opět stromovou strukturu.

Základní normou popisující jednotlivé záznamy (relativní rozlišitelná jména) je norma X.520, která mj. zavádí:

Identifikátor atributu	Zkratka	Význam	Hodnota identifikátoru
Country	C	Stát podle ISO 3166 (obdoba Top Level Domén)	{joint-iso-ccitt(2) ds(5) attributeType(4) 6}
State or Province	SP	Vyšší uzemně správní jednotka	{2 5 4 8}
Locality	L	Místo	{2 5 4 7}
Organization	O	Název organizace, firmy	{2 5 4 10}
Organization Unit	OU	Název části firmy	{2 5 4 11}
Street Address		Ulice a číslo domu	{2 5 4 9}
Common Name	CN	Název objektu, pod kterým je místně znám, např. jméno a příjmení, DNS-jméno serveru atp.	{2 5 4 3}

Velice důležitý atribut je atribut CN specifikující lokálně jednoznačný název objektu.

Uveďme několik příkladů (mohou být případně doplněny o další atributy SP, L atd.):

Libor Dostálek jako obyvatel Česka:

```
CN=Libor Dostálek,C=CZ
```

Libor Dostálek jako zaměstnance PVT, které je v Česku:

```
CN=Libor Dostálek,O=PVT,C=CZ
```

Libor Dostálek, zaměstnanec oddělení ET, které je součástí PVT:

```
CN=Libor Dostálek,OU=ET,O=PVT,C=CZ
```

Avšak, specifikace PVT jako firmy:

```
CN=PVT,C=CZ
```

Server firmy PVT:

```
CN=server.firma.cz,O=PVT,C=CZ
```

Firma RSA zavedla v normě PKCS#9 identifikátor atributu pro internetový e-mail:

Identifikátor	Zkratka	Význam	Hodnota identifikátoru
Email Address	Email	mailová adresa podle RFC-822.	{1 2 840 113549 1 9 1}

I když můžeme server specifikovat, tj. v *Common Name* uvedeme DNS-jméno serveru, tak chybí možnost specifikovat části serveru (např. procesy). RFC-2247 proto zavádí normu na specifikaci atributů pro doménová jména:

Identifikátor	Zkratka	Význam	Hodnota identifikátoru
Domain Component	DC	Jeden řetězec v doménovém jméně	{1 3 6 1 4 1 1466 115 121 1 26}

Před tím, než si uvedeme příklad, musíme upozornit na skutečnost, že jeden záznam může obsahovat více atributů.

Příklad:

Relativní jedinečné jméno (záznam) pro doménové jméno server.cbu.firma.cz lze zapsat jako:

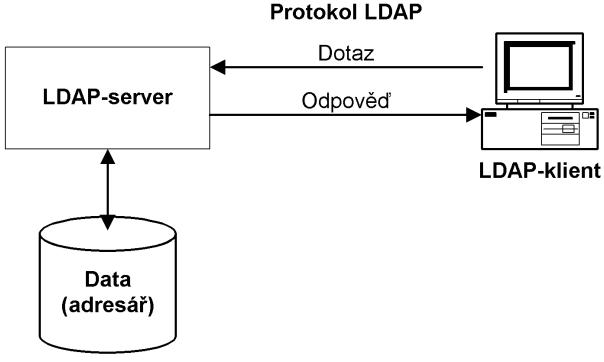
```
DC=server,DC=cbu,DC=firma,DC=CZ
```

11.20.2 LDAP

X.500 používá pro přístup do svých adresářů protokol *X.500 Directory Access Protocol* (DAP). Zjednodušením protokolu DAP vznikl v Internetu protokol *Lightweight Directory Access Protocol* (LDAP). Protokoly DAP i LDAP jsou aplikační protokoly.

Protokol LDAP verze 3 je specifikován normami: RFC-2251 až RFC-2255. Pro programátory je navrženo API v RFC-1823.

Obr. 11.19 LDAP-klient pokládá dotaz LDAP-serveru, který odpovídá



Protokolem LDAP se provádí dotazy do adresáře. Tyto dotazy lze provádět např. webovým prohlížečem. Uvedeme příklady URL (viz RFC-2255):

- Výpis záznamu o objektu CZ (LDAP-server běží na počítači server.ica.cz):

  ```
  ldap://server.ica.cz/C=CZ
  ```

- Výpis záznamu o objektu „Libor Dostalek“ z LDAP-serveru server.firma.cz (podezřelé znaky jako mezera se nahrazují obdobně jako v protokolu HTTP znakem procenta následovaným hexadecimální hodnotou znaku):

  ```
  ldap://server.firma.cz/CN=Libor%20Dostalek,C=CZ
  ```

- Výpis záznamu o objektu, který by měl obsahovat např. mailovou adresu:

  ```
  ldap://server.firma.cz/CN=dostalek,O=PVT,DC=pvt,DC=cz
  ```

- Avšak výpis samotné e-mailové adresy by byl:

```
ldap://server.firma.cz/CN=Libor%20Dostalek,OU=PVT,C=CZ?email
```

11.21 Přehled norem týkajících se DNS

- *RFC-1032 – Domain administrators guide.*
- *RFC-1033 – Domain administrators operations guide*
- *RFC-1034 – Domain names – concepts and facilities.*
- *RFC-1035 – Domain Names – Implementation and specification*
- *RFC-1101 – DNS encoding of network names and other types*
- *RFC-1183 – New DNS RR Definitions*
- *RFC-1123 – Requirements for Internet hosts – application and support.*
- *RFC-1183 – New DNS RR Definitions*
- *RFC-1348 – DNS NSAP RRs*
- *RFC-1383 – An Experiment in DNS Based IP Routing.*
- *RFC-1394 – Relationship of Telex Answerback Codes to Internet Domains*
- *RFC-1401 – Correspondence between the IAB and DISA on the use of DNS*
- *RFC-1464 – Using the Domain Name System To Store Arbitrary String Attributes.*
- *RFC-1535 – A Security Problem and Proposed Correction With Widely Deployed DNS Software*
- *RFC-1536 – Common DNS Implementation Errors and Suggested Fixes*
- *RFC-1537 – Common DNS Data File Configuration Errors*
- *RFC-1591 – Domain Name System Structure and Delegation*
- *RFC-1611 – DNS Server MIB Extensions.*
- *RFC-1612 – DNS Resolver MIB Extensions.*
- *RFC-1637 – DNS NSAP Resource Records*
- *RFC-1664 -Using the Internet DNS to Distribute RFC1327 Mail Address Mapping Tables*
- *RFC-1706 – DNS NSAP Resource Records.*
- *RFC-1712 – DNS Encoding of Geographical Location*
- *RFC-1713 – Tools for DNS debugging*
- *RFC-1788 – ICMP Domain Name Messages*
- *RFC-1794 – DNS Support for Load Balancy*
- *RFC-1876 – A Means for Expressing Location Information in the Domain Name System*
- *RFC-1886 – DNS Extensions to support IP version 6*
- *RFC-1912 – Common DNS Operational and Configuralion Errors*
- *RFC-1982 – Serial Number Arithmetic*
- *RFC-1995 – Incermental Zone Transfer in DNS*
- *RFC-1996 – A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)*

- *RFC-2010 – Operational Criteria for Root Name Servers*
- *RFC-2052 – A DNS RR for specifying the location of services (DNS SRV)*
- *RFC-2065 – Domain Name System Security Extensions*
- *RFC-2136 – Dynamic Update in the Domain Name System (DNS UPDATE)*
- *RFC-2137 – Secure Domain Name System*
- *RFC-2163 – Using the Internet DNS to Distribute MIXER Conformant Global Address Mapping*
- *RFC-2168 – Resolution of Uniform Resource Identifiers using the Domain Name Systém*
- *RFC-2181 – Clarifications to the DNS Specification*
- *RFC-2182 – Selection and Operation of Secondary DNS Servers.*
- *RFC-2219 – Use of DNS Aliases for Network Services.*
- *RFC-2230 – Key Exchange Delegation Record for the DNS.*
- *RFC-2240 – A Legal Basis for Domain Name Allocation*
- *RFC-2247 – Using Domains in LDAP/X.500 Distinguished Names.*
- *RFC-2308 – Negative Caching of DNS Queries (DNS NCACHE)*
- *RFC-2317 – Classless IN-ADDR.ARPA delegation*
- *RFC-2345 – Domain Names and Company Name Retrieval*
- *RFC-2352 – A Convention For Using Legal Names as Domain Names*
- *RFC-2517 – Building Directories from DNS*
- *RFC-2535 – Domain Name System Security Extensions*
- *RFC-2536 – DSA KEYs and SIGs in the Domain Name System (DNS)*
- *RFC-2537 – RSA/MD5 KEYs and SIGs in the Domain Name System (DNS)*
- *RFC-2538 – Storing Certificates in the Domain Name System (DNS)*
- *RFC-2539 – Storage of Diffie-Hellman Keys in the Domain Name System (DNS).*
- *RFC-2540 – Detached Domain Name System (DNS) Information.*
- *RFC-2541 – DNS Security Operational Considerations.*
- *RFC-2606 – Reserved Top Level DNS*
- *RFC-2671 – Extension Mechanisms for DNS*
- *RFC-2672 – Non-Terminal DNS Name Redirection*
- *RFC-2673 – Binary Labels in the Domain Name System*
- *RFC-2694 – DNS extensions to Network Address Translators (DNS_ALG)*
- *RFC-2782 - A DNS RR for specifying the location of services (DNS SRV)*

Kromě norem RFC, které jsou jistě základem, jsou pravidla dále určena regionálními normami (pro Evropu normami RIPE – viz http://www.ripe.net).

12

Implementace jmenného serveru

12.1 Implemetace v Unixu (program named verze 4)

Programem *named* je v Unixu realizován jmenný server. Program *named* si při startu přečte databáze DNS na disku a uloží je do paměti.

Program named si při svém startu přečte konfigurační soubor **named.boot**. Konfigurační soubor je implicitně umístěn v adresáři */etc*, jiné umístění konfiguračního souboru je nutné specifikovat parametrem na příkazovém řádku při spouštění programu *named*. Konfigurační soubor named.boot obsahuje následující příkazy:

`directory` – specifikuje adresář, ve kterém jsou uloženy databáze DNS na disku. V dalších příkazech se pak uvádějí jména souborů bez cesty. Příklad:

```
directory       /etc/namedb/namedb
```

`primary` – určuje, že jmenný server bude primárním jmenným serverem pro doménu uvedenou jako první parametr příkazu a příslušná databáze je v textovém souboru uvedeném jako druhý parametr. Příklad:

```
primary         pvt.cz                  db.pvt.cz
```

Každý jmenný server musí být primárním jmenným serverem alespoň pro doménu *0.0.127.in-addr.arpa*. Tj. ani v případě cacheování pouze jmenného serveru nesmí v jeho konfiguračním souboru chybět např. příkaz:

```
primary         0.0.127.in-addr.arpa    db.0.0.127
```

`secondary` – určuje, že jmenný server bude sekundárním jmenným serverem pro doménu určenou prvním parametrem. Další parametry (musejí být uvedeny jako IP-adresy) jsou pak IP-adresy serverů, odkud se budou programem named-xfer přenášet data. Je-li uveden poslední parametr (nesmí být ve for-

mě IP-adresy), pak se chápe jako jméno souboru, kam se mají data po přenesení na počítači uložit. Příklad:

```
secondary cbu.pvt.cz    172.17.14.1     172.17.18.1     cbu.pvt.cz.tmp
```

ukazuje, že autoritativní data o doméně **cbu.pvt.cz** se mají kopírovat ze serveru o IP-adrese 172.17.14.1 a uložit do souboru cbu.pvt.cz.tmp. Když tento server není dostupný, pak se zkopírují ze serveru 172.17.18.1.

Není-li uvedeno jméno souboru (poslední parametr), pak se přenesená data neukládají na disk.

`cache` – určuje z jakého souboru se mají zkopírovat do paměti informace o kořenových jmenných serverech. Příklad:

```
cache           .       cache.db
```

říká, že do paměti se mají uložit ze souboru cache.db informace o kořenových jmenných serverech. Tento soubor neobsahuje autoritativní data, tj. na počátku neobsahuje záznam SOA, takže každý záznam musí obsahovat explicitně uvedené všechny údaje – zejména pole TTL.

Avšak pozor, kořenový jmenný server bude mít v konfiguračním souboru příkaz:

```
primary         .       db.root
```

Kde soubor db.root bude obsahovat obdobná data jako soubor cache.db, avšak bude obsahovat na počátku záznam typu SOA a jednotlivé záznamy souboru nemusí obsahovat pole TTL – převezme se jeho hodnota ze záznamu typu SOA.

`forwarders` – určuje, že jmenný server je forwarding server. Jako další parametry se uvádějí IP-adresy jmenných serverů dostupných v Internetu (forwarderů). Příklad:

```
forwarders 193.85.240.40 193.85.240.40
```

Ne, neudělal jsem chybu, že jsem napsal stejnou IP-adresu dvakrát. To je u forwarderů častý trik. Tím se totiž zvětší časový interval, po který forwarding server čeká na odpověď, než začne sám kontaktovat kořenové jmenné servery.

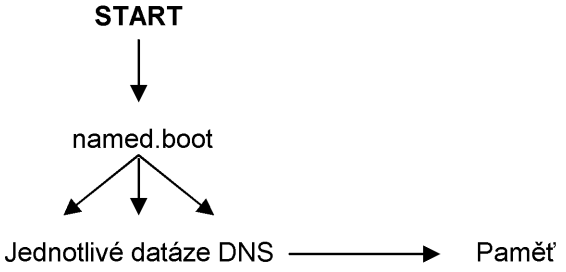
`slave` – následuje v případě podřízeného serveru (slave serveru) za příkazem forwarders a určuje, že jmenný server nemá kontaktovat v žádném případě kořenové jmenné servery. Příklad:

```
forwarders 193.85.240.40 193.85.240.40
slave
```

Příklad konfiguračního souboru pro primární jmenný server domény pvt.cz:

```
directory  /etc/namedb
primary         pvt.cz                  db.pvt.cz
primary         17.172.in-addr.arpa     db.172.17
primary         0.0.127.in-addr.arpa    db.127.0.0
cache           .                       db.cache
```

12.1.1 Databáze



Obr. 12.1
Program *named* při svém startu zjišťuje informace o databázích DNS v souboru named.boot

Databáze DNS jsou uloženy na primárním jmenném serveru v souborech. Při startu jmenného serveru se jejich obsah nahraje do paměti.

Databáze DNS se skládá z jednotlivých souborů, které jsou specifikovány vždy jako poslední parametry jednotlivých příkazů konfiguračního souboru named.boot. Databáze na disku může obsahovat následující typy dat:

- Autoritativní data k obhospodařované zóně. Musí začínat záznamem typu SOA. Mohou být udržována pouze na primárním jmenném serveru. Sekundární jmenný server je obdrží pomocí dotazu zónového přenosu z primárního nebo jiného sekundárního jmenného serveru.
- Data umožňující přístup ke kořenovým jmenným serverům. Nezačínají záznamem SOA. Explicitně se v nich uvádí pole TTL. Jsou to neautoritativní data pro místní jmenný server. Musí být na každém jmenném serveru s výjimkou podřízeného serveru.
- Data pomocí nichž deleguje jmenný server autoritu k subdoménám na jiné jmenné servery. K delegování autority se používají věty typu NS.
- V případě, že se deleguje autorita na jmenný server, jehož doménové jméno je součástí delegované subdomény, pak je třeba ještě přidat záznam typu A specifikující IP-adresu tohoto jmenného serveru.

Obecná syntaxe jednotlivých řádků databáze (tzv. *resource records*) je:

```
[name]      [TTL]   třída   typ     Data_závislá_na_typu_věty
```

Pole v [] jsou nepovinná – jejich hodnoty se přejímají z předchozího záznamu, případně ze záznamu SOA. Komentáře jsou uvozeny středníky.

Význam jednotlivých polí:

- Pole `name` obsahuje doménové jméno. Může nastat několik situací:
  - Pole není vyplněné, pak se jeho hodnota bere z pole `name` předchozího záznamu.
  - V záznamu typu SOA může mít pole `name` hodnotu `@`. Takováto hodnota znamená, že se do pole `name` má dosadit jméno domény uvedené v příslušném příkazu souboru named.boot.
  - Doménové jméno je uvedeno v poli `name` bez tečky na konci, pak se automaticky za toto jméno dodá jméno domény uvedené ve větě SOA. V případě, že před větou (bez tečky na konci) je uveden příkaz $ORIGIN, pak se dodává jméno domény uvedené v příkazu $ORIGIN.

  - Doménové jméno je uvedeno v poli `name` s tečkou na konci, pak se jedná o tzv. absolutní jméno, které se bere tak, jak je napsáno.
- Pole `TTL` obsahuje ve vteřinách dobu života záznamu v paměti. Jmenný server tuto hodnotu automaticky snižuje. Dosáhne-li hodnota nuly, pak se záznam prohlásí za neplatný. Implicitně má pole hodnotu nula. Avšak pokud předchází záznamu záznam typu SOA, pak se implicitní hodnota bere z pole TTL záznamu typu SOA. Záznam typu SOA je uveden vždy na počátku souboru, tj. nemusí náš záznam předcházet zcela bezprostředně.
- Třída je IN (Internet), HS (Hesiod) či CH (Chaos). V našem případě se budeme zabývat výhradně záznamy typu IN. (Existují i implementace programu named podporující Hesiod, my se jimi však zabývat nebudeme.)
- Typ je jeden z typů uvedených v tab. 11.1.
- Poslední pole obsahuje data závislá na typu záznamu. Pokud se zde použije doménové jméno, pak nesmíme za ním zapomenout napsat tečku, protože v opačném případě by se za jméno automaticky dodalo jméno domény. Naopak pokud je zde IP-adresa, tak za čtvrtou číslicí v IP-adrese tečka být nesmí.

Blíže viz RFC-1035.

12.1.2 Formát záznamů

Jména v databázi musí začínat na první pozici. Pokud je prvním znakem řádku mezera, pak se použije jméno z předchozího řádku. Soubor se skládá z typů záznamů (tzv. *resource records*) uvedených v tab. 11.1.

12.1.2.1 SOA

Záznam typu **SOA** (*Start Of Authority*) – určuje jmenný server, který je autoritativním zdrojem informací pro danou doménu. Věta SOA je vždy právě jedna, a to na počátku souboru.

Jako příklad uveďme záznam pro server domény **pvt.cz**.

```
@       IN SOA cbu.pvt.cz. bindmaster.cbu.pvt.cz. (
                        1               ;Serial
                        86400           ;Refresh after 24 hours
                        600             ;Retry after 5 min.
                        120960  ;Expire after 2 weeks
                        86400)  ;Minimum TTL of 1 day
```

- Jméno pvt.cz, musí začínat od první pozice na řádku a musí končit tečkou. Označuje název zóny. Zpravidla se zde místo jména zóny napíše @, který říká, aby se název zóny vzal z druhého parametru (tj. jména domény) příkazu souboru named.boot.
- **IN** označuje typ adresy (IN Internet)
- První jméno za SOA (cbu.pvt.cz.) je jméno počítače, na kterém jsou data uložena (jméno primárního jmenného serveru) a druhé jméno (bindmaster.cbu.pvt.cz.) definuje poštovní adresu osoby zodpovědné za data. Jelikož znak @ má v záznamu SOA zvláštní význam, tak se místo znaku @ v poštovní adrese píše tečka, tj. místo bindmaster@cbu.pvt.cz se píše bindmaster.cbu.pvt.cz.
- Závorky umožňují pokračování záznamu na dalších řádcích (pro přehlednost).

- **Serial** označuje sériové číslo verze databázového souboru. Pokud soubor změníme, musíme zvětšit i toto sériové číslo. Doporučujeme zásadně používat číslo tvaru rrrrmmddčč (rok, měsíc, den, číslo aktualizace v rámci dne).

  Sekundární jmenný server se dotazuje primárního jmenného serveru pouze na záznam typu SOA. Porovná hodnotu pole `serial` v záznamu typu SOA, a pouze v případě, že primární jmenný server má v záznamu typu SOA hodnotu pole `serial` větší než má sekundární jmenný server, pak se provede přenos zóny z primárního jmenného serveru na sekundární jmenný server.

  Takže pokud správce opraví databázi DNS na primárním jmenném serveru a opomene zvýšit hodnotu pole `serial`, pak se žádná sekundární data nepřenesou a změny se na sekundární jmenný server vůbec nedostanou. Pokud takovouto chybu správce primárního jmenného serveru zjistíte, pak máte jedinou možnost zrušit na sekundárním jmenném serveru soubor pro příslušnou zónu, program named na sekundárním jmenném serveru ukončit a znovu jej nastartovat.

  Hodnota pole `serial` neovlivňuje chování primárního jmenného serveru, tj. pokud pole opomenete na primárním jmenném serveru zvýšit, tak po restartu jmenného serveru se na primárním jmenném serveru změny provedou.

- Další údaje udávají různé časové údaje v sekundách:
  - **Refresh** jak často mají sekundární servery ověřovat svá data. Zjistí-li při ověřování, že mají data s nižším serial, pak provedou protokolem TCP transfer zóny.
  - **Retry** jestliže sekundární server nemůže kontaktovat primární po uplynutí intervalu refresh, bude to dále zkoušet každých retry sekund.
  - **Expire** jestliže se sekundárnímu jmennému serveru nepodaří kontaktovat primární do expire sekund, přestane poskytovat informace (data jsou příliš stará). Musí platit: **Expire > Refresh.**
  - **TTL** tato hodnota se vztahuje ke všem záznamům v db souboru (jako výchozí hodnota) a je poskytována jmenným serverem v každé jeho odpovědi. Říká, jak dlouho mohou ostatní servery (tj. neautoritativní servery) udržovat daný záznam ve své paměti cache (nula znemožňuje ukládání vět do cache).

Nevíte-li, co zadat, pak RFC-1537 doporučuje pro top level domény:

```
  86400 ; Refresh     24 hours
   7200 ; Retry        2 hours
2592000 ; Expire      30 days
 345600 ; Minimum TTL  4 days
```

Pro ostatní domény:

```
  28800 ; Refresh     8 hours
   7200 ; Retry       2 hours
 604800 ; Expire      7 days
  86400 ; Minimum TTL 1 day
```

12.1.2.2 A

Záznamy typu **A** (*Address*) přiřazují doménovým jménům počítačů IP-adresy. Za IP-adresou nesmí být tečka.

Příklad 1:

```
pvt.cz.                 IN      SOA     …
…
www                     IN      A       172.17.14.1
www.cbu                 IN      A       172.17.18.1
muj.cbu.pvt.cz.         IN      A       172.17.14.2
tvuj                    IN      A       10.1.1.3
…
```

V příkladě 1 záznamu typu A přiřazují IP-adresy počítačům: `www.pvt.cz`, `www.cbu.pvt.cz`, `muj.cbu.pvt.cz` a `tvuj.pvt.cz`.

12.1.2.3 CNAME

Větami typu CNAME vytváříme synonyma k doménovým jménům. Často se říká, že vytváříme „aliasy k jménům počítačů".

Příklad 2:

```
firma.cz.               IN      SOA     …
…
mail                    IN      A       192.1.1.2
www                     IN      CNAME   mail.firma.cz.
…
```

Příklad 2 popisuje situaci, kdy firma má jeden počítač mail.firma.cz, který chce také používat jako www-server.

Na pravé straně příkazu CNAME musí být doménové jméno, kterému je přiřazena IP-adresa záznamem typu A. Na pravé straně nesmí být synonymum, tj. CNAME nesmí ukazovat na CNAME. Příklad 3 ukazuje chybnou delegaci jmen.

Příklad 3 (chybný!):

```
firma.cz.               IN      SOA     …
…
mail                    IN      A       192.1.1.2
www                     IN      CNAME   mail.firma.cz.
server                  IN      CNAME   www.firma.cz.
…
```

V záznamech typu CNAME se zásadně snažíme na pravé straně zapisovat úplné doménové jméno s tečkou na konci. V případě, že tečku neuvedeme, pak se dodá ještě jméno domény. Toho se sice v malých databázích dá využít, ale jak velikost databáze roste, tak se stává nepřehlednou a případné chyby tohoto druhu se někdy i těžko dohledávají.

12.1.2.4 HINFO a TXT

Záznamy typu HINFO a TXT jsou pouze informativní záznamy.

Záznam typu HINFO má ve své datové části dva údaje. Prvním údajem je informace o hardwaru a druhým údajem informace o softwaru.

Záznam typu TXT obsahuje ve své datové části obecný textový řetězec.

Příklad 4:

```
firma.cz.               IN      SOA     …
…
mail                    IN      A       192.1.1.2

                        IN      HINFO   AlphaServer UNIX
                        IN      TXT     Muj server
…
```

12.1.2.5 NS

Záznamy typu NS definují autoritativní jmenné servery pro doménu. Na pravé straně musí být doménové jméno, kterému je přiřazena IP-adresa větou typu A. Na pravé straně nesmí být synonymum, tj. záznam typu NS nesmí ukazovat na záznam typu CNAME.

Stejné záznamy typu NS jsou vždy ve dvou databázích:

1. V databázi domény vyšší úrovně. Těmito záznamy typu NS je delegována pravomoc na jmenný server nižší úrovně. V případě, že jméno jmenného serveru nižší úrovně samo leží v doméně nižší úrovně, pak musí být za tento záznam typu NS přidán ještě záznam typu A s IP-adresou jmenného serveru. To je nutné proto, že jmenný server vyšší úrovně musí IP-adresu jmenného serveru nižší úrovně uvádět v dodatečných informacích v DNS-odpovědi – tj. aby bylo možné se na jmenný server nižší úrovně vůbec dostat.
2. Na autoritativním jmenném serveru pro doménu (tj. podle terminologie předchozího bodu na jmenném serveru "nižší úrovně").

Příklad 5:

Jmenný server domény firma.cz, tj. autoritativní jmenný server domény vyšší úrovně s přiřazeným neautoritativním záznamem typu A pro ns.cbu:

```
firma.cz.               IN      SOA     …
                        IN      NS      ns.provider.cz.
                        IN      NS      ns.firma.cz
ns                      IN      A       11.1.1.1
cbu                     IN      NS      ns.firma.cz.
                        IN      NS      ns.cbu.firma.cz.
ns.cbu                  IN      A       11.2.2.2
…
```

Jmenný server domény cbu.firma.cz, tj. autoritativní jmenný server domény nižší úrovně má k dispozici databázi:

```
cbu.firma.cz.           IN      SOA     …
                        IN      NS      ns.firma.cz.
                        IN      NS      ns.cbu.firma.cz.
ns                      IN      A       11.2.2.2
…
```

Opět musíme zdůraznit, že na pravé straně vět typu NS je třeba psát úplná doménová jména s tečkou na konci.

12.1.2.6 MX

Záznamy typu MX specifikují poštovní server domény. V drtivé většině případů totiž nechceme mít mailovou adresu tvaru:

```
uživatel@počítač.firma.cz
```

ale pouze tvaru:

```
uživatel@firma.cz
```

tj. přejeme si skrýt jméno poštovního serveru.

Záznam typu MX ukazuje na jaký počítač má být pro doménu dopravena pošta. Navíc však v záznamu typu MX je číselná priorita, pomocí které lze určit několik počítačů, na než může být pošta pro doménu posílána. Nejprve se zkouší odeslat pošta na počítač s nejvyšší prioritou (nejnižším číslem), když se to nepodaří, pak na další počítač (s vyšším číslem) atd. Příklad 6 popisuje situaci, kdy pošta pro doménu firma.cz má být doručována na počítač mail.firma.cz, pokud tento počítač není dostupný, pak se pošta odešle na počítač mail1.provider.cz, kde vyčká do dostupnosti počítače mail.firma.cz. Pokud ani počítač mail1.provider.cz není dostupný, pak se pošta odešle na počítač mail2.provider.cz.

Příklad 6:

```
firma.cz.               IN      SOA     …
                        IN      MX      30 mail2.provider.cz.
                        IN      MX      20 mail1.provider.cz.
                        IN      MX      10 mail.firma.cz.
mail                    IN      A       11.1.1.8
…
```

12.1.2.7 PTR

Záznam typu PTR slouží k překladu IP-adresy na doménové jméno. Tj. k překladu prvků domény in-addr.arpa na jméno počítače.

Příklad 7

Záznamy typu PTR pro počítač `ns.firma.cz` o IP-adrese 195.47.200.1 a pro počítač `www.firma.cz` o IP-adrese 195.47.200.201:

```
1.200.47.195.in-addr.arpa.      IN      PTR     ns.firma.cz.
201.200.47.195.in-addr.arpa.    IN      PTR     www.firma.cz.
```

Jenže takovýto příklad je zcela vytržený z kontextu. Ve skutečnosti je třeba si uvědomit celý mechanismus delegací. Náš příklad je podrobně popsán v příkladu 8.

Příklad 8:

Předpokládejme, že naší firmě (firma.cz) jsou přiděleny IP-adresy v rozsahu 195.47.200.0/24, tj. celá síť třídy C. Pak pro reverzní překlad musí být:

1. **Na jmenném serveru ns.ripe.net, tj. jmenný server vyšší úrovně pro Evropu (Amsterodam):**
   A. V souboru named.boot bude uveden např. řádek

```
pirmary        195.in-addr.arpa195.rev
```

   B. V souboru 195.rev je pak např. uvedeno:

```
195.in-addr.arpa.              IN      SOA     …
…
200.47                 IN      NS      ns.firma.cz.
                               IN      NS      ns.provider.cz.
…
```

2. **Na jmenném serveru ns.firma.cz (primární jmenný server):**
   A. V souboru named.boot bude uveden např. řádek:

```
primary        200.47.195.in-addr.arpa 200.47.195.rev
```

   B. V souboru 200.47.195.rev je pak např. uvedeno:

```
200.47.195.in-addr.arpa.       IN      SOA     …
                       IN      NS      ns.firma.cz.
                               IN      NS      ns.provider.cz.
1                              IN      PTR     ns.firma.cz.
201                    IN      PTR     www.firma.cz.
…
```

3. **Na jmenném serveru ns.provider.cz (sekundární jmenný server)**
   V souboru named.boot bude uveden např. řádek:

```
secondary 200.47.195.in-addr.arpa 195.47.200.1 200.47.195.rev
```

Je třeba opět připomenout, že se nesmí zapomínat psát tečky za jménem počítače (na pravé straně), protože v případě opomenutí tečky se dodá doména končící in-addr.arpa, což je opravdu nepoužitelné.

Asi čekáte, že také zdůrazním, že na pravé straně nesmí být synonymum (CNAME), tj. že záznam typu PTR nesmí ukazovat na větu typu CNAME. Avšak není tomu tak. Dlouhá léta to bylo považováno za chybu v programu BIND, až se to stalo velice užitečnou pomůckou a posléze dokonce normou RFC-2317. Využití tohoto mechanismu se věnuje kap. 16.

12.1.2.8 $ORIGIN

V parametru `name` databázového záznamu se uvádí doménové jméno buď absolutně (s tečkou na konci) nebo relativně (bez tečky na konci). Právě za relativní doménové jméno se automaticky přidává implicitní doména. Příkaz $ORIGIN slouží pro změnu implicitní domény. Databáze DNS může obsahovat příkaz:

```
$ORIGIN implicitní_doména
```

Od tohoto okamžiku je implicitní doména změněna na hodnotu uvedenou jako první parametr příkazu $ORIGIN.

V případě, že se uvede relativní jméno, pak se doplňuje na úplné jméno přidáním domény definované v záznamu typu SOA nebo parametrem příkazu $ORIGIN, který předchází databázový záznam. Příkaz $ORIGIN tak mění implicitně nastavenou doménu.

Není-li změněna implicitní doména příkazem $ORIGIN, pak se bere doména z příkazu SOA. Je-li v příkazu SOA místo domény znak @, pak se bere první parametr příkazu primary resp. secondary ze souboru /etc/named.boot.

12.1.2.9 $INCLUDE

Do zdrojového souboru na disku je možné vložit další soubor příkazem:

```
$INCLUDE soubor
```

Soubor se vloží na místo příkazu. Je možné uvést ještě:

```
$INCLUDE soubor implicitní_doména
```

Tím se nejenom vloží soubor, ale změní se i implicitní doména. Změna implicitní domény platí jen pro řádky vloženého souboru.

12.1.3 Signály kill

Programu named je možné v Unixu vyslat signál příkazem kill. Pomocí signálů lze spustit diagnostiku. Zpravidla jsou zpracovávány následující signály: HUP, INT, IOT, KILL, TERM, USR1 a USR2. V konkrétní implementaci jmenného serveru závisí též na parametrech, se kterými byl program named kompilován.

Příkaz kill má jako druhý parametr číslo procesu (PID). Číslo procesu pod kterým běží program named lze zjistit např. příkazem ps. Avšak program named při svém startu zapíše číslo procesu do souboru */cesta*/named.pid. Umístění i jméno souboru je možné ovlivnit při kompilaci programu named.

Syntaxe příkazu kill je např. pro signál HUP následující:

```
kill  -HUP `cat /cesta/named.pid`
```

Chcete-li spustit diagnostiku programu named již při jeho startu, tak zadejte příslušný parametr na příkazovém řádku, kterým se named spouští. Blíže viz příkaz:

```
man named
```

Signál HUP

Signálem HUP donutíte jmenný server znovu načíst data z disku. Ovšem zpravidla se signálem HUP nevyčistí cache.

Signál INT

Signál INT způsobí vypsání všech dat (tj. autoritativních i neautoritativních) z paměti do souboru, který se zpravidla jmenuje /tmp/named_dump.db. Příklad části souboru:

```
; Dumped at Fri Feb 16 18:12:49 1996
; Note: Cr=(auth,answer,addtnl,cache) tag only shown for non-auth RR's
; Note: NT=milliseconds for any A RR which we've used as a nameserver
; -- Cache & Data --
$ORIGIN .
.       518339  IN      NS      A.ROOT-SERVERS.NET.
        518339  IN      NS      H.ROOT-SERVERS.NET.
        518339  IN      NS      B.ROOT-SERVERS.NET.
```

```
        518339  IN      NS      C.ROOT-SERVERS.NET.
        518339  IN      NS      D.ROOT-SERVERS.NET.
        518339  IN      NS      E.ROOT-SERVERS.NET.
        518339  IN      NS      I.ROOT-SERVERS.NET.
        518339  IN      NS      F.ROOT-SERVERS.NET.
        518339  IN      NS      G.ROOT-SERVERS.NET.
        86348   IN      SOA     A.ROOT-SERVERS.NET. HOSTMASTER.INTERNIC.NET. (
                1996021400 10800 900 604800 86400 )     ;Cr=addtnl
;workgroup      548     IN      A       NXDOMAIN;-$
cz      172768  IN      NS      NS.EUNET.CZ.    ;Cr=addtnl
        172768  IN      NS      NS.CESNET.CZ.   ;Cr=addtnl
        172768  IN      NS      NS.EU.NET.      ;Cr=addtnl
        172768  IN      NS      SUNIC.SUNET.SE. ;Cr=addtnl
        172768  IN      NS      NS.UU.NET.      ;Cr=addtnl
        172768  IN      NS      SPARKY.ARL.MIL. ;Cr=addtnl
$ORIGIN 48.192.IN-ADDR.ARPA.
96      518384  IN      NS      NS.UU.NET.      ;Cr=addtnl
        518384  IN      NS      UUCP-GW-1.PA.DEC.COM.   ;Cr=addtnl
        518384  IN      NS      UUCP-GW-2.PA.DEC.COM.   ;Cr=addtnl
        518384  IN      NS      NS.EU.NET.      ;Cr=addtnl
$ORIGIN 96.48.192.IN-ADDR.ARPA.
16      86385   IN      PTR     relay6.UU.NET.
$ORIGIN 147.IN-ADDR.ARPA.
230     518391  IN      NS      BUBO.VSLIB.CZ.  ;Cr=addtnl
        518391  IN      NS      NS.CESNET.CZ.   ;Cr=addtnl
$ORIGIN 16.230.147.IN-ADDR.ARPA.
1       3591    IN      PTR     bubo.vslib.cz.
$ORIGIN 0.127.IN-ADDR.ARPA.
0               IN      SOA     mh.pvt.cz. hostmaster.pvt.cz. (
                94082701 10800 3600 360000 129600 )
                IN      NS      mh.pvt.cz.
$ORIGIN 0.0.127.IN-ADDR.ARPA.
1               IN      PTR     localhost.
$ORIGIN 85.193.IN-ADDR.ARPA.
240             IN      SOA     mh.pvt.cz. hostmaster.pvt.cz. (
                1996020801 28800 3600 604800 864000 )
                IN      NS      mh.pvt.cz.
                IN      NS      runner.pvt.cz.
                IN      NS      ns.eunet.cz.
$ORIGIN 240.85.193.IN-ADDR.ARPA.
1               IN      PTR     Ceske-Budejovice.pvt.cz.
$ORIGIN MIL.
ARL     518368  IN      NS      ADMII.ARL.mil.  ;Cr=addtnl
        518368  IN      NS      VGR.ARL.ARMY.mil.       ;Cr=addtnl
        518368  IN      NS      SLADW.ARL.mil.  ;Cr=addtnl
        518368  IN      NS      DNS1.ARL.mil.   ;Cr=addtnl
$ORIGIN ARL.MIL.
DNS1    518368  IN      A       131.218.24.3    ;Cr=addtnl
SLADW   518368  IN      A       155.148.8.2     ;Cr=addtnl
        518368  IN      A       155.148.6.90    ;Cr=addtnl
ADMII   518368  IN      A       128.63.31.4     ;Cr=addtnl
        518368  IN      A       128.63.5.4      ;Cr=addtnl
```

```
        518368  IN      A       192.5.25.5      ;Cr=addtnl
SPARKY  81548   IN      A       128.63.48.85    ;NT=481 Cr=answer
        81548   IN      A       192.5.23.200    ;NT=745 Cr=answer
$ORIGIN ARL.ARMY.MIL.
VGR     518368  IN      A       128.63.16.6     ;Cr=addtnl
        518368  IN      A       128.63.4.4      ;Cr=addtnl
        518368  IN      A       128.63.2.6      ;Cr=addtnl
$ORIGIN SUNET.SE.
SUNIC   172768  IN      A       192.36.125.2    ;NT=459 Cr=addtnl
        172768  IN      A       192.36.148.18   ;NT=436 Cr=addtnl
$ORIGIN COM.
GreatCircle     172787  IN      NS      MILES.GreatCircle.COM.  ;Cr=addtnl
        172787  IN      NS      NS.UU.NET.      ;Cr=addtnl
        3591    IN      A       198.102.244.34
pvt             IN      SOA     mh.pvt.cz. hostmaster.pvt.cz. (
                1996020802 10800 3600 360000 129600 )
                IN      NS      mh.pvt.cz.
                IN      NS      runner.pvt.cz.
                IN      NS      ns.eunet.cz.
                IN      MX      10 mh.pvt.cz.
                IN      MX      20 bb-prg.eunet.cz.
                IN      MX      150 mcsun.eu.net.
                IN      MX      200 relay1.uu.net.
                IN      MX      200 relay2.uu.net.
$ORIGIN pvt.cz.
Ceske-BudejoviceIN      A       193.85.240.1
                IN      HINFO   „Cisco“ „“
$ORIGIN unl.pvt.cz.
p56x01          IN      MX      10 mh.pvt.cz.
                IN      MX      20 bb-prg.eunet.cz.
$ORIGIN NET.
pvt     172781  IN      NS      ns.pvt.net.     ;Cr=addtnl
        172781  IN      NS      NS1.PVT.NET.    ;Cr=addtnl
        172781  IN      NS      NS0.PIPEX.NET.  ;Cr=addtnl
        172781  IN      NS      NS1.PIPEX.NET.  ;Cr=addtnl
$ORIGIN ROOT-SERVERS.NET.
A       518339  IN      A       198.41.0.4      ;NT=475 Cr=addtnl
B       518339  IN      A       128.9.0.107     ;NT=16833 Cr=addtnl
C       518339  IN      A       192.33.4.12     ;NT=19544 Cr=addtnl
D       518339  IN      A       128.8.10.90     ;NT=1040 Cr=addtnl
E       518339  IN      A       192.203.230.10  ;NT=1279 Cr=addtnl
F       518339  IN      A       192.5.5.241     ;NT=1076 Cr=addtnl
G       518339  IN      A       192.112.36.4    ;NT=411 Cr=addtnl
H       518339  IN      A       128.63.2.53     ;NT=19544 Cr=addtnl
I       518339  IN      A       192.36.148.17   ;NT=940 Cr=addtnl
$ORIGIN UU.NET.
NS      172787  IN      A       137.39.1.3      ;NT=773 Cr=addtnl
$ORIGIN EU.NET.
NS      172784  IN      A       192.16.202.11   ;NT=280 Cr=addtnl
$ORIGIN pipex.NET.
ns0     172781  IN      A       158.43.128.8    ;Cr=addtnl
NS1     172781  IN      A       158.43.192.7    ;Cr=addtnl
```

```
$ORIGIN pvt.NET.
ns1     172781  IN      A       194.149.103.201 ;Cr=addtnl
ns      172781  IN      A       194.149.105.18  ;Cr=answer
; —- Hints —-
$ORIGIN .
.       3600    IN      NS      NS.INTERNIC.NET.
        3600    IN      NS      NS1.ISI.EDU.
        3600    IN      NS      C.NYSER.NET.
        3600    IN      NS      TERP.UMD.EDU.
        3600    IN      NS      NS.NASA.GOV.
        3600    IN      NS      NS.NIC.DDN.MIL.
        3600    IN      NS      AOS.ARL.ARMY.MIL.
        3600    IN      NS      NIC.NORDU.NET.
$ORIGIN NIC.DDN.MIL.
NS      3600    IN      A       192.112.36.4
$ORIGIN ARL.ARMY.MIL.
AOS     3600    IN      A       128.63.4.82
        3600    IN      A       192.5.25.82
$ORIGIN NASA.GOV.
NS      3600    IN      A       128.102.16.10
        3600    IN      A       192.52.195.10
$ORIGIN UMD.EDU.
TERP    3600    IN      A       128.8.10.90
$ORIGIN ISI.EDU.
NS1     3600    IN      A       128.9.0.107
$ORIGIN NYSER.NET.
C       3600    IN      A       192.33.4.12
$ORIGIN NORDU.NET.
NIC     3600    IN      A       192.36.148.17
$ORIGIN INTERNIC.NET.
NS      3600    IN      A       198.41.0.4      ;NT=683
```

Signál IOT

Signál IOT způsobí vypsání statistiky zpravidla do souboru /tmp/named.stats. Příklad:

```
### (824490113) Fri Feb 16 18:01:53 1996
551359  time since boot (secs)  počet sekund od startu
551359  time since reset (secs)
631708  input packets           počet vstupních paketů
637573  output packets          počet výstupních paketů
621627  queries                 počet dotazů
0       iqueries                počet inverzních dotazů
552     duplicate queries       počet zopak. dotazů po dosažení intervalu
13053   responses               počet odpovědí od vzdálených jmenných serverů
282     duplicate responses     počet duplikovaných odpovědí od jmenných serverů
426098  OK answers              počet odpovědí bez příznaku chyby
178     FAIL answers            počet odpovědí s příznakem chyby
2       FORMERR answers         počet odepřených odpovědí
3525    system queries          počet dotazů lokálního serveru
3       prime cache calls       kolikrát se načetly údaje o kořenových serverech
2       check_ns calls          kolikrát vypršelo pole TTL větám
```

```
                              popisujícím přístup na kořenové jmenné
                              servery, po takovém vypršení pro
                              příslušný soubor znovu načte.
345     bad responses dropped  počet chybných odpovědí od vzdálených serverů
2       martian responses      počet odpovědí, které odeslali"marťani"
                              (odpovědí od neznámých vzdálených serverů)
194894  negative responses cached      počet "kešovaných" negativních odpovědí
0       Unknown query types    počet dotazů na neznámé typy záznamů
520940  A queries              počet dotazů na záznamy typu:A
14      NS queries                                          NS
316     CNAME queries                                       CNAME
819     SOA queries                                         SOA
2       MR queries                                          MR
13045   PTR queries                                         PTR
86064   MX queries                                          MX
2       AXFR queries                                        AXFR(zone transfer)
425     ANY queries                                         ANY (*)
```

Signál KILL

Abnormální ukončení programu named.

Signál TERM

Řádné ukončení programu named.

Signály USR1 a USR2

Signálem USR1 se zapíná ladící výstup do souboru /tmp/named.run. Dalším signálem USR1 se zvyšuje úroveň ladění, tj. množství zaznamenávaných informací. Úrovní je až 11. Signálem USR2 se ladící výstup zcela vypíná (nikoliv, že by se postupně snižovala úroveň ladění). Ladící výstup zachycuje jednotlivé kroky jmenného serveru.

Následuje příklad ladění úrovně 1. Jedná se o překlad jména test97.pvt.net na IP-adresu. Jelikož bylo zadáno jméno bez tečky, tak nejprve byla za jméno doplněna implicitní doména pvt.net. Přeložit test97.pvt.net.pvt.cz se nepodařilo, tak následuje pokus přeložit test97.pvt.net. Dotaz byl odeslán autoritativnímu jmennému serveru pro doménu pvt.net, který má IP-adresu 158.43.128.8.

```
Debug turned ON, Level 1        (Kill   -USR1 ...)

datagam from [193.85.240.30].1824, fd 5, len 39; now Fri Feb 16 18:18:56 1996
req: nlookup(test97.pvt.net.pvt.cz) id 512 type=1
req: found 'test97.pvt.net.pvt.cz' as 'pvt.cz' (cname=0)
ns_req: answer - [193.85.240.30].1824 fd=5 id=2 Local

datagram from [193.85.240.30].1825, fd 5, len 32; now Fri Feb 16 18:18:56 1996
req: nlookup(test97.pvt.net) id 768 type=1
req: found 'test97.pvt.net' as 'pvt.net' (cname=0)
forw: forw - [158.43.128.8].53 ds=7 nsid=72 id=3 0ms retry 4sec

datagram from [158.43.128.8].53, fd 5, len 196; now Fri Feb 16 18:18:57 1996
update_msg: msglen:196, c:9
update failed (-10)
```

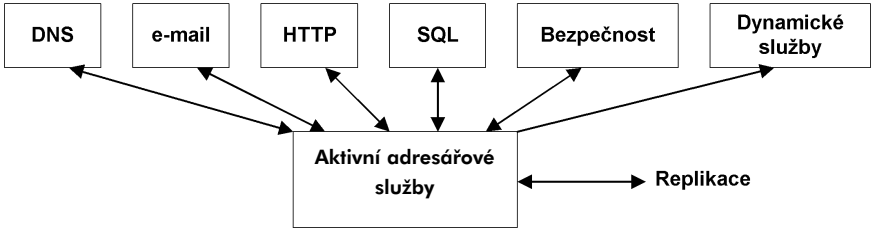
```
send_msg - [193.85.240.30] (UDP 5 1825) id=3
Debug turned OFF                (kill   -USR2 ...)
```

12.2 Implementace ve Windows 2000

Zatímco pro Windowns NT verze 3.51 bylo DNS nechtěným dítětem a až ve Windows NT verze 4 se DNS již objevilo oficiálně, tak ve Windows 2000 je DNS součástí komponenty Aktivní adresářové služby (*Active Directory*), která je stavebním pilířem Windows 2000.

Jádrem jsou aktivní adresářové služby, jejichž součástí je DNS. Kromě DNS obsahují i další data, takže aktivní adresářové služby jsou využívány i jako adresář pro elektronickou poštu, jsou využívány webovými aplikacemi, SQL atd.

Obr. 12.2 Aktivní adresářové služby



Aktivní adresářové služby jsou černou skříňkou, ve které jsou uložena data. Uživatele nemá zajímat jak jsou tam data uložena, ale pouze jak k nim přistupovat. K datům uloženým v aktivních adresářových službách se přistupuje zpravidla jednou z následujících tří metod:

1. Pomocí DNS.
2. Pomocí protokolu LDAP.
3. Přístup z programů je pomocí aplikačního programového rozhraní (API) pro jazyky C/C++.

Správa DNS se tak na rozdíl od správy DNS v operačním systému UNIX stává velice komplikovanou záležitostí.

12.2.1 Domain Controler a DNS

Windows vždy rozlišovaly mezi doménami Windows (protokol NetBIOS) a doménami DNS (tj. TCP/IP). Windows 2000 sjednocují jména v obou těchto doménách. Přesněji řečeno přejímají pro domény Windows názvy DNS.

Aktivní adresářové služby mají svá data udržována serverem, který se nazývá *Domain Controler* (DC). DC může být v síti více, mezi jednotlivými DC téže domény se provádí multi-master replikace. Data jsou udržována na všech DC současně (tj. neexistují již *Primary* a *Backup Domain Controlers,* jak tomu bylo u starších verzí Windows NT).

Jmenný server může ve Windows 2000 běžet samostatně, tj. i bez aktivovaných aktivních adresářových služeb.

Konfigurační soubor DNS je na jmenném serveru uložen buďto v databázi *registry Windows 2000,* nebo v adresářích aktivních adresářových služeb, nebo v souboru, který má formát jako konfigurační soubor programu *named* verze 4 (tj. soubor *named.boot*, ale soubor se jmenuje jen *boot*).

Existuje celá řada kombinací jmenného serveru a aktivních adresářových služeb, např.:

- Ve Windows 2000 neběží ani jmenný server, ani aktivní adresářové služby.
- Ve Windows 2000 běží jmenný server a nejsou aktivovány aktivní adresářové služby. Pak se jedná o obdobu UNIXu.
- Ve Windows 2000 běží jmenný server i aktivní adresářové služby:
  - Klient řeší překlady pomocí resolveru, tj. přes jmenný server (tj. porty 53).
  - Klient řeší překlady přes aktivní adresářové služby, pak ani jmenný server nepotřebuje. V intranetu může běžet pouze primární jmenný server a zálohy se provádí přes další DC.

V síti může být udržováno více domén. Každá doména má svůj DC. Vyhledávání informací v takto rozvětveném systému by bylo náročné, proto je zaveden tzv. globální katalog. Zatímco mezi dvěma DC téže domény se provádí úplná replikace, tak na globální katalog se replikují ze všech DC pouze globální informace, tj. provádí se jen částečná replikace nejdůležitějších údajů o každém objektu a informace kde nalézt další údaje. Globální katalog zrychluje vyhledávání údajů v aktivních adresářových službách.

12.2.2 DNS

Windows 2000 podporují jednak dynamické přidělování IP-adres protokolem DHCP, také však podporují dynamické DNS, tj. dynamický UPDATE DNS (viz kapitola 11.15). Tj. při zapnutí stanice se jí nejen přidělí IP-adresa, ale toto přidělení se dynamicky zanese do DNS.

Navíc Windows 2000 používají věty typu SRV, takže při startu služby se služba dynamicky zaregistruje do DNS.

Windows 2000 zavádí i nové termíny:

- **Strom domén** (*Domain Tree*) – doména může být dělena na subdomény. Stromem se rozumí doména se všemi svými subdoménami.
- **Les domén** (*Forest*) – firma může obhospodařovat několik domén (resp. několik stromů domén). Např. domény firma1.cz, firma2.cz a firma3.cz. Firma však neobhospodařuje doménu cz, tj. firmou obhospodařované domény netvoří strom, ale několik samostatných stromů – tj. les.

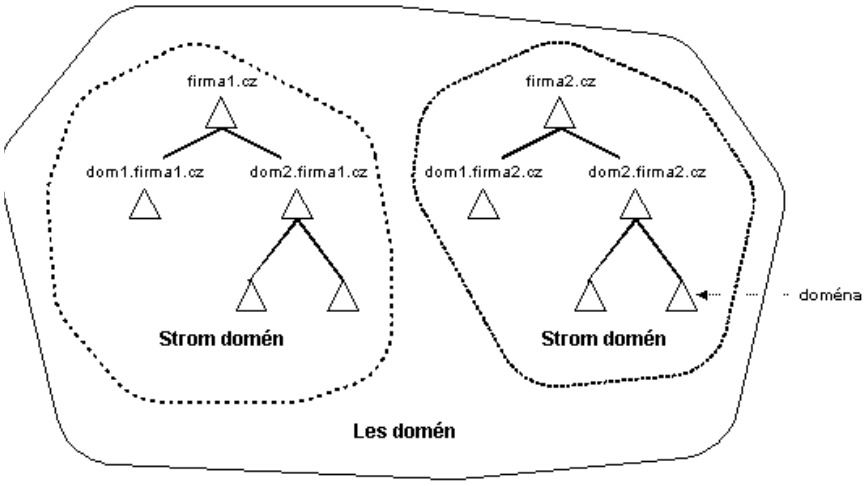
Máme-li instalovány Windows 2000, pak můžeme začít s aktivací jmenného serveru. Pomocí volby *Start -> Programs -> Administrative Tools -> DNS* spustíme DNS manager (obr. 12.13 a 12.11). V okně DNS manageru zvolíme *Action -> Configure the server,* jak je znázorněno na obr. 12.4.

Tím se automaticky spustí průvodce instalací DNS serveru (viz obr. 12.5).

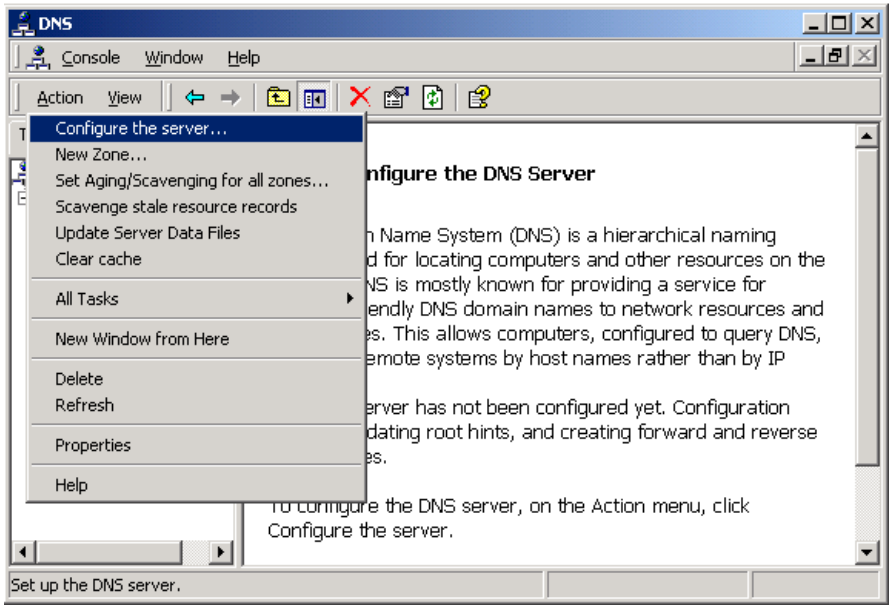
Prvním dotazem průvodce instalací je, zdali budeme vytvářet zónu pro překlad jmen na IP-adresy (viz obr. 12.6).

Dalším dotazem je, zdali budeme vytvářet primární nebo sekundární jmenný server pro zónu (viz obr. 12.7). Třetí možností (na obr. 12.7 jako první zašedlá možnost) je zónu umístit přímo do Aktivních adresářových služeb.

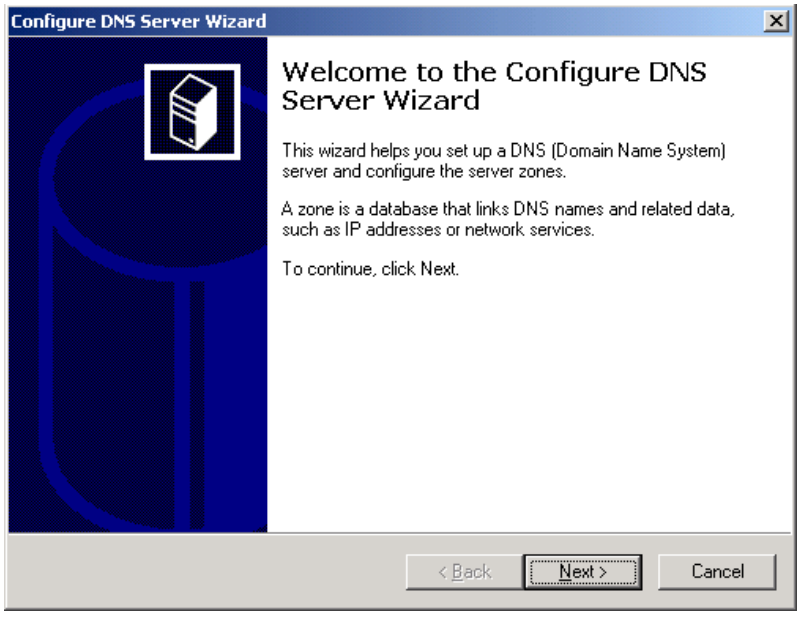
Obr. 12.3
Doména, strom domén a les domén



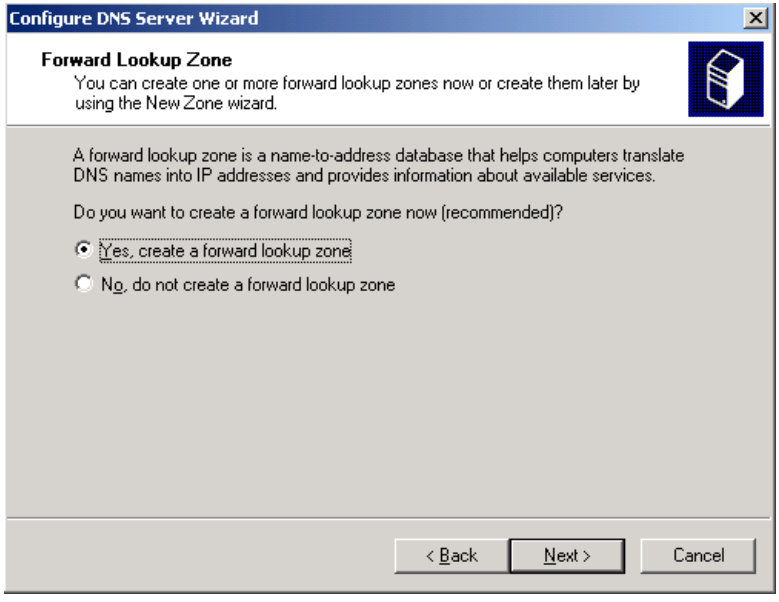
Obr. 12.4
DNS manager



Obr. 12.5
Průvodce instalací DNS serveru

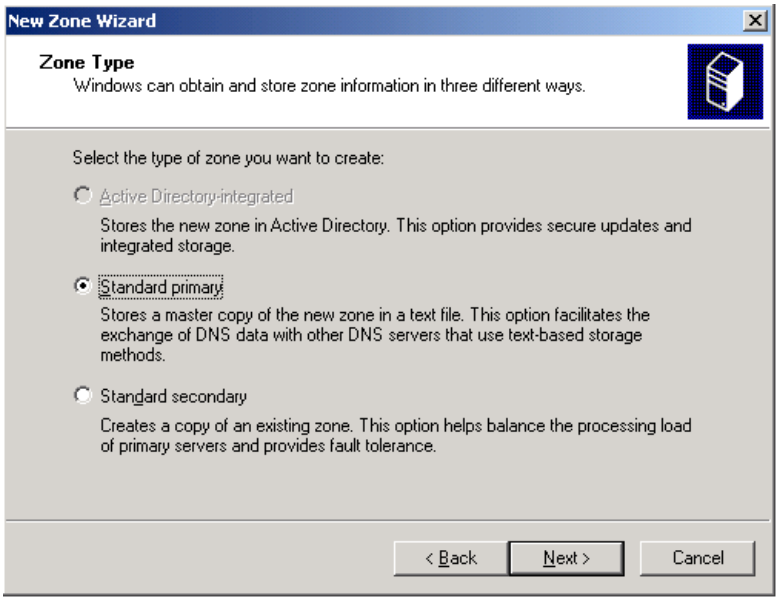


Obr. 12.6
Budeme vytvářet zónu pro překlad jmen na IP-adresy?

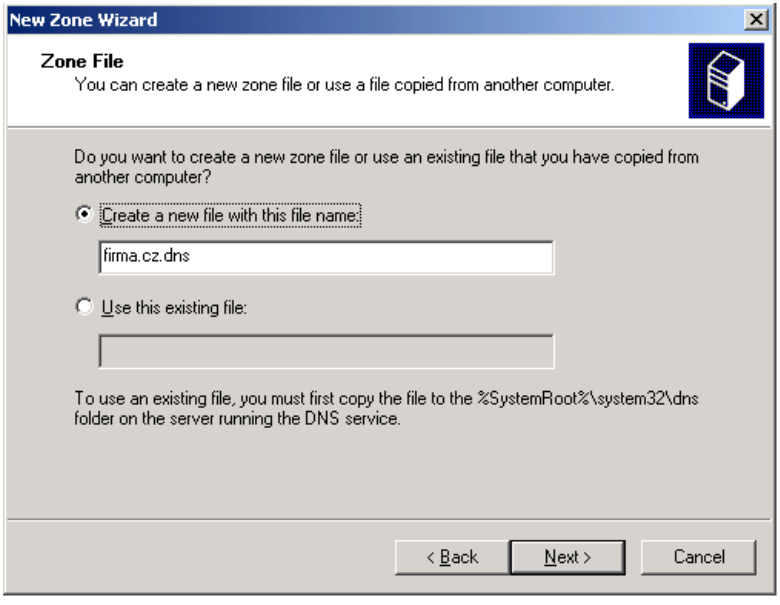


Dále jsme dotázáni, zdali chceme vytvořit nový nebo použít již existující soubor pro uložení zónových dat na disku (viz obr. 12.8).

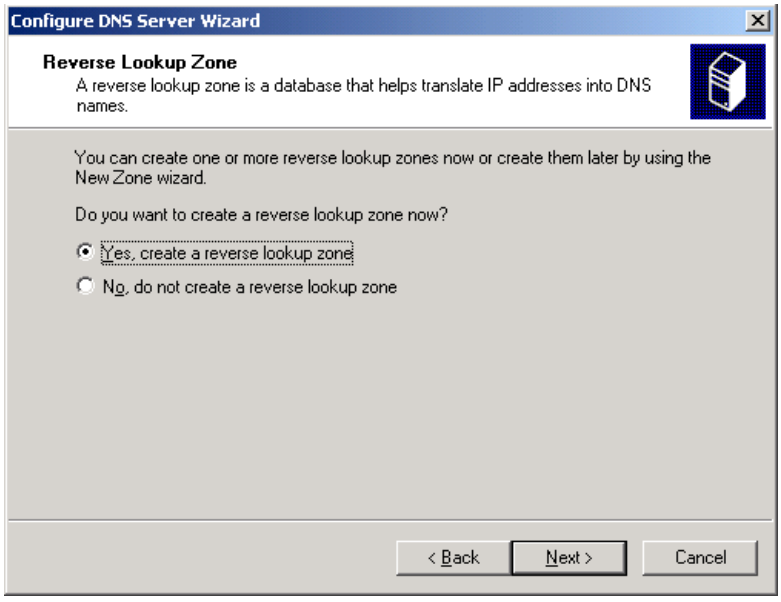
Obr. 12.7
Zóna bude v Aktivních adresářových službách, jako primární jmenný server či jako sekundární jmenný server?



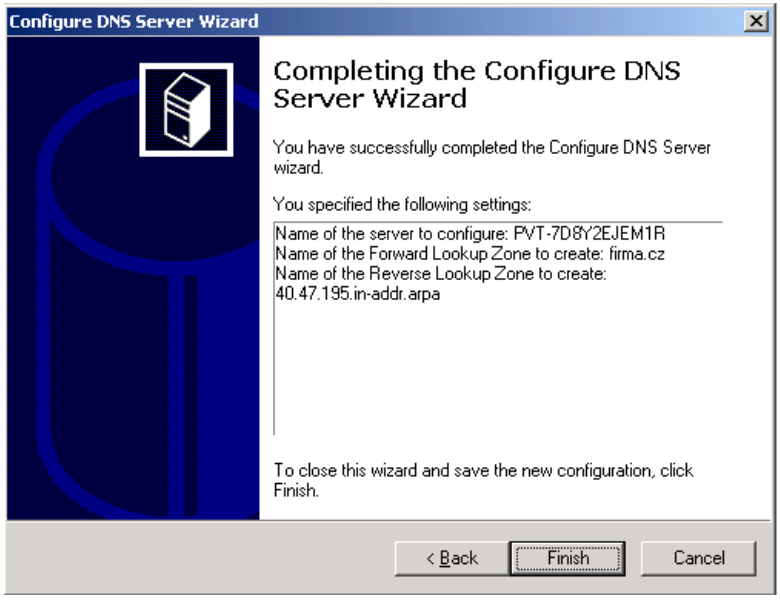
Obr. 12.8
Vytvořit nový či použít existující zónový soubor?



Následuje dotaz, zdali budeme vytvářet také zónu pro reverzní překlad (viz obr. 12.9). Pro reverzní zónu jsme dotazováni na analogické informace jako u zóny.



Obr. 12.9
Budeme také vytvářet reverzní zónu?



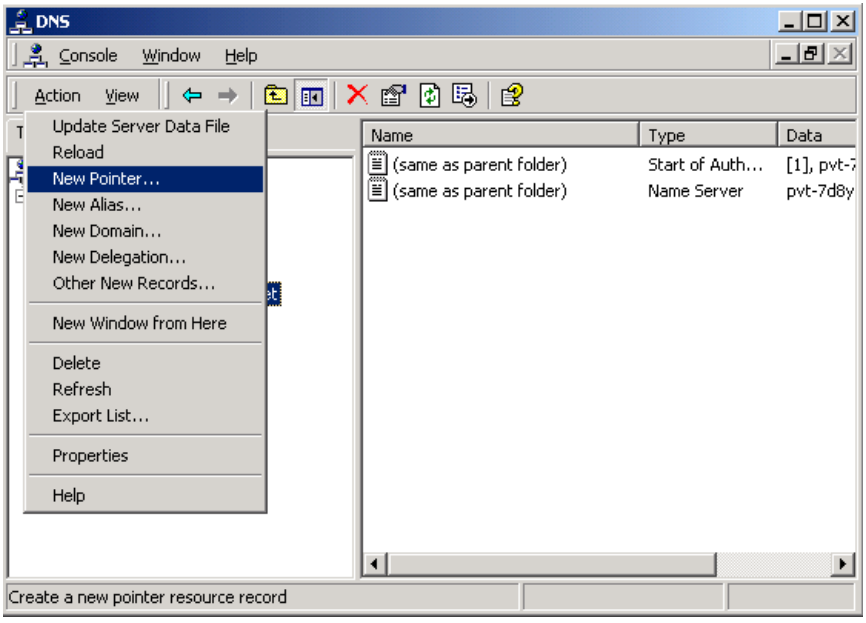
Obr. 12.10
Poslední okno průvodce instalací DNS serveru

Na obr. 12.10 je zobrazeno poslední okno průvodce instalací DNS serveru obsahující rekapitulaci nastavení.

Nyní se můžeme vrátit k práci s DNS managerem. Volbou *Action* provádíme základní akce s DNS managerem, jako je vkládání nových záznamů do databáze, restart databáze atd. Na obr. 12.11 je zvýrazněna volba *Action -> New Pointer* pro vložení nové věty typu PTR do databáze. Na obr. 12.12 je pak zobrazeno okno s příkladem vkládání údajů do věty typu PTR.

Pokud zvýrazníme reverzní zónu do které byl právě vložen nový PTR záznam (viz obr. 12.13), pak v pravé části okna tento nový záznam uvidíme.

Obr. 12.11
Volba pro vložení nové věty PTR do databáze DNS



DNS manager se sice ovládá pomocí menu, ale praktičtější je ovládání pravým tlačítkem myši. Tj. zvolíme příslušný objekt a po zmáčknutí pravého tlačítka se zpravidla objeví možnost vkládání objektů, rušení objektů a zejména vlastnosti objektů.

DNS server se po konfiguraci zpravidla sám nastartuje. Existuje několik možností, jak jej nastartovat/zastavit/restartovat. Kromě volby v DNS manageru máme možnost tyto akce provádět i pomocí příkazu *net* v okně MS-DOS. DNS server nastartujeme příkazem:

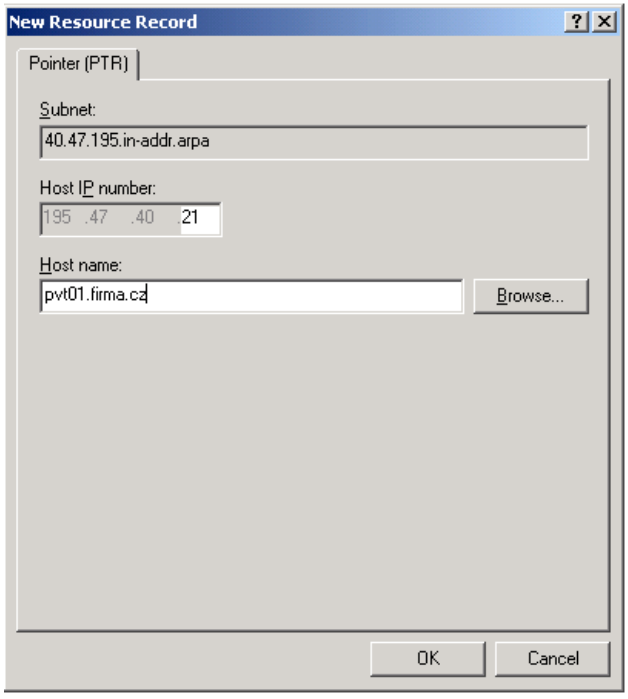
```
C:\ >net start dns
```

příkazem:

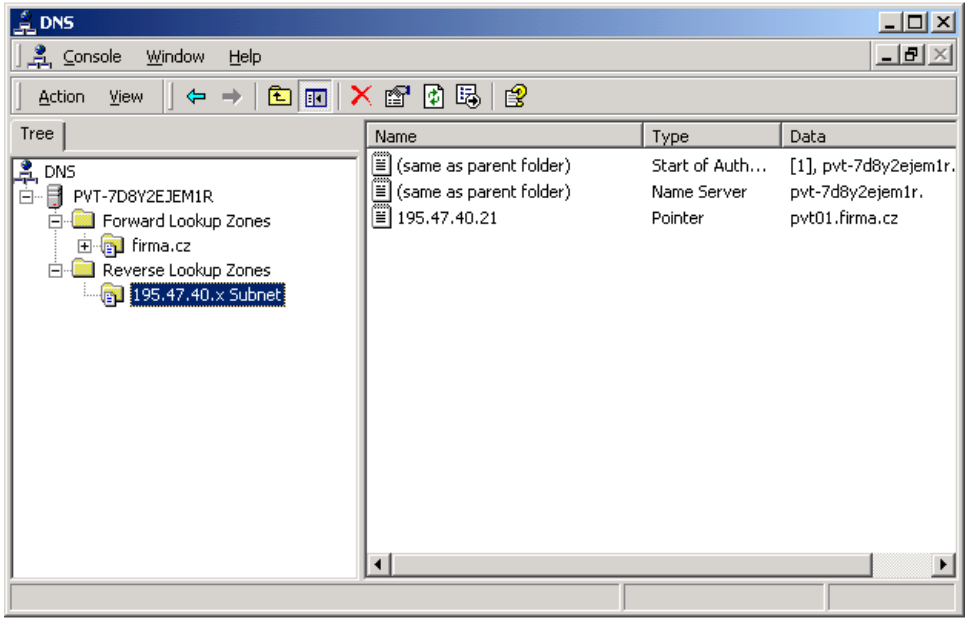
```
C:\ >net stop dns
```

bychom DNS zase ukončili.

Pomocí klepnutí pravým tlačítkem myši na jmenný server v okně DNS manageru lze získat i vlastnosti jmenného serveru. Na obr. 12.14 je zobrazena volba Advanced z vlastností jmenného serveru.

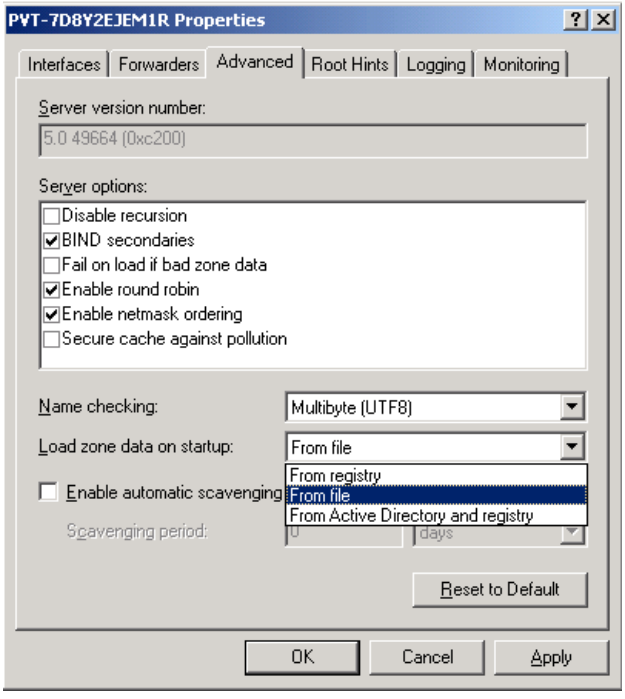


Obr. 12.12
Vkládání údajů do věty typu PTR

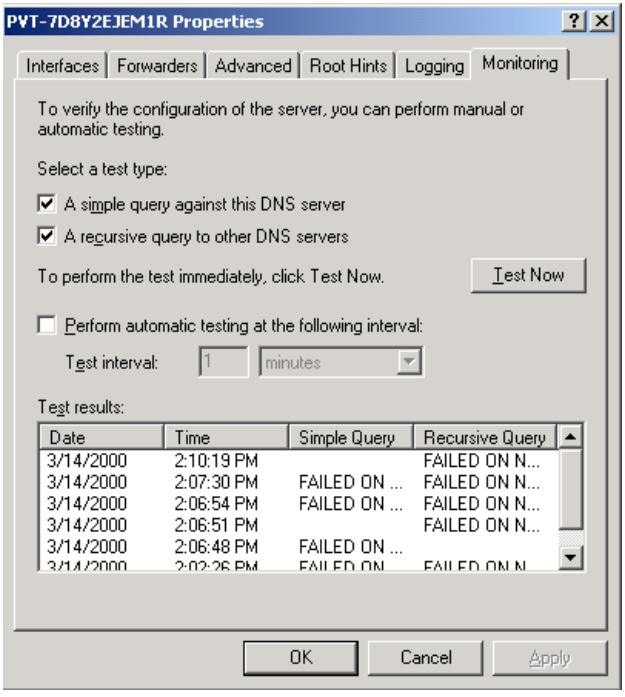


Obr. 12.13
Nová věta typu PTR se objeví v příslušné zóně

Obr. 12.14
Vlastnosti jmenného serveru (volba *Advanced*)



Obr. 12.15
Vlastnosti jmenného serveru (volba *Monitoring*)



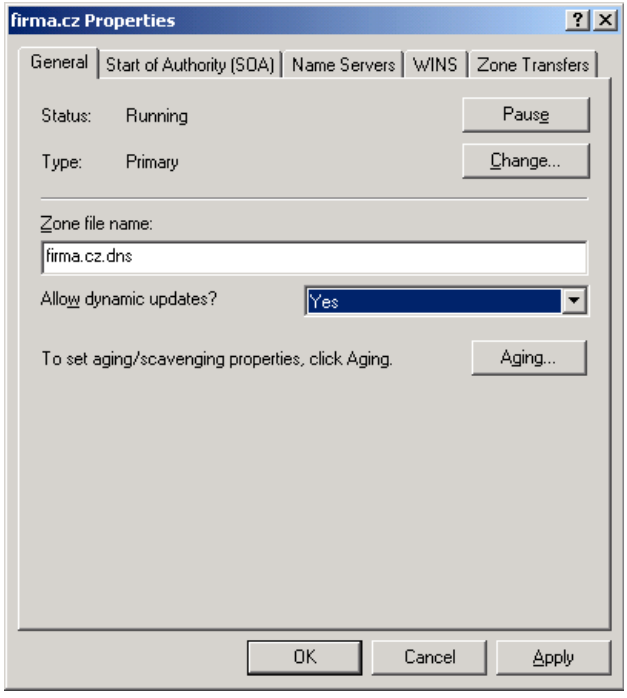
Ve volbě Advanced si povšimneme:

- „*Disable Recursion*“ – zákaz odbavovat rekurentní překlady. Rekurentní překlady jsou zakázány např. u kořenových jmenných serverů Internetu (viz RFC-2010). Zákazem rekurentních překladů zamezíme, aby náš server odbavoval dotazy od klientů (resolverů). Budou tak povoleny pouze dotazy od jmenných serverů.
- „*Name checking*“ – lze nařídit načítání zónových souborů se jmény neobsahujícími rozšířené znaky (podtržítka atp.)
- „*Load zone data on startup*“ – umístění zónových souborů: v registry Windows 2000, v souboru (obdoba UNIXu) nebo v Aktivních adresářových službách.

Na obr. 12.15 je zobrazena volba „Monitoring“ z vlastností jmenného serveru. Pomocí této volby je možné otestovat jmenný server. Pomocí volby „Perform automatic testing …“ lze zvolit časový interval, ve kterém se budou testy pravidelně opakovat.

Pomocí pravého tlačítka myši lze také zobrazit vlastnosti zóny (viz obr. 12.16). Velice důležitá je volba „General“, kde je možné zvolit, zdali má zóna podporovat Dynamický UPDATE. Bez tohoto povolení nastanou potíže při aktivaci aktivních adresářových služeb.

Obr. 12.16
Povolení dynamického UPDATE (RFC-2136)



Nyní se ještě zastavme u jednotlivých vzniklých souborů:

soubor boot:

```
;
;  Boot information written back by DNS server.
;

primary     .                           root.dns
primary     40.47.195.in-addr.arpa      40.47.195.in-addr.arpa.dns
primary     firma.cz                    firma.cz.dns
```

soubor firma.cz.dns (obdoba je i pro reverzní domény):

```
;
;  Database file firma.cz.dns for firma.cz zone.
;      Zone version:  1
;

@                       IN  SOA (
                        pvt01   ; primary DNS server
                        administrator   ; zone admin e-mail
                        1           ; serial number
                        3600        ; refresh
                        600         ; retry
                        86400       ; expire
                        3600      ) ; minimum TTL

;
;  Zone NS records
;

@                       NS      pvt01

;
;  Zone records
;

pvt01                   A       195.47.40.15
```

soubor firma.cz.dns.log:

V kap. 11.17 o inkrementálním zone transferu jsme uváděli, že jmenný server musí udržovat databázi jednotlivých verzí zónových souborů. Zde se jedná o obdobu. Navíc však díky používání vět SRV jednotlivé verze rychle přibývají, jak ukazuje naše ukázka části souboru:

```
$VERSION 67
$DELETE
pvt01                   1200    A       195.47.40.15
                                A       195.47.40.15
$ADD
                        1200    A       195.47.40.15
$DELETE
_ldap._tcp.default-first-site-name.sites 60     SRV     0 0 389 p39aah
```

```
$VERSION 44
$DELETE
_ldap._tcp.default-first-site-name.sites.dc.ms-dcs 60   SRV     0 0 389 p39aah

$VERSION 45
$DELETE
_ldap._tcp.13407fdd-17a2-11d3-8ea2-90ea275e6a4b.domains.ms-dcs 60       SRV     0 0 389 p39aah

$VERSION 46
$DELETE
6ba859b1-175d-11d3-9818-c666c4134500 60 CNAME   p39aah

$VERSION 47
$DELETE
@                       60      A       195.47.40.17

$VERSION 48
$DELETE
_kdc._tcp.dc.ms-dcs       60    SRV     0 0 88  p39aah

$VERSION 49
$DELETE
_kdc._tcp.default-first-site-name.sites.dc.ms-dcs 60    SRV     0 0 88  p39aah

$VERSION 50
$DELETE
_ldap._tcp.writable.ms-dcs 60   SRV     0 0 389 p39aah

$VERSION 51
$DELETE
_ldap._tcp.default-first-site-name.sites.writable.ms-dcs 60     SRV     0 0 389 p39aah

$VERSION 52
$DELETE
_ldap._tcp                60    SRV     0 0 389 pvt01

$VERSION 53
$DELETE
_ldap._tcp.dc.ms-dcs       60   SRV     0 0 389 pvt01
```

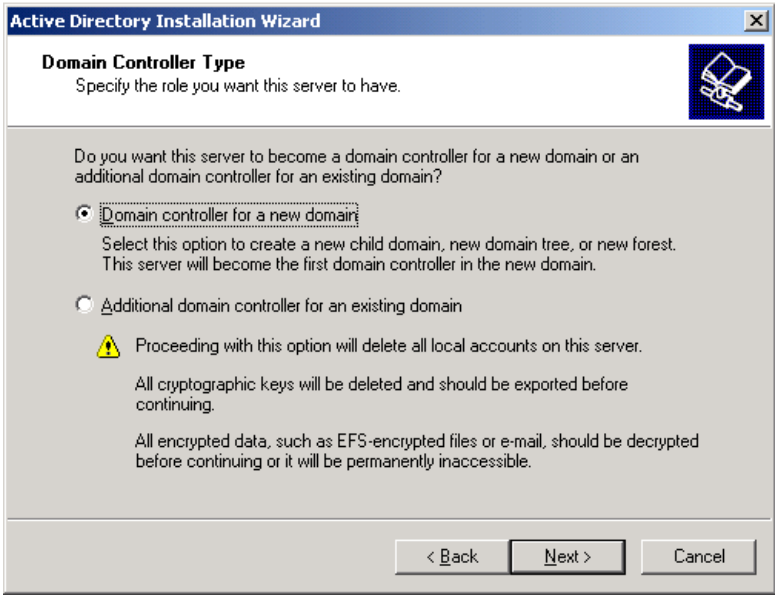
…

12.2.3 Aktivní adresářové služby

Aktivní adresářové služby lze z MS-DOS okna aktivovat pomocí programu dcpromo:

```
C:\ >dcpromo
```

Čímž se nastartuje průvodce aktivací aktivních adresářových služeb (viz obr. 12.17).

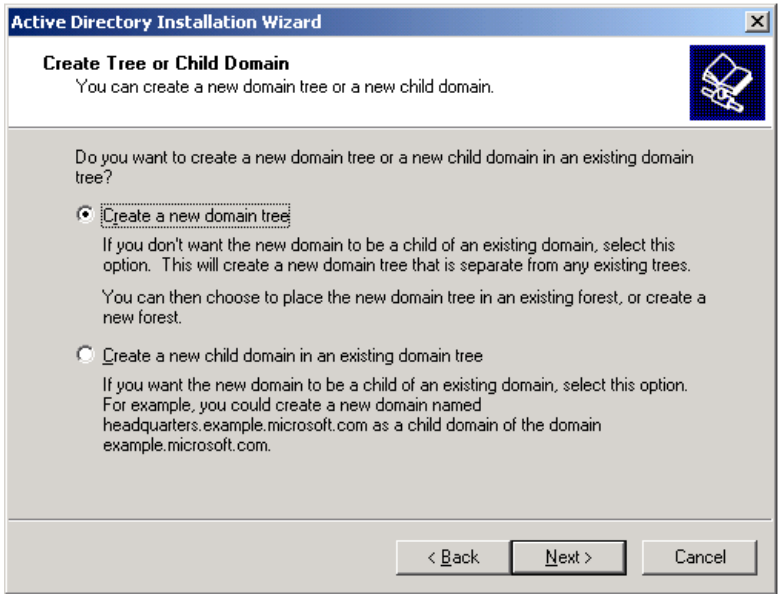


Obr. 12.17
Průvodce instalací Aktivních adresářových služeb se ptá, zdali vytváříme první DC v nové doméně nebo další DC v již existující doméně

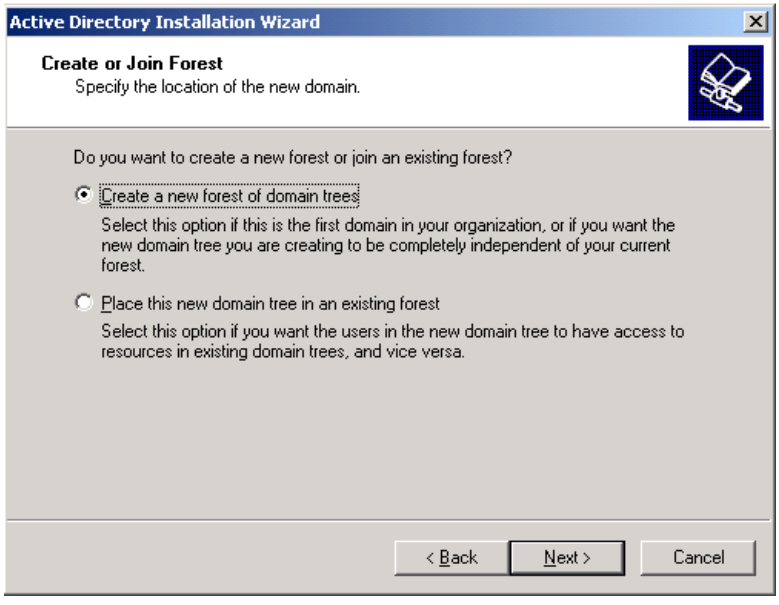
Poznamenejme, že v případě aktivní adresářové služby se jedná o doménu Windows. Tj. Windows používají slovo doména ve dvou různých významech. Jednou jako doména Windows a jednou jako doména DNS. V případě Windows 2000 je pak doména DNS integrována do domény Windows.

Na obr. 12.18 jsme tázáni, zdali se má vytvořit nový strom domén, nebo zdali vytváříme pouze doménu nižší úrovně v existujícím stromu.

Obr. 12.18
Má se vytvořit nový strom či se má vytvořit pouze nová doména v již existujícím stromu?

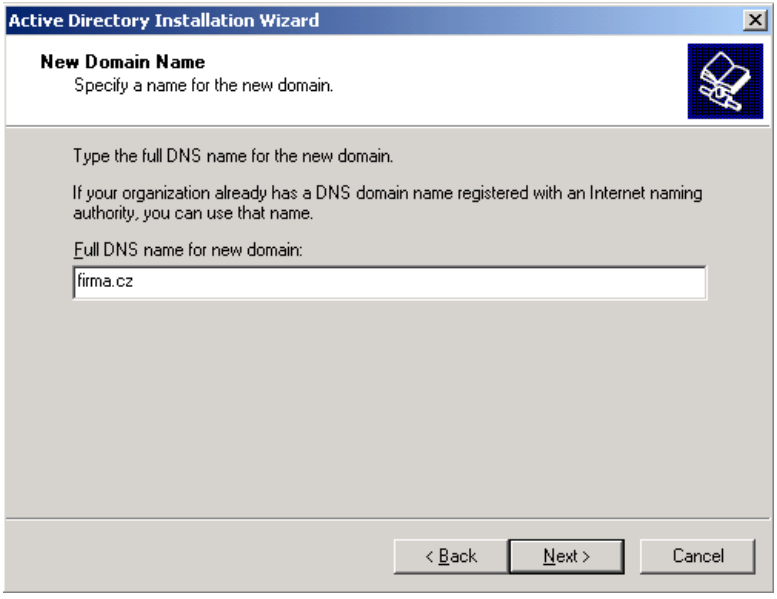


Obr. 12.19
Má se vytvořit nový les nebo se má vytvářená doména začlenit do existujícího lesa



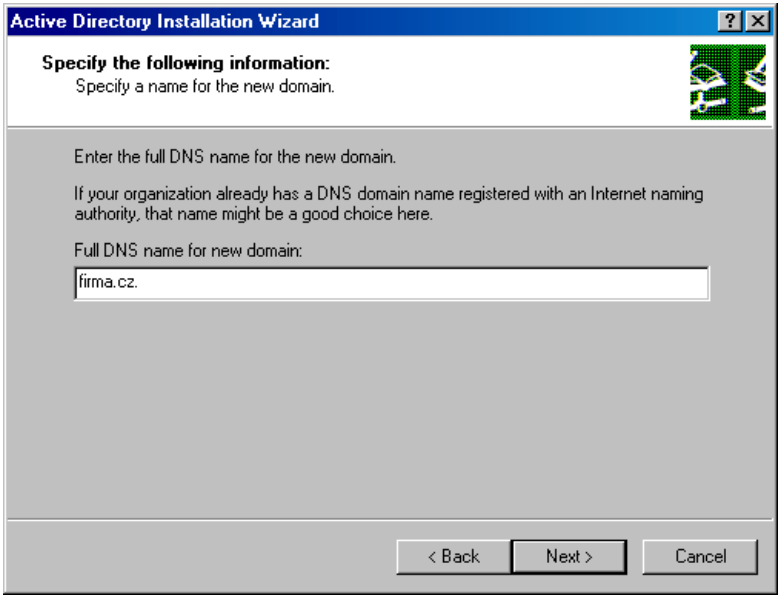
Na obr. 12.19 jsme dotazováni, zdali se má vytvořit nový les, nebo nově vytvářená doména se má zasadit do existujícího lesa.

Obr. 12.20
Okno pro zadání absolutního jména DNS domény

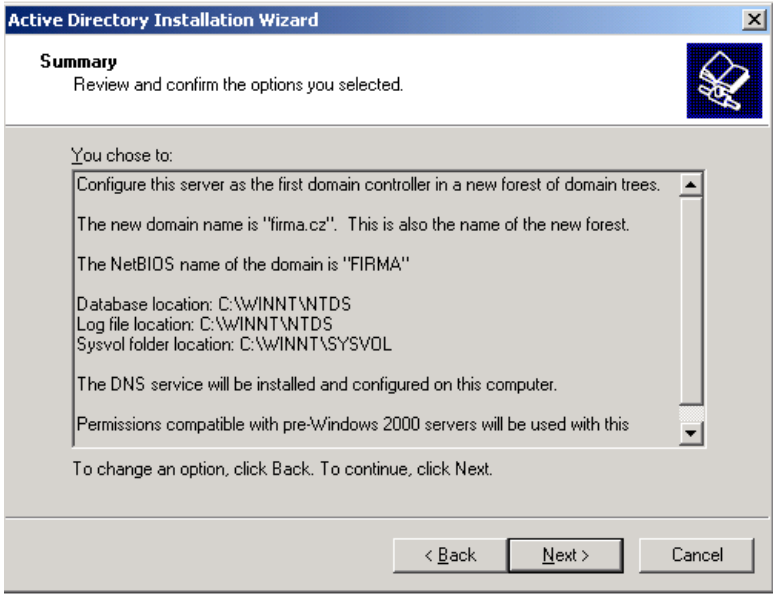


Na obr. 12.20 jsme dotazováni na jméno nově vytvářené domény.

Obr. 12.21
Okno pro zadání domény pro protokol NetBIOS



Na obr. 12.21 jsme dotázáni na název domény pro protokol NetBIOS (jméno domény Windows).



Obr. 12.22 Závěrečné okno průvodce instalací Adresářových služeb obsahující rekapitulaci nastavení

Dále jsme dotázáni na jméno souboru, ve kterém má být aktivní adresářová služba umístěna a jméno souboru pro log. Dalším oknem jsme dotázáni na jméno, pod kterým se má soubor s adresářovými službami sdílet v síti. Zbývá závěrečné okno průvodce instalací (obr. 12.23).

Po restartu Windows 2000 by aktivní adresářové služby a DNS měly začít korektně pracovat.

13

BIND 8

BIND je od verze 8 zcela přepracován. Nová verze BINDu již podporuje některé nové mechanismy DNS. Od verze 8.1 je podporováno:

- Dynamická aktualizace DNS – Dynamic update (RFC-2136).
- Upozorňování na změny v DNS – DNS notify (RFC-1996).
- Inkrementální zónový přenos – IXFR (RFC-1995).

Od verze 8.2 je podporováno:

- Negativní Caching (RFC-2308)
- DNS Clarifications (RFC-2181)
- Bezpečné DNS (RFC-2065)
- Podpora virtuálním jmeným serverům

Od verze 8.2.2 je pak podporována interoperabilita s Windows 2000.

Nejprve v přehledu uveďme hlavní změny ve vlastní implementaci BINDu 8.x oproti BINDu verze 4.x:

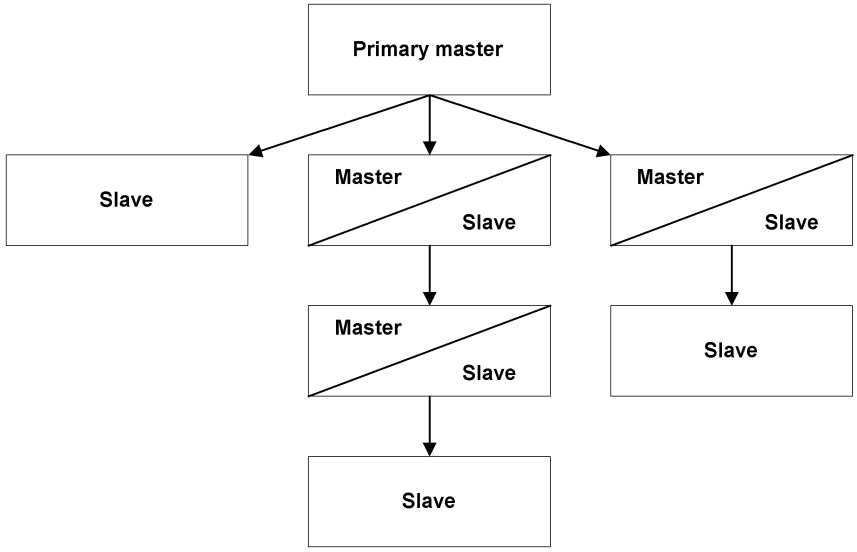
- BIND 8 používá nový konfigurační soubor – **/etc/named.conf**, konfigurační soubor má nové jméno i novou syntaxi.
- BIND 8 umožňuje podrobně **konfigurovat protokolování zpráv**.
- BIND 8 zavádí kontrolu přístupu pomocí **ACL**.
- BIND 8 používá novou architekturu **master – slave**.

13.1 Typy DNS serverů a zón

Nový BIND používá nové termíny pro označování typů zón a serverů.

Primary master

Primární hlavní server (primary master server) je hlavním zdrojem dat pro zónu. Je uveden v záznamu typu SOA a může být uveden v NS záznamu. Primární hlavní server je pro každou zónu pouze jeden. V předchozí verzi BINDu se používalo pro tento typ serveru označení primární server.



Obr. 13.1
Architektura master – slave

Master

Hlavní server (master server) je autoritativní server pro zónu. Je uveden v NS záznamu. Je zdrojem dat o zóně pro podřízené servery (slave servery). Hlavních serverů může existovat několik.

Slave

Podřízený server (slave server) je autoritativní server. Je uveden v NS záznamu. Podřízený server používá zónový přenos pro získání dat o zóně z hlavního serveru. V předchozí verzi BINDu se používalo pro tento typ serveru označení sekundární server. Podřízený server může data získaná z hlavního serveru ukládat jen na disk do souboru. Jeden server může být pro tutéž zónu hlavní i podřízený. Podřízených serverů pro danou zónu může existovat několik. V BINDu 4.x se také používal termín podřízený pro označení jmenného serveru, **význam však byl zcela jiný!** Podřízený server v BINDu 4.x bylo označení pro server, který pouze předával dotazy na jiné jmenné servery a sám nic nepřekládal.

Zóna udržovaná na podřízeném serveru se někdy označuje jako podřízená zóna.

Zóna stub

Zóna stub je v podstatě podřízená zóna, která však z master serveru přebírá pouze věty NS pro zónu, nikoli celou zónu.

Zóna hint

V zóně hint je uveden seznam kořenových jmenných serverů (neautoritativní data načítaná do paměti při startu jmenného serveru). V předchozí verzi BINDu se používalo pro tento typ zóny označení cache.

Stealth

Stealth server je tajný server. Není uveden ve větě NS. Je to autoritativní server. Získává data pro zónu pomocí zónového přenosu. Může být hlavním serverem pro zónu. Je znám pouze serverům, které mají staticky uvedenu jeho IP adresu v konfiguraci.

13.2 Konfigurační soubor

Konfigurační soubor BINDu verze 8 se zpravidla jmenuje /etc/named.conf. Tento soubor má zcela novou syntaxi.

Konfigurační soubor pro BIND 8 je tvořen příkazy a komentáři. Příkazy jsou ukončeny středníkem (;).

Konfigurační soubory používané v BIND 4.9.x je možné konvertovat do nového formátu pomocí perlovského skriptu `named-bootconf.pl`, který je součástí zdrojového kitu BIND 8.

I když konfigurační soubor má ve verzi 8 zcela odlišnou syntaxi oproti konfiguračnímu souboru ve verzi 4, tak syntaxe databází DNS (věty typu SOA, A, PTR, NS atd.) zůstává nezměněna.

13.2.1 Příkazy konfiguračního souboru

acl Definuje seznam IP adres, který se používá např. k řízení přístupu.

include Vkládá soubor.

key Definuje informace používané při autentizaci a autorizaci.

logging Definuje události, které se budou protokolovat a kde se protokolované informace uloží.

options Definuje celkovou konfiguraci serveru.

server Nastavuje některé vlastnosti pro vzdálené servery.

zone Definuje zónu.

Příkazy `logging` a `option` se mohou vyskytnout v konfiguračním souboru pouze jednou.

Na začátek uveďme pro představu jednoduchý příklad konfiguračního souboru pro BIND 8:

```
#
# prvni named.conf
#

options {
       directory „/etc/master“;
} ;
logging {
  channel protokol {
      file „log/protokol.txt“ versions 5 ;
      severity debug;
  } ;
  channel vyst {
      file „log/vy“;
  } ;
  category default {
     protokol;
  } ;
  category ncache {
    vyst;
  } ;
  category db {
```

```
    vyst;
  } ;
} ;
zone „0.0.127.in-addr.arpa“ in {
      type master;
      file „127.0.0“;
} ;
zone „.“ in {
      type hint;
      file „named.cache“;
} ;
zone „abcde.cz“ in {
      type master;
      notify yes;
      file „abcde.cz.zone“;
} ;
zone „pvtnet.cz“ in {
      type slave;
      masters { 194.149.105.18;} ;
      file „pvtnet.cz.zone“;
} ;
zone „pvt.net“ in {
      type stub;
      masters { 194.149.105.18;} ;
      file „pvt.net.zone“;

} ;
```

13.2.2 Komentáře

Komentáře v konfiguračním souboru mohou být uvedeny ve třech tvarech:

```
/* ve tvaru stejném jako v C */
// ve tvaru stejném jako v C++
# ve tvaru stejném jako v perlu
```

Komentář ve stylu C (*/) může označovat komentářový text na části řádku nebo naopak několikařádkový text. Komentář ve stylu C++ nebo Perlu naopak znamená vždy jednořádkový komentář, přesněji řečeno, za komentář se považuje text od znaku // nebo # do konce řádky.

POZOR! nepoužívejte jako komentářový znak středník, který zde má význam konce příkazu.

Příklad:

```
/* Víceřádkový komentář ve stylu C
   je uzavřen do závorek tvořených znaky
   hvězdička a lomítko */

// Víceřádkový komentář ve stylu C++ musí na každé
// řádce začínat znaky dvě lomítka
   tato řádka není komentářem a tedy způsobí chybu
//

# Komentář ve stylu perlu
# další řádka komentáře
```

13.2.3 Příkaz acl

Syntaxe:

```
acl jmeno {
   seznam IP adres
} ;
```

Popis:

Příkaz `acl` vytváří a pojmenovává seznam IP adres použitých pro kontrolu přístupu. Tento seznam musí být definován dříve, než bude kdekoli použit.

Předdefinované jsou následující seznamy acl:

- *any* – povoluje všechny uzly
- *none* – zakazuje všechny uzly
- *localhost* – povoluje IP adresu všech interface na systému
- *localnets* – povoluje všechny uzly na sítích, do kterých má systém interface

Seznam IP adres

Syntaxe:

```
Seznam_IP_adres = 1*prvek_seznamu
Seznam_IP_adres = [„!“] (ip_adresa|ip_prefix|acl_jmeno|seznam_IP_adres)
```

Popis:

Seznam IP adres je seznam prvků. Prvkem může být:

- IP adresa (dekadická čísla oddělená tečkou)
- IP prefix (v „/“ notaci)
- jméno dříve definovaného seznamu IP adres
- seznam IP adres

Prvky mohou být negovány použitím znaku „!“ před prvkem.

Při porovnávání je seznam IP adres procházen zleva. Při nalezení prvního výskytu vhodného prvku se s porovnáváním končí. Kladná porovnání povolují přístup, porovnání na negaci zakazují přístup. Pokud se IP v seznamu nenajde, má počítač přistupující z dané IP-adresy přístup zakázán.

Takto definovaný seznam IP je možné použít v parametrech allow-query, allow-transfer, allow-update a listen-on ostatních příkazů.

Příklad:

```
1.2.3/24; ! 1.2.3.13;      # 1.2.3.13 je zcela zbytečné
! 1.2.3.13; 1.2.3/24;      # správně, pouze IP 1.2.3.13 má povolen přístup, ostatní z 1.2.3.* mají přístup zakázán.
```

13.2.4 Příkaz include

Syntaxe:

```
include cesta;
```

Popis:

Příkaz `include` vloží specifikovaný soubor na místo, kde je uveden příkaz include. Příkaz include není možné použít uvnitř jiného příkazu. Příklad **chybného** použití:

```
acl int_host {  „include int_host.acl“} ;
```

Příklad:

```
include „/etc/security/keys.bind“;
include „/etc/acls.bind“;
```

13.2.5 Příkaz key

Syntaxe:

```
key key_id {
  algorithm algorithm_id;

  secret secret_string;
} ;
```

Popis:

Příkaz *key* definuje šifrovací klíče. Tyto klíče se používají k definici autentizační metody při přístupu k jiným jmenným serverům v příkazu *server*. Příkazem key musí být šifrovací klíče definovány dříve než budou použity v příkazu *server*. Tj. příkaz key musí být v konfiguračním souboru uveden dříve než příkaz `server`.

Algorithm_id je řetězec, který specifikuje autentizační algoritmus.

Secret_string je heslo použité algoritmem.

Tento příkaz zatím není implementován, kontroluje se pouze jeho syntaxe.

13.2.6 Příkaz zone

Syntaxe:

Příkaz zone používá tři typy syntaxe:

```
zone domain_name  [ ( in | hs | hesiod | chaos ) ] {
  type master;
  file path_name;
  [ check-names ( warn | fail | ignore ); ]
  [ allow-update {  address_match_list } ; ]
  [ allow-query {  address_match_list } ; ]
```

```
  [ allow-transfer {  address_match_list } ; ]
  [ notify yes_or_no; ]
  [ also-notify {  ip_addr; [ ip_addr; ... ] } ;
} ;


zone domain_name [ ( in | hs | hesiod | chaos ) ] {
  type ( slave | stub );
  [ file path_name; ]
  masters {  ip_addr; [ ip_addr; ... ] } ;
  [ check-names ( warn | fail | ignore ); ]
  [ allow-update {  address_match_list } ; ]
  [ allow-query {  address_match_list } ; ]
  [ allow-transfer {  address_match_list } ; ]
  [ max-transfer-time-in number; ]
  [ notify yes_or_no; ]
  [ also-notify {  ip_addr; [ ip_addr; ... ] } ;
} ;


zone . [ ( in | hs | hesiod | chaos ) ] {
  type hint;
  file path_name;

  [ check-names ( warn | fail | ignore ); ]
} ;
```

Popis:

Příkaz zone definuje jednotlivé zóny.

Type – Typ zóny:

master

Hlavní zdroj dat pro zónu. (V předchozí verzi BINDu šlo o primární zónu).

slave

Podřízená zóna je kopie hlavní zóny. (V předchozí verzi BINDu šlo o sekundární zónu). Seznam masters specifikuje jednu nebo více IP adres, které slave kontaktuje při aktualizování zóny. Pokud je uveden parametr file, pak se zapíše kopie do souboru. Použití file je doporučeno.

stub

Zóna stub je vlastně podřízená zóna, která však z hlavního serveru přebírá pouze NS věty pro zónu, nikoli celou zónu.

hint

V zóně hint je uveden seznam kořenových jmenných serverů. Při startu serveru použije zónu hint jmenného serveru k vyhledání kořenových serverů.

Jméno zóny může být následováno třídou. Pokud třída není uvedena, použije se `in` (Internet).

Parametry:

check-names

Server může kontrolovat doménová jména podle kontextu, např. doménové jméno použité jako jméno uzlu může být kontrolováno na platná jména uzlů, tj. aby jméno neobsahovalo např. znaky podtržítko, hvězdička atd. Použitelné jsou tři metody:

`ignore` – neprovádí se žádná kontrola,
`warn` – jméno je kontrolováno, chybná jména jsou zaznamenána, ale zpracování pokračuje dále,
`fail` – jména jsou kontrolována, chybná jména jsou zaznamenána, avšak zpracování je ukončeno.

allow-query

Definuje, které uzly mohou položit běžný dotaz. Pokud není uvedeno, je implicitně položení běžného dotazu povoleno pro všechny uzly. `allow-query` může být uvedeno také v příkazu `option`, pak má jeho uvedení v příkazu `zone` přednost před nastavením v příkazu `option`.

allow-update

Definuje uzly, které mají povoleno provést dynamický update serveru. Implicitně je dynamický update zakázán ze všech uzlů.

allow-transfer

Definuje, které uzly mají povolen zónový přenos ze serveru. Pokud není uvedeno, je implicitně zónový přenos povolen ze všech uzlů. `allow-transfer` může být uvedeno také v příkazu `option`, pak má jeho uvedení v příkazu `zone` přednost před nastavením v příkazu `option`.

max-transfer-time-in

Počet minut po který může trvat zone transfer. Pokud trvá déle, bude ukončen. Implicitní hodnota je 120 minut.

notify

Je-li nastaveno na yes, pak je při změně zóny, pro kterou je server autoritou, vysílána serverem zpráva notify. Implicitní hodnota je yes. Podřízený server, který obdrží zprávu notify a zprávě rozumí, bude kontaktovat hlavní server pro danou zónu a pokud zjistí, že je potřeba zónový přenos, okamžitě jej provede. Použití NOTIFY urychluje konvergenci mezi hlavním serverem a podřízenými servery. Volba může být též uvedena v příkazu `zone`, pak má přednost před nastavením v příkazu `option`.

also-notify

Má význam pouze pokud je aktivní *notify* pro zónu. Uzly, které obdrží DNS NOTIFY pro tuto zónu, jsou všechny podřízené jmenné servery pro zónu a dále IP specifikované v *also-notify*. *Also-notify* nemá význam pro stub zóny. Implicitní je prázdný seznam.

13.2.7 Příkaz server

Syntaxe:

```
server ip_addr {
  [ bogus yes_or_no; ]
  [ transfers number; ]
```

```
  [ transfer-format ( one-answer | many-answers ); ]

  [ keys {  key_id [key_id ... ] } ; ]
} ;
```

Popis:

Příkaz server definuje charakteristiky spojené se vzdáleným jmenným serverem.

bogus

Pokud zjistíte, že server poskytuje chybná data, označte ho jako **bogus**. Tím zabráníte dalším dotazům na tento server. Implicitní hodnota *bogus* je *no*.

transfer-format

Server podporuje dvě metody zone transferu:

`one-answer` – Jedna DNS zpráva pro přenos jednoho záznamu.

`many-answers` – Do jedné DNS zprávy zabalí tolik záznamů, kolik je možné. Metoda `many-answers` je efektivnější, je však podporovaná pouze BINDem 8 a pomocí záplat ošetřenou verzí 4.9.5. Implicitní hodnota je one-answer.

Parametry **transfers** a **keys** jsou rezervované pro budoucí použití, kontroluje se pouze syntaxe.

13.2.8 Příkaz logging

Syntaxe:

```
logging {
  [ channel channel_name {
    ( file path_name
       [ versions ( number | unlimited ) ]
       [ size size_spec ]
     | syslog ( kern | user | mail | daemon | auth | syslog | lpr |
                news | uucp | cron | authpriv | ftp |
                local0 | local1 | local2 | local3 |
                local4 | local5 | local6 | local7 )
     | null );

    [ severity ( critical | error | warning | notice |
                 info  | debug [ level ] | dynamic ); ]
    [ print-category yes_or_no; ]
    [ print-severity yes_or_no; ]
    [ print-time yes_or_no; ]
  } ; ]

  [ category category_name {
    channel_name; [ channel_name; ... ]
  } ; ]

  ...
} ;
```

Popis:

Příkaz `logging` konfiguruje přihlašování. Definuje, jaké typy událostí se budou zaznamenávat, v jakém formátu a kam se budou zaznamenávat. Uplatní se pouze první příkaz logging.

Příkaz logging používá dva pojmy:

- **channel** – kanál (kam se bude zapisovat)
- **category** – kategorie (co se bude zapisovat)

Jeden příkaz `logging` může definovat libovolné množství kanálů a kategorií. Pokud není příkaz `logging` uveden, použije se implicitní nastavení.

channel

Každý kanál je označen jménem. Kanál je definován třemi prvky:

- **Kam budou zprávy zapisovány** – do souboru (file), do systémového logu (syslog), nikam (null).

  null – zprávy se nikam nezapisují

  file – zprávy se zapisují do souboru. Je možné definovat maximální velikost log souboru (size), počet verzí, které se mají udržovat (versions).

  size – pokud soubor dosáhne definované velikosti, named již další zprávy nezapisuje, dokud nebude soubor znovu otevřený. Přesažení velikosti nezpůsobí automatické znovuotevření souboru. Implicitně není velikost souboru omezena.

  version – počet udržovaných verzí. Nejnovější je verze 0. Implicitně se verzování neuplatní. Unlimited = 99 verzí.

  syslog – logování bude předáváno systému syslog, který používá konfigurační soubor syslog.conf (blíže viz příslušný manuál systému).

- **Závažnost chyby** – implicitní je závažnost info.

  severity Zprávy s nižší závažností než severity nebudou tímto kanálem akceptovány. Při použití systémového žurnálu pak je hodnota severity vyhodnocována dle konfigurace v souboru syslog.conf, má-li se událost zaznamenat.

- **Zapisování/nezapisování dodatečných informací** – časová značka, jméno kategorie, závažnost chyby. Implicitně se tyto informace nezapisují.

  print-time pokud je on, zapisuje se datum a čas do logu.

  print-category zapisuje se kategorie zprávy.

  print-severity zapisuje se závažnost zprávy. Print volby je možné použít v libovolné kombinaci.

Jsou předdefinovány 4 kanály:

```
channel default_syslog { # Zprávy se zapisují do systémového logu
  severity info;         # Zapisují se zprávy o závažnosti info a vyšší
  syslog daemon;         # Syslog zaznamenává jaká část systému
                         # generuje událost (Kernet,mail,...).
                         # Tento kanál bude zaznamenávat události
                         # jakoby je generovali demoni.
 } ;
```

```
channel default_debug {
    file „named.run“;       # Zprávy se zapisují do souboru named.run
                            # v pracovním adresáři
    severity dynamic;       # Zapisují se zprávy podle aktuálně
                            # nastavené úrovně debug
} ;

channel default_stderr {    # Zprávy se zapisují na stderr
    file „<stderr>“;        # jde pouze o ilustraci
    severity info;
} ;

channel null {
    null;                   # Všechny zrávy poslané na tento
                            # kanál se zahodí
} ;
```

Vedle těchto čtyř předdefinovaných kanálů si může správce DNS serveru definovat kanály další. Jakmile je kanál definován, není možné jej předefinovat. Nelze tedy změnit definici kanálu přímo, ale můžete modifikovat implicitní log tak, že změníte kategorie (na stejné kanály můžete posílat jiné zprávy.)

category

Všechny zprávy generované programem named jsou rozděleny podle typu do několika skupin – kategorií. Každá kategorie je označena jménem.

Použitelné kategorie:

default – všechny zprávy, i ty, které nejsou rozděleny do dalších kategorií. Pokud nespecifikujete pro některou z dále uvedených kategorií kanál, posílají se zprávy daného typu na stejný kanál jako zprávy kategorie default.

config – Závažné chyby v konfiguračním souboru.

parser – Chyby nižší úrovně v konfiguračním souboru.

gueries – Krátká zpráva pro každý obdržený dotaz.

lame-servers – Zprávy typu Lame server on.

statistics – Statistiky.

panic – Named nemůže dále běžet z důvodu interního problému.

update – Dynamická aktualizace.

ncache – Negativní cacheování.

xfer-in – Příchozí zónové přenosy.

xfer-out – Odchozí zónové přenosy.

db – Všechny databázové operace.

eventlib – Ladění informací od systému událostí `category eventlib { default_debug; } ;`. Může být uveden pouze jeden kanál a musí to být soubor.

packet – dump příchozích i odchozích paketů. Může být uveden pouze jeden kanál a musí to být soubor. Pokud jej nedefinujete, použije se tato: `category packet { default_debug; } ;`

notify – Notify protokol.

cname – Zprávy typu ... points to a CNAME.

security – Povolený/nepovolený dotaz.
os – Problémy OS.
insist – Chyby při kontrole interní konzistence.
maintenance – Periodická údržba.
load – Zprávy o načtení zóny.
response-checks – Zprávy z kontroly odpovědí. zprávy typu „invalid RR type...“, „unrelated additional info...“ atd.

Jednotlivé kategotie zpráv je možné posílat na různé kanály. Pokud nespecifikujete seznam kanálů pro kategorii, pak jsou zprávy z dané kategorie posílány na stejný kanál jako zprávy kategorie default. Pokud nespecifikujete kategorii default, použije se tato definice:

```
category default {  default_syslog; default_debug; } ;
```

Implicitně jsou tedy všechny zprávy generované programem named posílány na kanály: `default_syslog; default_debug;`, tzn do systémového žurnálu a do souboru named.run v pracovním adresáři.

Příklad:

Definuji speciální kanál pro logování zpráv závažnosti info a vyšší kategorie security

```
channel my_security_channel {
    file „my_security_file“;
    severity info;
} ;
category security {  my_security_channel; default_syslog; default_debug; } ;
```

Chcete-li zprávy zahazovat, použijte kanál null:

```
category lame-servers {  null; } ;
category cname {  null; } ;
```

13.2.9 Příkaz option

Syntaxe:

```
options {
  [ directory path_name; ]
  [ named-xfer path_name; ]
  [ dump-file path_name; ]
  [ pid-file path_name; ]
  [ statistics-file path_name; ]
  [ auth-nxdomain yes_or_no; ]
  [ fake-iquery yes_or_no; ]
  [ fetch-glue yes_or_no; ]
  [ multiple-cnames yes_or_no; ]
  [ notify yes_or_no; ]
  [ recursion yes_or_no; ]
  [ forward ( only | first ); ]
  [ forwarders {  [ in_addr ; [ in_addr ; ... ] ] } ; ]
  [ check-names ( master | slave | response ) ( warn | fail | ignore); ]
  [ allow-query {  address_match_list } ; ]
```

```
  [ allow-transfer {  address_match_list } ; ]
  [ listen-on [ port ip_port ] {  address_match_list } ; ]
  [ query-source [ address ( ip_addr | * ) ] [ port ( ip_port | * ) ] ; ]
  [ max-transfer-time-in number; ]
  [ transfer-format ( one-answer | many-answers ); ]
  [ transfers-in  number; ]
  [ transfers-out number; ]
  [ transfers-per-ns number; ]
  [ coresize size_spec ; ]
  [ datasize size_spec ; ]
  [ files size_spec ; ]
  [ stacksize size_spec ; ]
  [ clean-interval number; ]
  [ interface-interval number; ]
  [ statistics-interval number; ]
  [ topology {  address_match_list } ; ]
} ;
```

Popis:

Příkaz `option` nastavuje globální volby pro program named. Příkaz může být v konfiguračním souboru uveden pouze jednou. Pokud není uveden, použijí se výchozí nastavení.

13.2.10 Parametry

13.2.10.1 Specifikace souborů

directory

Pracovní adresář serveru. Adresář musí být uveden ve tvaru absolutní cesty. Každá relativní adresářová cesta uvedená v konfiguračním souboru je vyhodnocována vzhledem k pracovnímu adresáři serveru. V tomto adresáři je implicitně umísťována většina výstupních souborů serveru. Pokud adresář není uveden, pak je za implicitní považován adresář, ze kterého byl server startován.

named-xfer

Cesta k programu named-xfer, který server používá pro příchozí zónový přenos. Pokud není cesta uvedena, je dána systémem.

dump-file

Cesta k souboru, kam server zapíše dump, pokud obdrží signál SIGINT. Pokud není uvedena, je implicitní „`named_dump.db`“.

pid-file

Cesta k souboru, kam server zapisuje své ID procesu. Pokud není uvedeno, pak to obvykle bývá /etc/named.pid.

statistics-file

Cesta k souboru, do kterého server zapisuje statistiky po obdržení signálu SIGILL. Pokud není uvedena, je implicitně použito `named.stats`.

13.2.10.2 Parametry typu boolean

auth-nxdomain

Je-li nastaveno na hodnotu yes, pak je AA bit vždy nastaven v odpovědi NXDOMAIN (negativní odpovědi), i když server není autoritou. Implicitní hodnota je yes.

fake-iquery

Je-li nastaveno na hodnotu yes, pak server simuluje DNS dotaz typu IQUERY. Implicitní hodnota je no.

fetch-glue

Implicitní hodnota je yes. Pokud je yes, server přidává přilepené (*glue*) věty. Nastavení no je možné použít ve spojení s `recursion no`.

multiple-cnames

Je-li nastaveno na yes, pak je povoleno více CNAME vět pro doménové jméno. Implicitní hodnota je no. Povolení více vět CNAME je proti standardu a není doporučováno. Předchozí BIND umožňoval více CNAME.

notify

Je-li nastaveno na yes, pak je při změně zóny, pro kterou je server autoritou, vysílána serverem notify zpráva. Implicitní hodnota je yes. Podřízený server, který obdrží notify zprávu a zprávě rozumí, bude kontaktovat hlavní server pro danou zónu a pokud zjistí, že je potřeba zónový přenos, okamžitě jej provede. Použití NOTIFY urychluje konvergenci mezi hlavním serverem a podřízenými servery. Volba může být též uvedena v příkazu `zone`, pak má přednost před nastavením v příkazu `option`.

recursion

Je-li nastaveno na yes a DNS dotaz vyžaduje rekurzi, pak se bude server snažit tento dotaz vyřešit. Pokud je nastaveno na no a server nezná odpověď, vrátí klientovi pouze odkaz. Implicitní hodnota je yes.

Forwarding

Forwarding je možné výhodně využít k naplnění a využívání velké paměti cache na několika serverech. Tím se snižuje provoz na linkách k externím jmenným serverům. Forwarding nastane pouze pokud pro dotaz není server autoritou a nemá odpověď v paměti.

forward

Tento parametr má smysl pouze pokud není seznam forwarderů prázdný. Implicitní hodnota je `first`. Hodnota first znamená, že nejprve server kontaktuje k vyřešení dotazu forwarder server a teprve pokud se nepodaří forwarderovi dotaz vyřešit, pokouší se o to server sám. Hodnota `only` znamená, že server k vyřešení dotazu kontaktuje forwardera a sám se nesnaží dotaz řešit.

forwarders

Uvádí IP adresy serverů pro forwarding. Implicitně je seznam prázdný (neprovádí se forwarding).

13.2.10.3 Kontrola jmen

check-names

Server může kontrolovat doménová jména podle kontextu, např: doménové jméno použité jako jméno uzlu může být kontrolováno na RFC definující platná jména uzlů.

Použitelné jsou tři metody:

`ignore` – neprovádí se žádná kontrola,

`warn` – jméno je kontrolováno, chybná jména jsou zaznamenávána, ale zpracování pokračuje dále,

`fail` – jména jsou kontrolována, chybná jména jsou zaznamenávána a chybná data jsou zahozena. Server může kontrolovat jména ve třech oblastech: v souborech hlavní zóny, v souborech podřízené zóny a v odpovědi na dotaz, který položil. Pokud je použito `check-names response fail` a server by jako odpověď měl klientovi odeslat chybné jméno, pošle odpověď REFUSED. Implicitní hodnoty jsou:

```
check-names master fail;
check-names slave warn;
check-named response ignore;
```

`check-names` může být uvedeno také v příkazu `zone`, pak má přednost před nastavením v příkazu `option` a neuvádí se oblast.

Access Control

Přístup k serveru je možné omezit na jisté IP adresy.

allow-query

Definuje, které uzly mohou položit běžný dotaz. Pokud není uvedeno, je implicitně položení běžného dotazu povoleno pro všechny uzly. `allow-query` může být uvedeno také v příkazu `zone`, pak má přednost před nastavením v příkazu `option`.

allow-transfer

Definuje, které uzly mají povolen zone transfer ze serveru. Pokud není uvedeno, je implicitně zónový přenos povolen ze všech uzlů. `allow-transfer` může být uvedeno také v příkazu `zone`, pak má přednost před nastavením v příkazu `option`.

13.2.10.4 Síťová rozhraní

listen-on

V parametru je možné uvést interface a porty, ze kterých bude server akceptovat a zodpovídat dotazy. Parametr je možno uvést vícekrát. Pokud není parametr `listen-on` uveden, pak server poslouchá na portu 53 na všech rozhraních.

Příklad:

```
listen-on {  194.149.100.33; } ;
listen-on port 2323 {  !195.47.127.44; 195.47/16 } ;
```

13.2.10.5 Dotazy

query-source

Pokud server neumí vyřešit dotaz, požádá další jmenné servery. Parametr `query-source` definuje adresy a porty pro tyto dotazy. Pokud je `address` vynechána nebo je *, bude použita libovolná IP-adresa (INADDR_ANY). Pokud je vynechán port nebo je *, bude použit náhodný neprivilegovaný port. Implicitní hodnota je:

```
query-source address * port *;
```

`query-source` se uplatní pouze u UDP dotazů, TCP dotazy vždy používají libovolnou IP a libovolný neprivilegovaný port.

13.2.10.6 Zónový přenos

max-transfer-time-in

Počet minut, po který může trvat zónový přenos. Pokud trvá déle, bude ukončen. Implicitní hodnota je 120 minut.

transfer-format

Server podporuje dvě metody zónového přenosu:

`one-answer` – Jedna DNS zpráva pro přenos jednoho RR záznamu.

`many-answers` – Do jedné DNS zprávy zabalí tolik RR záznamů, kolik je možné. Metoda `many-answers` je efektivnější, je však podporovaná pouze BINDem 8 a pomocí záplat ošetřenou verzí 4.9.5. Implicitní hodnota je one-answer.

transfers-in

Maximální počet současně probíhajících příchozích zone transferů. Implicitní hodnota je 10.

transfers-out

Tento parametr bude použit v budoucnu pro omezení počtu současně probíhajících odchozích zónových přenosů. Pouze se kontroluje syntaxe.

transfers-per-ns

Maximální počet příchozích zónových přenosů, které mohou být současně prováděny z daného vzdáleného jmenného serveru. Implicitní hodnota je 2.

Resource limits

Je možné omezit množství serverem používaných systémových zdrojů. Některé OS nepodporují některá omezení. Hodnoty mohou být např: `1G`, `unlimited` (neomezeno), `default` (limit platný při startu serveru).

coresize

Maximální velikost dump souboru. Implicitní hodnota je `default`.

datasize

Maximální velikost paměti, kterou server může používat. Implicitní hodnota je `default`.

files

Maximální počet souborů, které má server současně otevřeny. Implicitní hodnota je `unlimited`.

stacksize

Maximální množství zásobníkové paměti, kterou server může použít. Implicitní hodnota je `default`.

13.2.10.7 Časové intervaly

clean-interval

Počet minut n. Server bude odstraňovat neplatné záznamy z cache každých n minut. Implicitní hodnota je 60 minut. Pokud je nastaveno na 0, nikdy se neplatné věty neodstraňují.

interface-interval

Počet minut n. Server bude prohlížet síťové interface každých n minut. Pokud je nastaveno na 0, prohlížení síťových karet se provádí pouze při zavádění konfigurace. Po provedení prohlížení startuje naslouchač na nových síťových kartách.

statics-interval

Počet minut n. Statistiky o hlavním serveru budou zaznamenány každých n minut. Implicitní hodnota je 60 min. Pokud je nastavena hodnota 0, statistiky se nezaznamenávají.

14

Nástroje pro ladění DNS a běžné chyby v konfiguraci DNS

Po konfiguraci jmenného serveru a jeho spuštění musíme zkontrolovat, zda náš jmenný server pracuje korektně. Chyby DNS jsou velice nepříjemné. Při chybě DNS se někdy aplikace ani nerozběhnou, častěji se celý systém jeví jako výrazně pomalý. Zejména při konfiguraci firewallu, pokud zjistíme dlouhé odezvy firewallu, pak s největší pravděpodobností je na vině chybně pracující DNS.

Existují i informativní RFC zabývající se problémy DNS. Např. častým chybám DNS se věnuje RFC-1537 a prostředkům pro ladění DNS se věnuje RFC-1713.

Při kontrole konfigurace můžeme postupovat dvěma způsoby.

1. První způsob kontroly spočívá v tom, že se vžijeme do role resolveru a budeme na náš DNS server posílat DNS dotazy tak, jak to dělá resolver. Budeme tedy zkoušet, zda jmenný server odpoví na náš dotaz tak, jak očekáváme.
2. Druhý způsob kontroly je komplexní kontrola (ladění DNS) pomocí programu, který zná pravidla DNS a kontroluje splnění těchto pravidel u domén na našem jmenném serveru. Výsledkem kontroly je seznam chyb, které se v konfiguraci konkrétní domény vyskytují.

Pokud máte podezření na chybně pracující DNS, tak vždy jako předstupeň otestujte dostupnost Internetu. Pokud sedíte u PC, pak ověřte:

1. Zdali pracuje korektně TCP/IP na PC příkazem:

   ```
   ping  127.0.0.1
   ```

2. Zdali máte spojení na směrovač na LAN:

   `ping  IP-adresa_směrovače` (nikoliv jméno směrovače!)

3. Zdali máte spojení na místní jmenný server:

   `ping  IP-adresa_name_serveru` (nikoliv jméno jmenného serveru!)

4. Zdali máte spojení do světa:

   ```
   ping   IP-adresa_ve_světě
   ```

5. Pokud máte možnost, pak obdobně ověřte dostupnost Internetu i ze samotného jmenného serveru.

14.1 Program nslookup

Nejčastěji používaným programem pro kontrolu DNS je program nslookup. Tento program má jednu velkou výhodu. Je dnes totiž součástí balíku TCP/IP na Unixu i Windows a nemusíme jej tedy nikde shánět a kompilovat.

Programem nslookup posíláme DNS dotazy na DNS server a kontrolujeme, zda DNS server odpovídá správně. Pomocí programu nslookup můžeme vystupovat v roli resolveru a požadovat po jmenném serveru konečnou odpověď na náš dotaz. Nebo můžeme pomocí programu nslookup simulovat chování jmenného serveru, který komunikuje s jiným jmenným serverem (tj. požadovat jen částečné odpovědi). Záleží na tom, co chceme pomocí programu nslookup testovat.

Program nslookup posílá DNS dotazy implicitně na jmenný server, který je na systému nastavený jako resolver. V OS Unix tedy posílá dotazy na jmenný server uvedený v souboru /etc/resolv.conf.

Program nslookup spustíme v interaktivním režimu příkazem nslookup, výsledkem spuštění programu je např. prompt:

```
Default Server:  cbu.pvtnet.cz
Address:  194.149.105.18
>
```

Odpdověď nám oznamuje, že server cbu.pvtnet.cz je na našem cvičném systému definován v konfiguraci resolveru jako implicitní jmenný server. Tento jmenný server má IP adresu 194.149.105.18. Na výzvu (*prompt*) > zadáme svůj dotaz. Ptát se můžeme např. na IP adresu nebo jméno nějakého uzlu.

Zadám-li na prompt jméno uzlu např. www.pvt.cz, pokusí se jmenný server cbu.pvtnet.cz zjistit IP adresu tohoto uzlu.

Dotaz:

```
> www.pvt.cz
Server:  cbu.pvtnet.cz
Address:  194.149.105.18
```

Odpověď:

```
Name:    www.pvt.cz
Address:  194.149.104.206
>
```

Zadám-li na prompt IP adresu např. 194.149.104.206, pokusí se default server zjistit doménové jméno uzlu s touto IP adresou.

Dotaz:

```
> 194.149.104.206
Server:  cbu.pvtnet.cz
Address:  194.149.105.18
```

Odpověď:

```
Name:    www.pvt.cz
Address:  194.149.104.206
>
```

Jak je zřejmé z uvedených příkladů, program nslookup implicitně hledá v DNS odpovídající záznam A nebo záznam PTR. Programem nslookup se však můžeme zeptat jmenného serveru na libovolný záznam RR. Typ záznamu, který nás zajímá definujeme v programu nslookup pomocí příkazu

`set querytype=typ_záznamu`, který je možné zkrátit na `set q=typ_záznamu`.

Použití si opět ukážeme na příkladu. Tentokrát nás bude zajímat seznam serverů, na které je směrována pošta pro doménu pvt.cz. Již víme, že směrování pošty je definováno větami MX v zónovém souboru příslušné domény. Zajímají nás tedy všechny věty typu MX. Nastavíme tedy požadovaný typ vět MX takto:

Dotaz:

```
>set q=mx
> pvt.cz.
Server:  cbu.pvtnet.cz
Address:  194.149.105.18
```

Odpověď:

```
pvt.cz  preference = 20, mail exchanger = info.pvt.net
pvt.cz  preference = 5, mail exchanger = fw.pvt.cz
pvt.cz  preference = 10, mail exchanger = mh.pvt.cz
pvt.cz  nameserver = ns.pvt.net
pvt.cz  nameserver = ns1.pvt.net
pvt.cz  nameserver = snmp0.pvt.net
pvt.cz  nameserver = ns0.pipex.net
pvt.cz  nameserver = ns1.pipex.net
info.pvt.net    internet address = 194.149.104.203
fw.pvt.cz       internet address = 194.149.101.194
mh.pvt.cz       internet address = 194.149.105.36
ns.pvt.net      internet address = 194.149.105.18
ns1.pvt.net     internet address = 194.149.103.201
snmp0.pvt.net   internet address = 194.149.103.34
ns0.pipex.net   internet address = 158.43.128.8
ns0.pipex.net   internet address = 158.43.128.103
ns1.pipex.net   internet address = 158.43.192.40
ns1.pipex.net   internet address = 158.43.192.7
>
```

Pro doménu pvt.cz je pošta směrována na uzel fw.pvt.cz a na dva záložní uzly info.pvt.net a mh.pvt.cz.

Všimněte si, že program nslookup vypisuje nejen vlastní odpověď ale i ostatní informace z DNS paketu, který od serveru obdržel. Kromě vlastní odpovědi navíc vidíme i autoritativní servery pro doménu pvt.cz a IP adresy všech serverů v odpovědi. Tyto doplňkové informace nebudu v dalších příkladech z důvodu přehlednosti uvádět.

Další případ, kdy se nslookup často používá, je zjištění autoritativních serverů pro doménu. Zajímají nás tentokrát jména jmenných serverů, které se o danou doménu starají. Požadovanou informaci jednoduše získáme, když nastavíme typ věty na NS:

Dotaz:

```
> set q=ns
> pvt.cz
Server:  cbu.pvtnet.cz
Address:  194.149.105.18
```

Odpověď:

```
pvt.cz  nameserver = ns1.pvt.net
pvt.cz  nameserver = snmp0.pvt.net
pvt.cz  nameserver = ns0.pipex.net
pvt.cz  nameserver = ns1.pipex.net
pvt.cz  nameserver = ns.pvt.net
```

Doména pvt.cz je delegována na 5 autoritativních jmenných serverů.

Cvičení: Zjistěte autoritativní jmenné servery pro doménu fj.

Cvičení: Zjistěte kořenové jmenné servery Internetu (tj. autoritativní jmenné servery pro tečku).

Pokud např. nevíme, zda je doménové jméno kanonickým jménem nebo aliasem, můžeme využít nastavení `set q=any` a zjistit tak všechny věty vztahující se k danému doménovému jménu.

Dotaz:

```
> set q=any
> info.pvt.net
Server:  localhost
Address:  127.0.0.1
```

Odpověď:

```
info.pvt.net    CPU = AlphaServer 100   OS = OSF/1
info.pvt.net    text = "e-mail: dostalek@pvt.net"
info.pvt.net    internet address = 194.149.104.203
```

V našem případě je doménové jméno info.pvt.net definováno ve 3 větách, ve větě typu A, typu TXT a ve větě typu HINFO.

14.1.1 Ladící režim

Při hledání chyby v konfiguraci nám často nestačí informace běžně vypisované programem nslookup a chceme vědět více. Použijeme tedy ladící režim programu. U programu nslookup je možné nastavit dvě úrovně ladícího režimu: režim debug a režim d2. Ladící úrovně se nastavují příkazem set.

14.1.2 Ladící úroveň debug

Ladící úroveň debug vypisuje detailní informace z přicházejících DNS paketů.

Nastavení ladící úrovně debug:

```
> set debug
```


Vezmete-li si k ruce kapitolu 11 (Protokol DNS), pak je výstup ladícího režimu docela dobře čitelný. Ve výpisu jsou jednotlivé sekce uvozeny nadpisem. Do výpisu je pro lepší orientaci autorem doplněn český komentář.

Použití si ukážeme na příkladu. Zajímá nás IP adresa uzlu test100.pvt.net.

Dotaz:

```
> set debug
> test100.pvt.net
Server:  runner.pvt.cz
Address:  0.0.0.0
```

Odpověď:

```
------------
Got answer:     První odpověď neobsahuje ještě přeloženou adresu
    HEADER:     Sekce záhlaví
    opcode = QUERY, id = 1, rcode = NXDOMAIN
    header flags:  response, auth. answer, want recursion, recursion avail.
    questions = 1,  answers = 0,  authority records = 1,additional = 0

    QUESTIONS:  Sekce obsahující dotaz
        test100.pvt.net.pvt.cz, type = A, class = IN
    AUTHORITY RECORDS:  Sekce o autoritativních serverech
    ->  pvt.cz
        ttl = 129600 (1 day 12 hours)
        origin = mh.pvt.cz
        mail addr = hostmaster.pvt.cz
        serial = 1996020802
        refresh = 10800 (3 hours)
        retry   = 3600 (1 hour)
        expire  = 360000 (4 days 4 hours)
        minimum ttl = 129600 (1 day 12 hours)

------------
```

Druhý paket pak již v našem případě obsahuje odpověď:

```
------------
Got answer:
    HEADER:
       opcode = QUERY, id = 2, rcode = NOERROR
       header flags:  response, want recursion, recursion avail.
       questions = 1,  answers = 1,  authority records = 4,  additional = 4
```

```
    QUESTIONS:
        test100.pvt.net, type = A, class = IN
    ANSWERS:
    ->  test100.pvt.net
        internet address = 194.149.100.1
        ttl = 129175 (1 day 11 hours 52 mins 55 secs)
    AUTHORITY RECORDS:
    ->  pvt.NET
        nameserver = NS0.PIPEX.net
        ttl = 122697 (1 day 10 hours 4 mins 57 secs)
    ->  pvt.NET
        nameserver = NS1.PIPEX.net
        ttl = 122697 (1 day 10 hours 4 mins 57 secs)
    ->  pvt.NET
        nameserver = ns.pvt.net
        ttl = 122697 (1 day 10 hours 4 mins 57 secs)
    ->  pvt.NET
        nameserver = NS1.PVT.net
        ttl = 122697 (1 day 10 hours 4 mins 57 secs)
    ADDITIONAL RECORDS:
    ->  NS0.PIPEX.net
        internet address = 158.43.128.8
        ttl = 143625 (1 day 15 hours 53 mins 45 secs)
    ->  NS1.PIPEX.net
        internet address = 158.43.192.7
        ttl = 143625 (1 day 15 hours 53 mins 45 secs)
    ->  ns.pvt.net
        internet address = 194.149.105.18
        ttl = 129175 (1 day 11 hours 52 mins 55 secs)
    ->  NS1.PVT.net
        internet address = 194.149.103.201
        ttl = 129175 (1 day 11 hours 52 mins 55 secs)

------------
Non-authoritative answer:
Name:    test100.pvt.net
Address:  194.149.100.1
```

Resolver odeslal na jmenný server dva dotazy a jako odpověď obdržel dva pakety. Proč se jedná o dva dotazy by nám již mělo být jasné z kapitoly o resolveru. Pokud tomu tak není, napovím, pozorně se podívejte na doménové jméno, na které se v dotazu resolver ptá.

14.1.3 Ladící úroveň d2

Ladící úroveň d2 detailně vypisuje obsah odcházejících paketů (dotazů) i příchozích paketů (odpovědi). Použitím ladící úrovně d2 získáme úplný přehled o komunikaci resolveru se jmenným serverem, jehož vypovídající schopnost je téměř shodná s výstupy MS Network Monitoru.

Použití si ukážeme na příkladu stejném jako u ladící úrovně debug. Můžete porovnat odpovědi.

Dotaz:

```
> set d2
> test100.pvt.net
Server:  runner.pvt.cz
Address:  0.0.0.0
```

Odpověď:

```
------------
SendRequest(), len 40
    HEADER:
       opcode = QUERY, id = 3, rcode = NOERROR
       header flags:  query, want recursion
       questions = 1,answers = 0,authority records = 0,additional = 0

    QUESTIONS:
        test100.pvt.net.pvt.cz, type = A, class = IN

------------
------------
Got answer (96 bytes):
    HEADER:
    opcode = QUERY, id = 3, rcode = NXDOMAIN
    header flags:  response, auth. answer, want recursion, recursion avail.
    questions = 1,  answers = 0,  authority records = 1,  additional = 0

    QUESTIONS:
        test100.pvt.net.pvt.cz, type = A, class = IN
    AUTHORITY RECORDS:
    ->  pvt.cz
        type = SOA, class = IN, dlen = 38
        ttl = 129600 (1 day 12 hours)
        origin = mh.pvt.cz
        mail addr = hostmaster.pvt.cz
        serial = 1996020802
        refresh = 10800 (3 hours)
        retry   = 3600 (1 hour)
        expire  = 360000 (4 days 4 hours)
        minimum ttl = 129600 (1 day 12 hours)

------------
------------
SendRequest(), len 33
    HEADER:
       opcode = QUERY, id = 4, rcode = NOERROR
       header flags:  query, want recursion
       questions = 1,  answers = 0,  authority records = 0,  additional = 0

    QUESTIONS:
        test100.pvt.net, type = A, class = IN

------------
```

```
------------
Got answer (208 bytes):
    HEADER:
       opcode = QUERY, id = 4, rcode = NOERROR
       header flags:  response, want recursion, recursion avail.
       questions = 1,  answers = 1,  authority records = 4,  additional = 4
    QUESTIONS:
        test100.pvt.net, type = A, class = IN
    ANSWERS:
    ->  test100.pvt.net
        type = A, class = IN, dlen = 4
        internet address = 194.149.100.1
        ttl = 129025 (1 day 11 hours 50 mins 25 secs)
    AUTHORITY RECORDS:
    ->  pvt.NET
        type = NS, class = IN, dlen = 15
        nameserver = NS0.PIPEX.net
        ttl = 122547 (1 day 10 hours 2 mins 27 secs)
    ->  pvt.NET
        type = NS, class = IN, dlen = 6
        nameserver = NS1.PIPEX.net
        ttl = 122547 (1 day 10 hours 2 mins 27 secs)
    ->  pvt.NET
        type = NS, class = IN, dlen = 9
        nameserver = ns.pvt.net
        ttl = 122547 (1 day 10 hours 2 mins 27 secs)
    ->  pvt.NET
        type = NS, class = IN, dlen = 10
        nameserver = NS1.PVT.net
        ttl = 122547 (1 day 10 hours 2 mins 27 secs)
    ADDITIONAL RECORDS:
    ->  NS0.PIPEX.net
        type = A, class = IN, dlen = 4
        internet address = 158.43.128.8
        ttl = 143475 (1 day 15 hours 51 mins 15 secs)
    ->  NS1.PIPEX.net
        type = A, class = IN, dlen = 4
        internet address = 158.43.192.7
        ttl = 143475 (1 day 15 hours 51 mins 15 secs)
    ->  ns.pvt.net
        type = A, class = IN, dlen = 4
        internet address = 194.149.105.18
        ttl = 129025 (1 day 11 hours 50 mins 25 secs)
    ->  NS1.PVT.net
        type = A, class = IN, dlen = 4
        internet address = 194.149.103.201
        ttl = 129025 (1 day 11 hours 50 mins 25 secs)

------------
Non-authoritative answer:
Name:    test100.pvt.net
Address:  194.149.100.1
>
```

14.1.4 Změna implicitního jmenného serveru

DNS paket s dotazem můžeme pomocí programu nslookup poslat na libovolný jmenný server. Jméno testovaného serveru nastavíme příkazem *server*.

```
> server ns.internic.net
```

Po použití tohoto příkazu bude všechny následně zadané DNS dotazy řešit nově nastavený server, tedy v našem případě server ns.internic.net.

Toto nastavení je velice praktické, protože většinou se nám náš místní jmenný server z naší LAN jeví jako korektně pracující. Takže jistotu získáme, až si náš jmenný server ověříme z pohledu jiného jmenného serveru.

14.1.5 Výpis zóny

Pokud chceme, aby nám jmenný server poslal kompletní informace o nějaké zóně, pak použijeme příkaz ls –d. Dotaz v tomto případě musíme směrovat na autoritativní server pro doménu. Obvykle tedy příkazu ls –d předchází příkaz server.

```
>server ns.pvt.net
>ls -d pvt.cz
```

Příkaz ls –d simuluje zónový přenos ze jmenného serveru, proto se často používá při konfiguraci sekundárního jmenného serveru. Pomocí příkazu ls –d je možné snadno ověřit, zda primární jmenný server poskytuje data dané zony a zda je tedy poskytne sekundárnímu jmennému serveru pro zónu. Pokud se nám nepodaří získat tímto příkazem od primárního serveru odpověď, můžeme si být jisti, že se to s největší pravděpodobností nepodaří ani sekundárnímu jmennému serveru.

14.1.6 Simulace dotazů od jmenného serveru

Chceme-li simulovat komunikaci mezi jmennými servery, musíme potlačit dvě implicitní nastavení programu nslookup.

Program nslookup implicitně používá searchlist, tedy podobně jako resolver přidává implicitní doménu za doménové jméno, které není ukončeno tečkou. Toto chování zakážeme příkazem:

```
> set nosearch
```

Nslookup implicitně požaduje po jmenném serveru rekurzivní, tedy konečnou odpověď. Víme, že jmenné servery si mezi sebou posílají nerekurzivní odpovědi, a proto opět toto chování musíme potlačit, tentokrát příkazem:

```
> set norecurse
```

14.1.7 Nejčastější chybová hlášení programu nslookup

No records available – Neexistuje věta požadovaného typu

No response from server – Server neběží

No information – Server běží, ale nemá k doméně informace

Non-existent domain – Neexistuje reverzní záznam pro jméno jmenného serveru

Can't list domain …Query refused – Server běží, ale nemá k doméně data (data vypršela)

Unspecified error – Nespecifikovaná chyba

14.2 Další programy určené pro testování DNS

O některých dalších prostředcích pro ladění DNS informuje RFC-1713. Jde například o programy: ddt2, dnsparse, doc, host, inetrover a lamer, které jsou dostupné na ftp://ftp.uu.net/networking/ip/dns.

14.2.1 Dnswalk

Nejznámějším programem pro ladění DNS je program dnswalk. Dnswalk je skript napsaný v jazyce Perl. Program dnswalk „zná“ pravidla pro konfiguraci DNS a podle těchto pravidel kontroluje konfiguraci dané domény. Program dnswalk provede zónový přenos z autoritativního jmenného serveru a zkontroluje správnost konfigurace domény z mnoha pohledů. Program umí kontrolovat přímé domény i reverzní domény. Jméno kontrolované domény se programu předává jako parametr, jméno musí být ukončeno tečkou.

Opět je nejlépe spouštět dnswalk z cizího počítače, proto v Internetu existují webové servery nabízející formuláře na otestování cizích domén. Tyto webové servery spouští dnswalk jako cgi-skript.

Následující příklad ilustruje použití programu dnswalk pro kontrolu zóny pvt.net. (tečka na konci je povinná):

```
$ perl dnswalk pvt.net.
Getting zone transfer of pvt.net. from ns.pvt.net....done.
Checking pvt.net.
SOA=ns.pvt.net. contact=dostalek.pvt.cz.
 dhcp-cbu.pvt.net. A 194.149.104.3: no PTR record
 dhcp-cbu.pvt.net. A 194.149.104.11: no PTR record
 cbun000e01.pvt.net.    129600  CNAME   pipex-gw.pvt.net.pvt.net.: domain
occurred twice, forgot trailing '.'?
 cbun000e01.pvt.net. CNAME pipex-gw.pvt.net.pvt.net.: unknown host
```

Při kontrole dnswalk odhalil hned tři chyby:

V doméně pvt.net jsou uvedeny dvě věty typu A, ke kterým neexistuje odpovídající věta PTR. Ve jméně pipex-gw.pvt.net nám chybí tečka ne konci a CNAME tedy ukazuje na neexistující uzel.

Dnswalk může být spouštěn s různými parametry. Uveďme alespoň parametry pro nejčastěji používané kontroly:

Parametr	Význam
-f	Dnswalk zkouší provést zónový přenos z autoritativního jmenného serveru
-l	Kontrola na lame delegaci
-r	Rekurzivní kontrola subdomén domény
-a	Kontrola výskytu duplicitních vět typu A
-F	Kontrola vět A oproti větám PTR

Nejčastější volání programu dnswalk pro kontrolu domény je tedy:

```
dnswalk -Fralf domena.cz.
```

Dnswalk je dostupný na: ftp://ftp.math.psu.edu/pub/barr

14.2.2 Dig

Dalším ze známějších používaných programů pro kontrolu DNS je program dig. Program dig posílá na uvedený jmenný server paket DNS query a vrací uživateli informace o DNS. Uživatel může určit, který server má dotaz zodpovědět, jaké informace jej zajímají a specifikovat dodatečné podmínky dotazu. Výhodou tohoto programu je standardní formát odpovědi. Odpověď tedy můžeme dále zpracovávat svým programem. Zatímco nslookup se používá nejčastěji interaktivně, tak dig se spouští často z dávek.

Nejčastěji používaná syntaxe:

```
dig @server domena query-type
```

Za znakem @ uvedeme jméno serveru, kterého se chceme dotazovat. Druhý parametr je jméno domény, kterou chceme kontrolovat, query-type je požadovaný typ záznamu. Na místo řetězce query-type můžeme uvést libovolný typ záznamu RR nebo řetězec axfr, kterým požadujeme zónový přenos nebo řetězec any, který je požadavkem na libovolný typ záznamu.

Příklad použití programu dig:

V příkladu požadujeme kontrolu vět typu mx pro doménu pvt.net. Informace chceme od serveru ns.pvt.net.

```
dig @ns.pvt.net pvt.net mx

; <<>> DiG 2.1 <<>> @ns.pvt.net pvt.net mx
; (1 server found)
;; res options: init recurs defnam dnsrch
;; got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 10
;; flags: qr aa rd ra; Ques: 1, Ans: 2, Auth: 5, Addit: 15
;; QUESTIONS:
;;      pvt.net, type = MX, class = IN

;; ANSWERS:
pvt.net.        86400   MX      20 mail.uu.net.
pvt.net.        86400   MX      10 cbu.pvtnet.cz.

;; AUTHORITY RECORDS:
pvt.net.        86400   NS      ns.pvt.net.
pvt.net.        86400   NS      ns1.pvt.net.
pvt.net.        86400   NS      snmp0.pvt.net.
pvt.net.        86400   NS      ns0.pipex.net.
pvt.net.        86400   NS      ns1.pipex.net.

;; ADDITIONAL RECORDS:
mail.uu.net.    74570   A       192.48.96.15
mail.uu.net.    74570   A       192.48.96.16
mail.uu.net.    74570   A       192.48.96.17
mail.uu.net.    74570   A       192.48.96.5
mail.uu.net.    74570   A       192.48.96.7
mail.uu.net.    74570   A       192.48.96.8
mail.uu.net.    74570   A       192.48.96.14
cbu.pvtnet.cz.  86400   A       194.149.105.18
ns.pvt.net.     86400   A       194.149.105.18
```

```
ns1.pvt.net.    86400   A       194.149.103.201
snmp0.pvt.net.  86400   A       194.149.103.34
ns0.pipex.net.  16958   A       158.43.128.103
ns0.pipex.net.  16958   A       158.43.128.8
ns1.pipex.net.  16970   A       158.43.192.7
ns1.pipex.net.  16970   A       158.43.192.40

;; Total query time: 7 msec
;; FROM: info.pvt.net to SERVER: ns.pvt.net  194.149.105.18
;; WHEN: Tue Aug 18 11:15:20 1998
;; MSG SIZE  sent: 25  rcvd: 418
```

Místo jména serveru můžeme při volání programu dig uvést IP adresu dotazovaného serveru.

14.3 Deset nejčastějších chyb v konfiguraci DNS

1. Každý uzel na Internetu by měl mít doménové jméno korektně zavedené v DNS. Některé služby kontrolují existenci jména v DNS a bez jeho existence s uzlem nekomunikují.
2. Doménové jméno nesmí obsahovat jiné znaky, než ACSII písmena, číslice a znak pomlčka. Jméno by nemělo být tvořeno pouze číslicemi. Jméno nesmí začínat nebo končit pomlčkou. Podtržítko je jako součást doménového jména sice povoleno v RFC-1033, není však definováno jako standard, některé implementace mají s podtržítkem problémy, proto se podtržítku zásadně vyhýbáme.
3. Na konci úplných doménových jmen musí být tečka. Za IP adresou se tečka nedělá.
4. V mailové adrese v záznamu SOA musí být znak @ nahrazen tečkou.
5. Na pravé straně NS záznamu musí být uvedeno kanonické jméno, v NS záznamu nesmí být na pravé straně IP adresa.
6. Záznam A a záznam PTR musí obsahovat shodné informace.
7. V záznamech PTR, MX, NS a CNAME nesmí být na pravé straně uveden alias. Chcete-li mít pro uzel stejné jméno jako pro doménu, použijte následující konstrukci:

   ```
   firma.cz. IN      NS      ns1.firma.cz.
             IN      NS      ns2.firma.cz
             IN      A       1.2.3.4
   ```

8. Ke každému záznamu A musí existovat věta PTR. Uzel s více IP adresami musí mít více PTR vět.
9. Lame delegace – autoritativní jmenný server neobsahuje data pro doménu. Tento stav často vzniká po zrušení sekundárního jmenného serveru.

   Klasickým případem lame delegace je situace, kdy primární jmenný server pracuje korektně a sekundární jmenný server je chybně nakonfigurován (chybí věta v named.boot, neprovedl se zónový přenos atd.). Pak se stane, že v doméně vyšší úrovně je nastaven inkriminovaný nenakonfigurovaný jmenný server jako autoritativní jmenný server pro doménu.

   Odněkud je požadován dotaz na jméno z této domény. Nadřízený jmenný server odpoví, ať se tazatel dotáže našeho chybně nakonfigurovaného jmenného serveru coby autority. Tazatel tedy položí dotaz nenakonfigurovanému jmennému serveru, který by měl být autoritou, ale ten neví, co odpovědět …

10. Přilepená (*glue*) věta se neuvádí u reverzních domén. Pokud má jmenný server více IP adres, musí být v nadřízené doméně uvedeny glue věty pro všechny IP adresy.

15

Delegace a registrace domén

Delegace domény se skládá z několika kroků:

1. Zprovoznění primárního jmenného serveru pro doménu.
2. Konfigurace sekundárního jmenného serveru pro doménu, případně požádáme o jeho konfiguraci svého poskytovatele Internetu.
3. Žádost o delegaci domény v doméně vyšší úrovně.
4. Pokud se jedná o doménu druhé úrovně, je třeba ještě registrovat doménu v databázi domén.

Celý postup delegace a registrace domény si ukážeme na příkladu delegace subdomény domény cz.

Vžijme se do situace hostmastera společnosti Firma s.r.o, která se rozhodla používat doménu firma.cz.

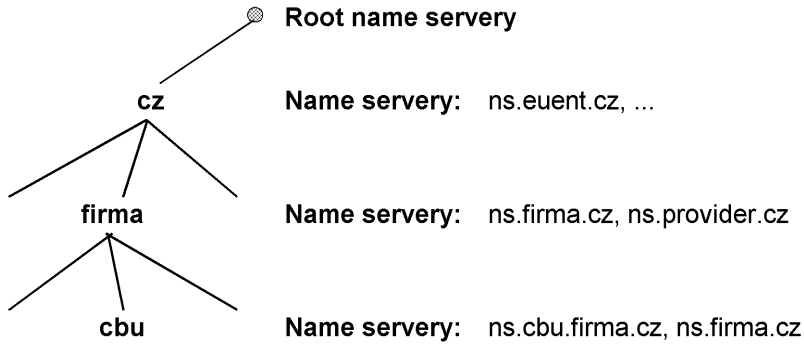
Firma s.r.o. je připojena k Internetu pevnou linkou. V případě, že by firma nebyla připojena pevnou linkou, pak není možné, aby si firma sama spravovala jmenný server. V takovém případě musí firma přenechat správu své domény např. poskytovateli Internetu.

15.1 Příklad 1

Správce společnosti rozhodl, že primární jmenný server bude spravovat na jednom z počítačů společnosti. Vybraný uzel bude mít jméno ns.firma.cz a IP adresu 194.149.10.11. Na uzlu ns.firma.cz je nainstalován Unix a BIND 4.9. Sekundární jmenný server bude spravovat poskytovatel na počítači jménem ns.provider.net.

Následují jednotlivé konfigurační soubory, respektive jejich části, kterými je zajištěna požadovaná delegace.

Obr. 15.1
Příklad delegace domén



Server ns.firma.cz

Soubor named.boot

```
…
primary         firma.cz        firma.cz.zone
…
```

Soubor firma.cz.zone

```
@               IN      SOA     ns.firma.cz hostmaster.firma.cz (
                              1998082402      ; Serial
                                   28800      ; Refresh 8 hours
                                    7200      ; Retry   2 hour
                                  604800      ; Expire  7 days
                                   86400 )    ; Minimum TTL 1 day

; ns záznamy, které specifikují autoritativní jmenné servery
            IN          NS          ns.firma.cz.
            IN          NS          ns.provider.net.
; platný záznam A pro server ns.firma.cz
ns              IN          A               194.149.10.11
…
```

Server ns.provider.net

Soubor named.boot

```
…
secondary               firma.cz          194.149.10.11 firma.cz.zone
```

Nyní ještě uvedeme informace, které se musí doplnit do konfiguračního souboru samotné domény CZ. Tyto informace doplní správce domény CZ až po příslušné administrativní proceduře popsané v kapitolách 15.3 a 15.4.

Soubor cz.zone

```
…
; ns záznamy zajišťující delegaci domény
firma                   IN  NS          ns.firma.cz.
                        IN  NS          ns.provider.net.
; platný záznam pro server ns.firma.cz
ns.firma        IN      A           194.149.10.11
```


15.2 Příklad 2

V rámci domény firma.cz firma plánuje vytvoření subdomény cbu.firma.cz pro svou pobočku v Českých Budějovicích. Pobočka v Českých Budějovicích si bude spravovat opět svůj vlastní jmenný server se jménem ns.cbu.firma.cz s IP adresou 194.149.10.129. Sekundární jmenný server pro doménu cbu.firma.cz bude nakonfigurován na uzlu ns.firma.cz.

Následují jednotlivé konfigurační soubory, respektive jejich části, kterými je zajištěna požadovaná delegace. Do konfiguračních souborů z předcházejícího příkladu doplním řádky, které se vztahují na delegaci domény cbu.firma.cz, řádky jsou zvýrazněny kurzívou.

Server ns.firma.cz

Soubor named.boot

```
…
primary    firma.cz                    firma.cz.zone
secondary      cbu.firma.cz         194.149.10.129     cbu.firma.cz.zone
```

Soubor firma.cz.zone

```
@               IN      SOA     ns.firma.cz hostmaster.firma.cz (
                                    1998082402      ; Serial
                                         28800      ; Refresh 8 hours
                                          7200      ; Retry   2 hour
                                        604800      ; Expire  7 days
                                         86400 )    ; Minimum TTL 1 day

; ns záznamy, které specifikují autoritativní jmenné servery
                  IN          NS          ns.firma.cz.
                  IN          NS          ns.provider.net.
; platný záznam A pro server ns.firma.cz
ns                IN          A           194.149.10.11
;
; delegace subdomény cbu.firma.cz na server ns.cbu.firma.cz,
; ns záznamy zajišťující delegaci

cbu               IN          NS          ns.cbu.firma.cz.
                  IN          NS          ns.firma.cz
; připojený záznam pro server ns.cbu.firma.cz
ns.cbu            IN          A           194.149.10.129
```

Server ns.cbu.firma.cz

Soubor named.boot

```
…
primary         cbu.firma.cz    cbu.firma.cz.zone
```

Soubor cbu.firma.cz.zone

```
@               IN      SOA     ns.cbu.firma.cz hostmaster.cbu.firma.cz (
                                  1998082502      ; Serial
                                       28800      ; Refresh 8 hours
                                        7200      ; Retry   2 hour
                                      604800      ; Expire  7 days
                                       86400 )    ; Minimum TTL 1 day
; ns věty, které určují autoritativní jmenné servery
                           IN          NS          ns.cbu.firma.cz.
                           IN          NS          ns.firma.cz..
; platná věta A pro server ns.cbu.firma.cz
ns                         IN          A           194.149.10.129
```

Připomeňme ještě důležitost připojeného (*glue*) záznamu typu A. Připojený záznam typu A musíme uvést v nadřízené doméně, pokud je doména delegována na server, který používá jméno v rámci delegované domény. V příkladu primární server domény cz deleguje autoritu pro doménu firma.cz na server ns.firma.cz. Tedy jméno primárního jmenného serveru ns.firma.cz je z domény firma.cz

Připojený záznam z prvního příkladu se využívá na uzlu ns.eunet.cz při překladu jména ns.firma.cz.

Důležité je uvědomit si skutečnost, že připojený záznam se udržuje v paměti spolu s NS záznamy na každém jmenném serveru, který řeší překlad nějakého jména z domény firma.cz. U připojeného záznamu se udržuje podle TTL uvedeného v nadřízené zóně.

Jakmile se nám podařilo nakonfigurovat a spustit primární i sekundární jmenný server pro doménu firma.cz, budou všechny uzly, které mají na tyto servery nasměrovaný resolver schopny překládat jména z domény firma.cz. Naším cílem je, aby všechny rozlišovače (resolvery) v Internetu uměly překládat jména z domény firma.cz. Požadovaného stavu dosáhneme, pokud na naše jmenné servery deleguje autoritu správce nadřízené domény. Proto v našem snažení pokračujeme dále a požádáme o delegaci a registraci domény firma.cz na jmenné servery ns.firma.cz a ns.provider.net.

Nyní si musíme připomenout, že za registraci domény a její držení se platí. Platbu můžeme provádět sami, nebo můžeme platit prostřednictvím našeho poskytovatele Internetu. Před vlastní registrací domény se tedy rozhodneme, jak budeme zajišťovat platbu. My jsme se rozhodli platit sami, a proto musíme zaregistrovat tzv. fakturační kontakt pro doménu. Pokud se vy rozhodnete pro druhou variantu, můžete následující kapitolu přeskočit.

15.3 Registrace subdomén domény cz

Bylo založeno sdružení CZ.NIC, které je oprávněno provádět registraci subdomén domény cz. Od 1. 9. 1999 vyhlásilo CZ.NIC zásadní změny v této registraci. Přesné aktuální informace lze nalézt na http://www.nic.cz.

15.3.1 Dokumenty CZ.NIC

Na serveru http://www.nic.cz jsou umístěny dokumenty:

- Pravidla registrace doménových jmen v doméně .cz
- Smlouva o registraci doménového jména
- Ceník služeb registrace doménových jmen v doméně .cz
- Fakturační kontakt
- Postup při řešení sporů souvisejících s doménovými jmény
- Zahájení sporu v rámci domény .cz

Z uvedených dokumentů je již zřejmé, že se jedná o placenou službu. Ceny jsou uvedeny v dokumentu „Ceník služeb registrace doménových jmen v doméně .cz".

Dokument „Smlouva o registraci doménového jména" obsahuje vzor smlouvy, která se na příslušnou registraci uzavírá. Žádosti o domény se vyřizují sekvenčně, tak jak přicházejí za sebou. Při registraci domény může mezi jednotlivými firmami dojít ke kolizi v tom, že si firma chce zaregistrovat doménu a při registraci zjistí, že doména stejného jména je již registrována pro jinou firmu.

Místo toho, aby se obě firmy vzájemně dohodly a vytvořily společnou webovou stránku s rozskokem na své servery, tak s největší pravděpodobnosti vznikne pře o doménu (u webového serveru by dohoda byla pravděpodobně jednoduchá, ale u e-mailu je to již komplikovanější). Postup při takových přích řeší dokumenty „Zahájení sporu v rámci domény .cz" a „Postup při řešení sporů souvisejících s doménovými jmény".

Základním dokumentem pro registraci subdomény domény cz je dokument „Pravidla registrace doménových jmen v doméně .cz". Jelikož se jedná o zásadní dokument pro správce DNS, tak uvádíme jeho verzi platnou od 1. 9. 1999 (zdroj: http://www.nic.cz/cznic/rules.html):

Pravidla registrace doménových jmen v doméně .cz

V platnosti od 1. 9. 1999

Tato Pravidla registrace doménových jmen v doméně .cz jsou nedílnou součástí Smlouvy o registraci.

1. Úvod

1.1. Účelem tohoto dokumentu je stanovit pravidla pro registraci („Pravidla") doménových jmen druhého stupně na Internetu („Doménová jména") pod doménou nejvyššího stupně .cz („Doména .cz"). Dále pro účely těchto Pravidel označuje výraz „Žadatel" fyzickou nebo právnickou osobu, která si pro sebe hodlá nechat registrovat Doménové jméno, a výraz „Zástupce" označuje řádnou plnou mocí zplnomocněného zástupce Žadatele.

Sdružení CZ.NIC, z.s.p.o. („CZ.NIC") je oprávněné registrovat doménová jména a zajišťovat transparentnost a správnost takového procesu registrace, pro který jsou tato Pravidla stanovena. CZ.NIC může být při provádění registrací nebo správy Doménových jmen nebo při souvisejících činnostech zastoupen svým smluvním partnerem, přičemž platnost těchto Pravidel zůstane i pro takové případy plně zachována. Pro účely těchto Pravidel mají výrazy užité s velkými počátečními písmeny stejný význam jako ve Smlouvě o registraci.

Pokud není dále výslovně uvedeno jinak, je pro účely těchto Pravidel a souvisejících dokumentů považován Žadatel o registraci za vlastníka žádosti o registraci Doménového jména, resp. za vlastníka registrace Doménového jména.

1.2. Tato Pravidla se vztahují na všechna Doménová jména, jež mají být umístěna do Domény .cz.

2. Principy registrace

2.1. CZ.NIC registruje Doménová jména podle došlého pořadí. Při určování pořadí rozhoduje datum a čas, kdy CZ.NIC obdržel vyplněný Formulář žádosti o registraci, osvědčující akceptaci Smlouvy o registraci včetně Pravidel Žadatelem. CZ.NIC je povinen potvrdit převzetí Formuláře žádosti o registraci a na potvrzení řádně a přesně v celých ukončených minutách uvést datum a čas převzetí. CZ.NIC rovněž neprodleně zanese údaje o každém převzatém Formuláři žádosti o registraci do svého logu. Pokud CZ.NIC zjistí formální nedostatky výše uvedených dokumentů zabraňující jejich plynulému zpracování, bude žádost neprodleně odmítnuta. V případě konfiguračních nebo provozních závad nameserverů, určí CZ.NIC Žadateli lhůtu 14 dnů k jejich odstranění. V případě, že Žadatel neodstraní oznámené nedostatky v této 14 denní lhůtě, registrační řízení je zastaveno a žádost žadatele zamítnuta.

3. Postup

3.1 Žadatel o registraci Doménového jména nebo jeho Zástupce zašle CZ.NICu řádně vyplněný Formulář žádosti. Formulář žádosti bude zaslán v elektronické formě na následující adresu (adresy) CZ.NICu: autoreg@nic.cz v souladu s pravidly vzájemné komunikace (viz bod 4. níže). Provedením tohoto kroku Žadatel (a nebo jeho Zástupce) současně potvrzují, že přijímají Smlouvu o registraci a zavazují se dodržovat její podmínky.

3.2 CZ.NIC po přijetí vyplněného Formuláře žádosti odešle Žadateli nebo jeho Zástupci elektronickou cestou protokol o přijetí žádosti, potvrzující zahájení registračního řízení. Pokud žádost Žadatele obsahuje nedostatky, bude Žadatel nebo jeho zástupce vyzván k jejich odstranění podle bodu 2.1 výše.

3.3 Poté, co CZ.NIC přijme Smlouvu o registraci na dané Doménové jméno uvedené ve Formuláři žádosti Žadatele, zapíše Doménové jméno a související údaje uvedené na Formuláři žádosti do rejstříku .cz Doménových jmen („Rejstřík"). Současně CZ.NIC zašle Žadateli, resp. jeho Zástupci výzvu k úhradě poplatků elektronickou poštou. Po obdržení úhrady zašle CZ.NIC Žadateli, resp. Jeho zástupci fakturu a počítačový výpis, obsahující údaje o provedené registraci a o majiteli registrovaného Doménového jména („Výpis z rejstříku").

3.4 Pokud CZ.NIC neobdrží platbu nejpozději do 30 dnů po uplynutí data splatnosti výzvy k úhradě, pozastaví s okamžitou platností registraci Doménového jména a zašle Žadateli a jeho Zástupci elektronickou poštou oznámení o pozastavení registrace a vyřazení Domény ze zóny CZ. Současně v tomto oznámení vyzve CZ.NIC Žadatele a jeho Zástupce, aby po vzájemné dohodě zaplatili Registrační poplatek v další 30-ti denní lhůtě. Pokud CZ.NIC neobdrží platbu Registračního poplatku ani v této dodatečné lhůtě, zruší s okamžitou platností registraci Doménového jména a informuje o tomto kroku elektronickou poštou Žadatele a jeho Zástupce.

3.5 Pokud CZ.NIC Smlouvu o registraci na dané Doménové jméno nepřijme, oznámí to Žadateli nebo jeho Zástupci s uvedením důvodů.

4. Pravidla vzájemné komunikace

4.1 Pokud kterákoliv ze stran této Smlouvy o registraci má zaslat jiné straně této Smlouvy o registraci určité dokumenty nebo zprávy elektronickou cestou, zavazuje se dodržovat pravidla elektronické komunikace („Pravidla komunikace"). CZ.NIC je oprávněn Pravidla komunikace průběžně měnit s tím, že jejich aktuální znění lze nalézt na http://www.nic.cz na World Wide Web.

4.2 Všechny strany této Smlouvy o registraci uznávají platnost a hodnověrnost elektronické komunikace prováděné v souladu s touto Smlouvou o registraci a dále se zavazují, že budou uznávat platnost a hodnověrnost elektronického logu dat CZ.NICu, pokud se neprokáže něco jiného.

5. Řešení sporů

5.1. V případě jakéhokoliv sporu týkajícího se Smlouvy o registraci včetně těchto Pravidel a nebo registrovaných Doménových jmen a nebo žádostí o registraci se zúčastněné strany budou snažit vyřešit sporné otázky dohodou. Pro tyto účely mohou zvážit použití neformální arbitráže, jejíž základní zásady jsou popsány v dokumentu s názvem „Postup řešení sporů souvisejících s doménovými jmény" („Postup řešení sporů"), který lze nalézt na http://www.nic.cz na World Wide Web. Pokud se strany nedohodnou na použití Postupu řešení sporů, mají plnou volnost vyřešit svůj spor v rámci platných právních předpisů.

6. Práva CZ.NICu

6.1. CZ.NIC je oprávněn dle svého uvážení kdykoliv zrušit nebo pozastavit provádění registrace nebo pozastavit nebo zrušit registraci Doménového jména s okamžitou platností, mimo jiné zjistí-li či se jinak dozví o tom, že došlo k následujícím situacím:

- pokud je Doménové jméno spravováno způsobem, který může ohrozit činnost internetové sítě nebo jakékoli její části (například situace, kde nameservery příslušející k registrovanému Doménovému jménu jsou mimo provoz déle než 14 po sobě následujících dnů, bude velmi pravděpodobně považována za ohrožení provozu místní části internetové sítě a CZ.NIC je oprávněn vykonat svá práva podle tohoto ustanovení);
- pokud dojde ke změně skutečností, na jejímž základě bylo Doménové jméno zaregistrováno, například organizace, která předložila žádost, již neexistuje;
- pokud se CZ.NIC dozví, že je Doménové jméno prokazatelně užíváno způsobem, který může být klamavým pro uživatele Internetu, je oprávněn pozastavit registraci Doménového jména;
- v dalších případech, kdy je právo CZ.NICu zrušit nebo pozastavit registraci Doménového jména výslovně upraveno ve Smlouvě o registraci.

7. Pravidla pro tvorbu Doménového jména

7.1. Nejsou povolena Doménová jména skládající se ze dvou písmen odpovídajících ISO kódům jednotlivých států.

7.2. Veškeré domény nejvyššího stupně nesmějí být užívány jako domény druhého stupně (například com.cz nebo uk.cz).

7.3. Veškeré nové požadavky na domény musí vyhovovat normám RFCs 1034, 1035, 1122, 1123 a jakýmkoliv je nahrazujícím normám.

8. Servery

8.1. Pro doménu se vyžadují alespoň dva nameservery, jejichž adresy musí být uvedeny ve Formuláři žádosti přiloženému ke Smlouvě o registraci a které jsou funkční v doběpředložení žádosti.

9. Požadavky na osobní přítomnost

9.1. Pokud je Žadatel fyzickou osobou, musí mít kontaktní adresu v České republice. Pokud je Žadatel právnickou osobou, musí být buď založen v ČR a nebo zde musí vyvíjet obchodní činnost a musí mít kontaktní adresu na území České Republiky.

10. Změna Pravidel

10.1 Pozdější změna Pravidel neovlivní stav jmen, která byla zapsána do Rejstříku před změnou, kromě věrohodných technických důvodů.

CZ.NIC z.s.p.o.
13. 8. 1999

14.4.2 Technický provoz domény .cz

Technický provoz zajišťuje firma KPNQwest Czechia s.r.o. Jádrem provozu je databáze o registrovaných doménách, jejich správcích, bankovních kontaktech atd. Tato databáze bude synchronizována s databází RIPE (RIPE viz kapitola 17).

Na http://www.nic.cz jsou vystaveny informace jak postupovat (komunikovat) při registraci a změnách registrace.

S registrací domény je nejjednodušší se obrátit na vašeho poskytovatele Internetu a celou relativně komplikovanou záležitost ponechat na něm. Pokud přece jenom chcete proniknout do systému registrace subdomény domény cz, pak před tím než začnete, důkladně prostudujte aktuální dokumentaci na http://www.nic.cz. Vystavené dokumenty jsou precizně vypracovány. Dále uvádíme dva z těchto dokumentů (*Pravidla komunikace a Popis rozhraní formulářů)*, které považujeme za základní.

KPNQwest Czechia s.r.o.
Technický provoz domény nejvyšší úrovně .cz

Pravidla komunikace

v1.0 1.12.1999 EUnet

I. Vymezení pojmů

1. Žadatel (Vlastník) – IDADM

Právnická nebo fyzická osoba, na níž bude doména registrovaná.

2. Plátce – IDACC

Právnická nebo fyzická osoba, která bude odvádět poplatky za registraci a správu domény Žadatele (Vlastníka).

3. Technický správce – IDTECH

Právnická nebo fyzická osoba, která je zplnomocněná vykonávat činnosti spojené s technickým provozem domény.

4. Žádající

Odesílatel elektronické žádosti o registraci, změnu, převod či zrušení domény.

II. žádost o novou doménu

1. Žádající zašle předepsaný formulář el. poštou na adresu auto-reg@nic.cz
2. Provede se administrativní kontrola žádosti; pokud něco není v pořádku, žádost se odmítne a Žádající obdrží el. poštou oznámení o odmítnutí žádosti. Žádostí se nadále již nikdo nezabývá.
3. Pokud je vše v pořádku, žádost se zařadí do fronty k dalšímu zpracování a Žádající obdrží el. poštou oznámení o přijetí žádosti včetně ticketu.
4. Provádí se technická kontrola žádosti; pokud není něco v pořádku, Žádající dostane el. poštou oznámení o chybě; kontrola se provádí opakovaně 2x/den; pokud nedojde k odstranění chyby do 7 dnů, Žádající obdrží el. poštou oznámení o odmítnutí žádosti; žádostí se nadále již nikdo nezabývá.
5. Pokud je vše v pořádku, požadovaná doména je zanesena do zóny .CZ a je o tom informován Žádající. Plátce obdrží el. poštou výzvu k zaplacení poplatku.
6. Nezaplatí-li Plátce poplatek do 30 dnů od odeslání výzvy, obdrží Plátce a Vlastník el. poštou 1. upomínku; nepřijde-li platba do 14 dnů od odeslání 1. upomínky, Plátce a Vlastník obdrží el. poštou 2. upomínku; doména je vyřazena z DNS a Plátce, Vlastník a Technický správce obdrží el. poštou oznámení o vyřazení z DNS; nepřijde-li platba do 14 dnů od odeslání 2. upomínky, doména je zrušena a uvolněna pro další zájemce; Vlastník, Plátce i Technický správce je informován o zrušení domény el. poštou a žádostí se nadále již nikdo nezabývá.
7. Po zaplacení poplatku obdrží Plátce pozemní poštou písemný daňový doklad o zaplacení poplatku a Vlastník el. poštou výpis z rejstříku domén.
8. 30 dní před vypršením platnosti poplatku za doménu Plátce obdrží výzvu k zaplacení ročního poplatku a opakuje se postup 6.-8.

III. žádost o změnu v doméně, převod či zrušení domény

1. Žádající zašle předepsaný formulář el. poštou na adresu auto-chg@nic.cz.
2. Provede se automatická část administrativní kontroly žádosti; pokud něco není v pořádku, žádost se odmítne a Žádající obdrží el. poštou oznámení o odmítnutí žádosti, žádostí se nadále již nikdo nezabývá.
3. Pokud je vše v pořádku, žádost se zařadí do fronty k dalšímu zpracování a Žádající obdrží el. poštou oznámení o zařazení do fronty včetně ticketu.
4. Žádající zašle předepsané písemné dokumenty s ticketem buď pozemní poštou na adresu:
   KPNQwest Czechia s.r.o.
   správa domény .cz
   Generála Janouška 902
   198 00 Praha 9 Černý Most
   anebo, pokud je to pro požadovaný typ změny akceptováno, faxem na číslo:
   +420 2 81081 082

5. Provede se manuální část administrativní kontroly žádosti (autentikace a autorizace s pomocí písemných dokumentů); pokud žádost není v pořádku, Žádající obdrží el. poštou, případně pozemní poštou nebo faxem oznámení o chybě; pokud není manuální část administrativní kontroly žádosti ukončena do 14 dnů od data na ticketu, Žádající obdrží el. poštou oznámení o odmítnutí žádosti a žádostí se nadále již nikdo nezabývá.
6. Pokud se jedná o změnu nameserverů nebo spojovacího záznamu (glue) provádí se technická kontrola žádosti; pokud není něco v pořádku, Žádající dostane el. poštou oznámení o chybě; kontrola se provádí opakovaně 2x/den; pokud nedojde k odstranění chyby do 1 dne, Žádající obdrží el. poštou oznámení o odmítnutí žádosti a žádostí se nadále již nikdo nezabývá 7. pokud je vše v pořádku, požadovaná změna je zanesena do databáze domén a do zóny .CZ; Žádající obdrží el. poštou potvrzení o provedení požadavku.

Pozn.: Nová data v databázi registračního systému se promítnou do zóny na primárním jmenném serveru pro doménu .CZ nejpozději do 24 hodin od provedení změny v databázi registračního systému.

KPNQwest Czechia s.r.o.
Technický provoz domény nejvyšší úrovně .cz

Popis rozhraní formulářů

v1.2 26.10.1999 EUnet

Následující popis obsahuje specifikaci rozhraní formulářů pro registraci a změny domén a souvisejících údajů. Položky označené [ ] nemusí při vyplňování formulářů obsahovat žádnou hodnotu. Při vyplňování položek, které mohou obsahovat více než jednu hodnotu (např. jména nameserverů nebo id kontaktních osob subjektů) je oddělovačem mezi jednotlivými hodnotami „ „. Všechny formuláře se skládají z hlavičky a těla. Hlavička obsahuje klíčové slovo RSDversion a platné číslo verze (2.1).

I. Rozhraní formuláře pro registraci

1. Registrace odpovědných osob subjektu

fname Křesní jméno osoby, znaková položka s maximální délkou 64 znaků

lname Příjmení osoby, znaková položka s maximální délkou 128 znaků

e-mail E-mailová adresa ve struktuře mail@domena s maximální délkou 128 znaků

id Idetifikátor osoby, znaková položka s maximální délkou 18 znaků. Může obsahovat znaky a-Z bez diakritiky, 0-9, _ a -

[notify] E-mail, kam se pošle informace o provedení změny

[phone] Telefonní číslo osoby s nepovinnou strukturou +420 2 12345678 s maximální délkou 64 znaků

[fax-no] Faxové číslo osoby s nepovinnou strukturou +420 2 12345678 s maximální délkou 64 znaků

[street] Ulice a číslo orientační lomeno číslo popisné

[zip] PSČ ve struktuře 182 00

[city] Název obce, znaková položka s maximální délkou 64 znaků
[country] Kód státu (ISO)
end Ukončení formuláře bez hodnoty atributu

Příklad:

```
RSDversion 2.1
--------------------------

fname: Jan
lname: Novak
e-mail: Jan.Novak@blabla.cz
id: HONZA123
phone: +420 2 12345678
fax-no: +420 2 12345678
notify: Jan.Novak@blabla.cz
street: Na hrazi 21/345
city: Trebon
zip: 345 32
country: cz
end:
```

2. Registrace subjektu

typ Typ subjektu [P,F] Právnická nebo fyzická osoba – určuje další zpracování atributu
name Obchodní jméno subjektu nebo jméno a příjmení fyzické osoby
id Identifikátor subjektu, znaková položka s maximální délkou 18 znaků. Může obsahovat znaky a-Z bez diakritiky, 0-9, _ a -
[fnane] Křestní jméno osoby, znaková položka s maximální délkou 64 znaků
[lname] Příjmení osoby, znaková položka s maximální délkou 128 znaků
[rc] Rodné číslo
[ico] Identifikační číslo, číselná položka ve struktuře 00123456
[dic] Daňové identifikační číslo, znaková položka ve struktuře 002-00123456
[bank] Bankovní spojení, položka ve struktuře číslo účtu/kód banky
[e-mail] E-mailová adresa ve struktuře mail@domena s maximální délkou 128 znaků
street Ulice a číslo orientační lomeno číslo popisné
zip PSČ ve struktuře 182 00
city Název obce, znaková položka s maximální délkou 64 znaků
[country] Kód státu (ISO)
admin-c id kontaktních osob (administrativní)
[tech-c] id kontaktních osob (technická)
[acc-c] id kontaktních osob (fakturační)
end Ukončení formuláře bez hodnoty atributu

Příklad:

```
RSDversion 2.1
-------------------------
typ: P
name: NOVACI s.r.o. rc:
fname:  lname:
ico: 12345678
dic: 001-12345678
bank: 00123477/0800
e-mail: novaci@xxxxx.cz
id: NN-123_XXX
street: Davidkova 59a/1132
city: Praha 8
zip: 180 00
country: cz
admin-c: Jouda
tech-c: Jouda
acc-c: Jouda
end:
```

3. Registrace domény

domain Jméno registrované domény

[desc] Popis k doméně

idadm Identifikátor žadatele – budoucího vlastníka (musí být již registrovaný subjekt)

[idtech] Identifikátor technického správce (musí být již registrovaný subjekt)

[idacc] Identifikátor plátce (musí být již registrovaný subjekt)

nserver Jména nameserverů

[glue] Jméno a IP adresa nameserveru

[pubkey] veřejný klíč

end Ukončení formuláře bez hodnoty atributu

Pozn.: glue záznam se vyplňuje pouze v případě, že daný nameserver leží v právě registrované doméně, jméno a IP adresa jsou odděleny mezerou. V případě, že v dané doméně leží oba nameservery, je nutné v glue uvést oba dva, záznamy jsou pak mezi sebou odděleny středníkem.

Příklad:

```
RSDversion 2.1
-------------------------
domain: blabla.cz
desc: Popis nove domeny
idadm: OWNER123
idtech: TECH123
idacc: FAK123
nserver: ns1.blabla.cz ns2.blabla.cz
glue: ns1.blabla.cz 124.11.124.1;ns2.blabla.cz 124.11.128.1
pubkey:
end:
```

II. Rozhraní formuláře pro změny

1. Změny údajů odpovědných osob subjektu

Struktura údajů předávaných při změnách údajů odpovědných osob subjektu je totožná se strukturou platnou pro registraci osoby. Mimo položky obsahující hodnoty, které jsou předmětem změny, musí být uvedena položka id osoby.

Příklad:

```
RSDversion 2.1
-------------------------
fname:
lname:
e-mail:
id: HONZA123
phone:
fax-no:
notify:
street: V rybniku 1/255
city:
zip:
country:
end:
```

2. Změny údajů subjektu

Struktura údajů předávaných při změnách údajů subjektu je totožná se strukturou platnou pro registraci subjektu. Mimo položky obsahující hodnoty, které jsou předmětem změny, musí být uvedena položka id osoby.

Příklad:

```
RSDversion 2.1
-------------------------
typ:
name:
rc:
fname:
lname:
ico:
dic:
bank:
e-mail:
id: NN-123_XXX
street:
city:
zip:
country: cz
admin-c: Jouda Jouda1 Jouda2
tech-c:
acc-c: Ucetni1 Ucetni2
end:
```

3. Změny údajů domény

Struktura údajů předávaných při změnách údajů domény je totožná se strukturou platnou pro registraci domény. Mimo položky obsahující hodnoty, které jsou předmětem změny, musí být uvedena položka domain. Případná změna idadm (vlastník) bude ignorována.

Příklad:

```
RSDversion 2.1
-------------------------
domain: blabla.cz
desc:
idadm:
idtech:
idacc:
nserver: ns3.ns3.cz ns4.ns4.cz
glue:
pubkey:
end:
```

Pozn.: glue záznam se vyplňuje pouze v případě, že jde o nový nameserver ležící v dané doméně, jméno a IP adresa jsou odděleny mezerou. V případě, že jde o 2 nameservery, je nutné v glue uvést oba dva, záznamy jsou pak mezi sebou odděleny středníkem. Při technické kontrole se kontroluje správnost trojice domain-nserver-glue.

III. Rozhraní formuláře pro převod a zrušení domény

1. Převod vlastnictví domény

domain Registrovaná doména

oldid Identifikátor subjektu, který byl vlastníkem domény

newid Identifikátor subjektu, který se stane novým vlastníkem

end Ukončení formuláře bez hodnoty atributu

Příklad:

```
RSDversion 2.1
-------------------------
domain: blabla.cz
oldid: OWNER1
newid: OWNER2
end:
```

2. Zrušení domény

domai Registrovaná doména

idadm Idetifikátor subjektu – vlastníka

end Ukončení formuláře bez hodnoty atributu

Příklad:

```
RSDversion 2.1
-------------------------
domain: blabla.cz
idadm: OWNER
end:
```

16

Delegace a registrace reverzních domén

Reverzní překlad je překlad IP adresy na doménové jméno. Již víme, že záznam definující vazbu IP adresy na doménové jméno je záznam PTR. Reverzní překlad využívají některé programy jako například ftp, traceroute, apod. Pokud není v DNS zaveden reverzní záznam pro doménové jméno, pak některé služby, jako je např. ftp, mohou odmítnout korektně pracovat. Je tedy velmi důležité nezapomínat na záznamy PTR, a tedy na reverzní domény.

Reverzní doména je vždy vytvářena a delegována pro síť IP adres. Např. pro síť 194.149.177 musí být v DNS vytvořena a delegována reverzní doména 177.149.194.in-addr.arpa. Reverzní doména tedy nijak nesouvisí s klasickou doménou. V jedné reverzní doméně se mohou vyskytovat a často se i vyskytují doménová jména z různých domén.

Typy reverzních domén se odvíjejí od rozsahu používané sítě. Uživatel obvykle pro svou síť používá rozsah 256 IP adres – síť třídy C nebo subsíť třídy C. Poskytovatelé pak mohou mít přiděleno 256 adres třídy C, tedy síť třídy B. Existují tedy tři varianty rozsahu IP adres a tedy tři varianty reverzních domén:

1. Je přiděleno 255 adres třídy C (tj. jakoby adresa třídy B, tj. prefix /16). Tato situace není běžná pro běžné uživatele, ale spíše pro poskytovatele Internetu.
2. Je přidělena jedna nebo více adres třídy C (méně než 255 nebo i více než 255, ale netvořících prefix /16).
3. Je přidělen interval IP-adres menší než jedna adresa třídy C.

O delegaci reverzních domén pro sítě třídy B a C se nestarají národní sdružení NIC, ale mezinárodní organizace, které přidělují IP adresy. Pro Evropu delegaci reverzních domén zajišťuje organizace RIPE. Na jmenný server RIPE ns.ripe.net jsou delegovány reverzní domény pro sítě IP adres, které RIPE přiděluje poskytovatelům, tedy např.: 193.in-addr.arpa, 194.in-addr.arpa, 195.in-addr.arpa atd. RIPE pak deleguje reverzní domény pro menší intervaly IP adres – sítě třídy C na jmenné servery poskytovatelů nebo koncových uživatelů.

Reverzní doména, podobně jako klasická doména musí být delegována minimálně na dva jmenné servery. Sekundární jmenný server pro reverzní doménu obvykle zajišťují poskytovatelé na svých jmenných serverech.

Delegace reverzní domény, podobně jako delegace klasické domény sestává z několika kroků:

1. Konfigurace primárního jmenného serveru
2. Konfigurace sekundárního jmenného serveru
3. Delegace reverzní domény
4. Registrace reverzní domény

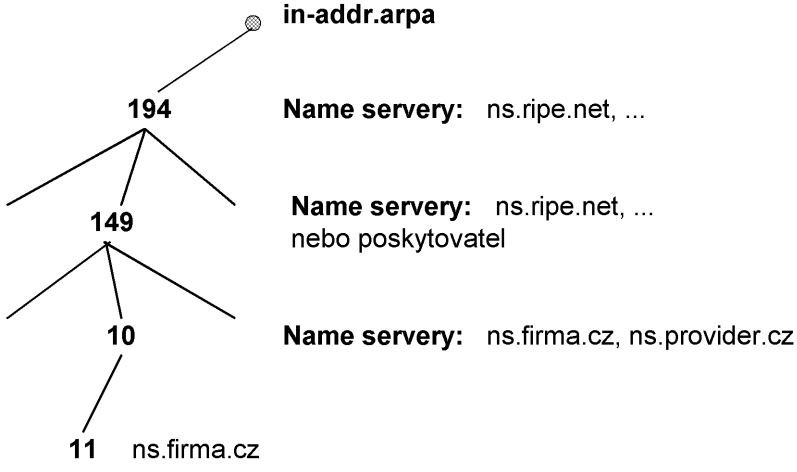
Postup delegace reverzní domény si ukážeme na příkladu.

V příkladu použijeme naši známou společnost Firma s.r.o.

Uživatel Firma s.r.o. používá pro připojení svých počítačů do Internetu síť 194.149.10.0, tedy síť třídy C. Firma s.r.o má vlastní jmenný server na počítači se jménem ns.firma.cz a IP adresou 194.149.10.11. Na uzlu ns.firma.cz je nainstalován OS Unix a BIND 4.9.

Správce společnosti Firma s.r.o musí zajistit delegaci reverzní domény 117.149.194.id-addr.arpa. na jmenný server ns.firma.cz. Sekundární jmenný server pro reverzní doménu bude zajišťovat provider na uzlu ns.provider.net (viz obr. 16.1).

Obr. 16.1
Delegace reverzní domény pro počítač *ns.firma.cz*



Následují jednotlivé konfigurační soubory, respektive jejich části, kterými je zajištěna požadovaná delegace.

Server ns.firma.cz:

Soubor named.boot:

```
…
primary  10.149.194.in-addr.arpa        10.149.194.zone
```

Soubor 10.149.194.zone

```
@               IN      SOA     ns.firma.cz hostmaster.firma.cz (
                                        1998082402      ; Seriál
                                             28800      ; Refresh 8 hours
                                              7200      ; Retry   2 hour
                                            604800      ; Expire  7 days
                                             86400 )    ; Minimum TTL 1 day
            IN      NS          ns.firma.cz.
            IN      NS          ns.provider.net.
11          IN      PTR         ns.firma.cz.
…
```

Server ns.provider.net:

Soubor named.boot:

```
…
secondary       10.149.194.in-addr.arpa 10.149.194.zone       194.149.10.11
```

Server ns.ripe.net *(autoritativní server pro nadřízenou doménu)*:

Soubor 149.194.zone:

```
…
10              IN      NS      ns.firma.cz.
                IN      NS      ns.provider.net.
```

Delegaci domény na fungující jmenné servery musí provést organizace RIPE. O tuto delegaci musí správce požádat správce RIPE pomocí formuláře. Postup s příkladem je uvedený v kapitole 16.1.

Společnost Firma s.r.o má pobočku v Českých Budějovicích. Tato pobočka používá 128 IP-adres, tj. síť 194.149.10.128 – 194.149.10.255. Pobočka v Českých Budějovicích si spravuje svůj vlastní jmenný server se jménem ns.cbu.firma.cz a IP adresou 194.149.10.129. Je tedy vhodné, aby reverzní doména pro subsíť 194.149.10/25 byla delegována na jmenný server ns.cbu.firma.cz.

Tento příklad je v praxi poměrně běžný a my ho využijeme a ukážeme jako ukázku delegace reverzní domény pro subsíť. Nejprve však trochu teorie.

Delegace reverzních domén pro subsítě se nepoužívá od začátku používání DNS. Reverzní domény pro subsítě jsou popsány v RFC-2317 a jsou nazývány "*Classless IN-ADDR.ARPA* delegace". Použitá metoda je kompatibilní s mechanismem DNS a nevyžaduje modifikaci používaného softwaru.

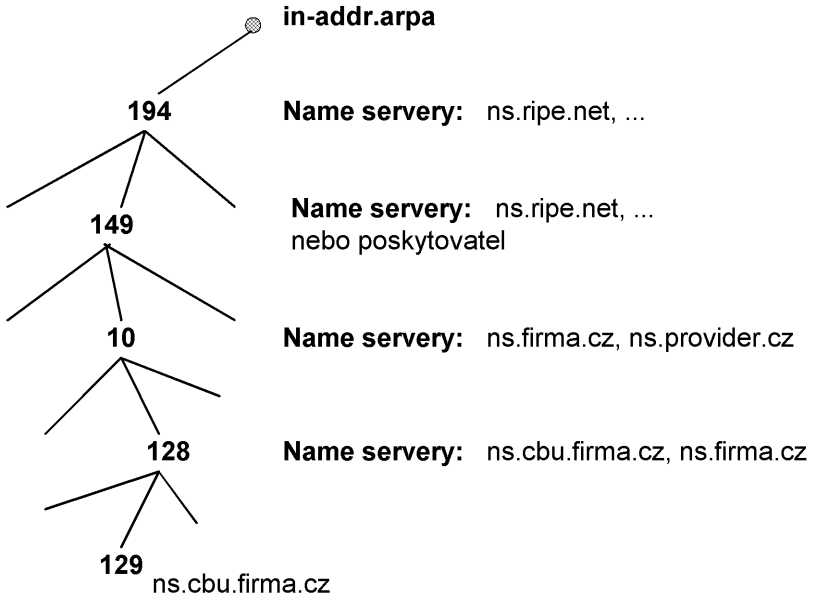
delegace *Classless IN-ADDR.ARPA* řeší nepříjemnou situaci, ke které dříve docházelo. Zákazník, který měl přidělenu subsíť IP adres a měl svůj jmenný server, se totiž dostával do situace, kdy klasickou doménu si spravovat sám a reverzní doménu spravoval poskytovatel. Každé přidání nového záznamu typu A pak s sebou přinášelo nutnost kontaktovat poskytovatele a požádat jej o přidání záznamu PTR do reverzní domény.

Ještě se zamysleme nad označením reverzní domény pro subsíť.

Pokud má zákazník přidělenu síť C 194.149.10, má reverzní doménu 10.149.194.in-addr.arpa. Uzel s IP adresou 194.149.10.11 má pak v reverzní doméně označení 11.10.149.194.in-addr.arpa.

Pokud má zákazník subsíť 194.149.10.128, má reverzní doménu 128.10.149.194.in-addr.arpa. Označení reverzní domény pro subsíť je neobvyklé v tom, že obsahuje čtyři čísla oddělená tečkou, podobně jako IP adresa. Uzel s IP adresou 194.149.10.130 má pak v reverzní doméně označení 130.128.10.149.194.in-addr.arpa, což je ještě neobvyklejší. Jde vlastně o umělou konstrukci, ve které je uplatněn princip vytváření doménového jména. Aby konstrukce byla ještě bizarnější, tak nadřízený jmenný server využije záznam typu PTR ukazující na záznam typu CNAME, které jsou definovány na jmenném serveru nižší úrovně (viz obr. 16.2).

Obr. 16.2
Delegace Classless IN-ADDR.ARPA



Nyní budeme pokračovat v předchozím příkladu.

Následují jednotlivé konfigurační soubory, respektive jejich části, kterými je zajištěna delegace reverzní domény pro subsíť 194.149.10.128. Do konfiguračních souborů z předcházejícího příkladu doplním řádky, které se vztahují k delegaci domény 128.10.149.194.in-addr-arpa, řádky jsou zvýrazněny kurzívou.

Server ns.firma.cz:

Soubor named.boot:

```
primary 10.149.194.in-addr.arpa                          10.149.194.zone
secondary   128.10.149.194.in-addr.arpa   194.149.10.129  128.10.149.194.zone
```

Soubor 10.149.194.zone:

```
@               IN      SOA     ns.firma.cz hostmaster.firma.cz (
                                    1998082402      ; Serial
                                         28800      ; Refresh 8 hours
                                          7200      ; Retry   2 hour
                                        604800      ; Expire  7 days
                                         86400 )    ; Minimum TTL 1 day
            IN          NS          ns.firma.cz.
            IN          NS          ns.provider.cz.
11          IN          PTR         ns.firma.cz.
128         IN          NS          ns.cbu.firma.cz.
            IN          NS          ns.firma.cz.
129         IN          CNAME       129.128.10.149.194.in-addr.arpa.
130         IN          CNAME       130.128.10.149.194.in-addr.arpa.
131         IN          CNAME       131.128.10.149.194.in-addr.arpa.
132         IN          CNAME       132.128.10.149.194.in-addr.arpa.
133         IN          CNAME       133.128.10.149.194.in-addr.arpa.
134         IN          CNAME       134.128.10.149.194.in-addr.arpa.
… Atd. až
254         IN          CNAME       254.128.10.149.194.in-addr.arpa.
```

Server ns.cbu.firma.cz

Soubor named.boot

```
primary         128.10.149.194.in-addr-arpa       128.10.149.194.zone
```

Soubor 128.10.149.194.zone

```
@               IN      SOA     ns.cbu.firma.cz hostmaster.cbu.firma.cz (
                                 1998082502      ; Serial
                                      28800      ; Refresh 8 hours
                                       7200      ; Retry   2 hour
                                     604800      ; Expire  7 days
                                      86400 )    ; Minimum TTL 1 day
            IN          NS          ns.cbu.firma.cz.
            IN          NS          ns.firma.cz.
129         IN          PTR         ns.cbu.firma.cz.
130         IN          PTR         jmeno1.cbu.firma.cz.
131         IN          PTR         jmeno2.cbu.firma.cz.

… atd. až
254         IN          PTR         jmeno.cbu.firma.cz.
```

16.1 Registrace reverzní domény

Registraci můžeme provést až v případě, že máme zcela funkční alespoň dva jmenné servery pro příslušné reverzní domény.

Existují zde dva případy:

1. Registrace reverzní subdomény druhé úrovně.

   Poskytovateli je přidělen blok 255 adres třídy C (jakoby adresa třídy B), pak jmenný server ns.ripe.net obsahuje odkaz (záznamy typu NS) na tento blok („adresu třídy B") ve jmenném serveru poskytovatele. Jmenný server poskytovatele pak obsahuje odkaz (záznamy typu NS) na jmenné servery zákazníků. Jmenný server ns.ripe.net musí v tomto případě být povinně sekundárním jmenným serverem k jmennému serveru poskytovatele pro příslušný blok.

2. Registrace reverzní domény třetí úrovně.

   Poskytovateli je přidělen blok adres menší než 255. Pak jmenný server ns.ripe.net obsahuje záznamy typu NS odkazující přímo na jmenný server zákazníka. Dále se budeme zabývat pouze tímto druhým případem.

Registrace reverzních domén se provádí podle RIPE-185. Formulář obsahuje následující položky:

```
inetnum:viz předchozí odstavec
netname:viz předchozí odstavec
descr:
country:
admin-c:
tech-c:
aut-sys:Číslo AS
rev-srv:počítač s autoritativním jmenným serverem pro reverzní doménu
changed:
source:         RIPE
```

Dále následují záznamy typu NS, které si přejeme vložit do konfiguračního souboru jmenného serveru ns.ripe.net. Stačí uvést věty pro první síť i v případě, že registrujeme interval sítí.

Příklad:

```
From: dostalek@pvt.cz
To: RIPE Hostmaster <auto-inaddr@ripe.net>
Subject: 96 - 111.149.194.in-addr.arpa delegation please
Please delegate 96 - 111.149.194.in-addr.arpa as specified below.
Thank you!
For the PVT Corporation
Libor Dostalek

inetnum:   194.149.96.0 - 194.149.111.255
netname:   PVTNET
descr:     PVT Czech Internet Provider Network
country:   CZ
admin-c:   FD14-RIPE
tech-c:    LD11-RIPE
rev-srv:   ns.pvt.net
```

```
rev-srv:    ns1.pvt.net
changed:    dostalek@pvt.cz 960205
source:     RIPE

96.149              IN      NS      ns.pvt.net.
96.149              IN      NS      ns1.pvt.net.
```

Odpověď je negativní v případě syntaktické chyby nebo chybně nakonfigurovaného jmenného serveru, protože robot auto-inaddr@ripe.net provádí nejen syntaktickou kontrolu, ale i test DNS zákazníka. V případě kladné odpovědi je odpověď zaslána nejenom vám, ale je předána i správci ns.ripe.net, který ručně provede aktualizaci konfiguračního souboru jmenného serveru ns.ripe.net (vloží zaslané věty typu NS). Po provedení změn vám správce pošle mail. Zatímco odezva od robota je do několika minut, tak správce ns.ripe.net vám odpoví asi do týdne.

17

Internet Registry

17.1 Přidělování IP-adres, domén a čísel AS

V síti Internet je potřeba jednoznačně přidělovat IP adresy, čísla autonomních systémů a jména domén. Žádná centrální autorita, která by řídila Internet, však neexistuje. Proto bylo založeno několik mezinárodních organizací, které mají za úkol bdít nad celosvětově jednoznačným přidělováním IP adres, čísel autonomních systémů a doménových jmen.

17.2 Mezinárodní organizace

Mezinárodní organizace jsou z hlediska rozsahu své působnosti uspořádány do stromové struktury. Názorně je struktura organizací patrná z obrázku 17.1.

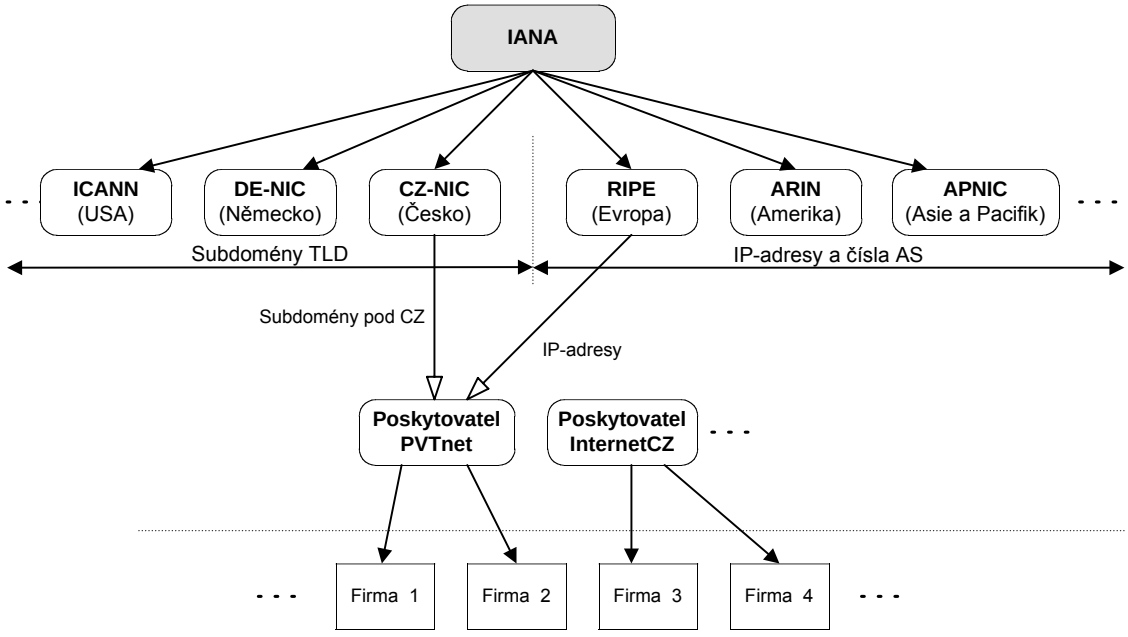
Nejvyšší autoritou v Internetu je The Internet Assigned Numbers Authority (IANA). IANA jednoznačně rozděluje intervaly čísel pro IP-adresy, AS atd. a přiděluje tyto intervaly jednotlivým regionálním IR (Internet Registries). Regionální IR spravují větší oblast (např. kontinent).

Svět je tak geograficky rozdělen mezi regionální IR. Území pokrývané jedním regionálním IR je rozděleno mezi lokální IR. Lokální IR jsou zpravidla poskytovatelé Internetu (ISP).

Jednotlivé regionální IR přidělují IP adresy, doménová jména a čísla autonomních systémů a vedou záznamy o přidělených číslech a subjektech ve svých databázích. Regionální IR komunikují pouze s tzv. lokálními IR (nejčastěji poskytovatelé Internetu), kteří se také finančně podílejí na činnosti regionální IR.

Lokální IR pak obsluhují koncové uživatele. Koncovým uživatelem se nerozumí uživatel PC, ale centrální síťový manager podnikové sítě, který prostřednictvím lokálního IR žádá o přidělení čísel IP-adres, subdomén nebo čísel AS.

Koncový uživatel (tj. správce podnikové sítě) se obrací na lokálního IR (tj. poskytovatele Internetu), ten přesně zformuluje jeho požadavky a zašle je regionální IR. Požadavky se zasílají elektronickou poštou a v regionálním IR je často automaticky zpracovává robot (tj. program). V poslední době se však začíná místo komunikace elektronickou poštou stále více prosazovat zadávání požadavků pomocí webových formulářů.



Obr. 17.1 Hierarchie organizací

Regionální IR jsou:

- ARIN – Amerika blíže viz *http://www.arin.net*
- RIPE NCC (Reseaux IP Europeens Network Coordination Centre) – Evropa. Blíže viz *http://www.ripe.net*
- APNIC – Asijsko-Pacifická oblast. Blíže viz *http://www.apnic.net*

Uvažovalo se i o vytvoření samostatného regionálního IR pro Afriku (AfriNIC), bývalý Sovětský Svaz (RusNIC) a Jižní Ameriku.

Pro nás, obyvatele Evropy je aktuální RIPE NCC.

RIPE pro uživatele z Evropy přiděluje na základě žádosti např. IP adresy. Přidělené IP adresy a další informace eviduje v databázi. Jednotlivá přidělená čísla tvoří tzv. objekty v databázi regionálního IR. Kromě objektů, jako je číslo IP-adresy či číslo AS, jsou v databázi RIPE i objekty popisující osoby zodpovědné za administrativní a technický kontakt, tj. správci sítí. Dále tzv. route objekty popisující směrování mezi AS a např. také mntner objekty autorizující přístup k změně popisu objektů.

Databáze RIPE je veřejně přístupná. Pro čtení informací z databází regionálních IR slouží příkaz whois. Informace je ale možné získat pomocí brány do www, kterou má každý IR zpravidla na svém www serveru. Přímo na www serveru *www.ripe.net* organizace RIPE je také brána do databáze. Dále zpravidla existují i možnosti pomocí ftp, gopher atp.

Regionální IR vytvářejí normy, kterým se musí lokální IR (poskytovatelé) s koncovými uživateli podřídit. RIPE vytváří normy nazývající se RIPE-číslo (např. RIPE-159). Všechny normy RIPE jsou veřejně do-

stupné na ftp://ftp.ripe.net/ripe/docs/. Obdobně APNIC má např. normu APNIC-19, která je opět veřejně dostupná na příslušném serveru APNIC.

17.3 Rozdělení světa mezi IR a kódy zemí podle ISO-3166

Země	doména	IR	Země	doména	IR
AFGHANISTAN	AF	APNIC	LEBANON	LB	RIPE
ALBANIA	AL	RIPE	LESOTHO	LS	ARIN
ALGERIA	DZ	RIPE	LIBERIA	LR	RIPE
AMERICAN SAMOA	AS	APNIC	LIBYAN ARAB JAMAHIRIYA	LY	RIPE
ANDORRA	AD	RIPE	LIECHTENSTEIN	LI	RIPE
ANGOLA	AO	ARIN	LITHUANIA	LT	RIPE
ANGUILLA	AI	ARIN	LUXEMBOURG	LU	RIPE
ANTARCTICA	AQ	ARIN	MACAU	MO	APNIC
ANTIGUA AND BARBUDA	AG	ARIN	MACEDONIA, THE FORMER YUGOSLAV REPUBLIC OF	MK	RIPE
ARGENTINA	AR	ARIN	MADAGASCAR	MG	APNIC
ARMENIA	AM	RIPE	MALAWI	MW	ARIN
ARUBA	AW	ARIN	MALAYSIA	MY	APNIC
AUSTRALIA	AU	APNIC	MALDIVES	MV	APNIC
AUSTRIA	AT	RIPE	MALI	ML	RIPE
AZERBAIJAN	AZ	RIPE	MALTA	MT	RIPE
BAHAMAS	BS	ARIN	MARSHALL ISLANDS	MH	APNIC
BAHRAIN	BH	RIPE	MARTINIQUE	MQ	ARIN
BANGLADESH	BD	APNIC	MAURITANIA	MR	RIPE
BARBADOS	BB	ARIN	MAURITIUS	MU	APNIC
BELARUS	BY	RIPE	MAYOTTE	YT	APNIC
BELGIUM	BE	RIPE	MEXICO	MX	ARIN
BELIZE	BZ	ARIN	MICRONESIA, FEDERATED STATES OF	FM	APNIC
BENIN	BJ	RIPE	MOLDOVA, REPUBLIC OF	MD	RIPE
BERMUDA	BM	ARIN	MONACO	MC	RIPE
BHUTAN	BT	APNIC	MONGOLIA	MN	APNIC
BOLIVIA	BO	ARIN	MONTSERRAT	MS	RIPE
BOSNIA AND HERZEGOWINA	BA	RIPE	MOROCCO	MA	RIPE
BOTSWANA	BW	ARIN	MOZAMBIQUE	MZ	ARIN
BOUVET ISLAND	BV	ARIN	MYANMAR	MM	APNIC
BRAZIL	BR	ARIN	NAMIBIA	NA	ARIN
BRITISH INDIAN OCEAN TERRITORY	IO	APNIC	NAURU	NR	APNIC
BRUNEI DARUSSALAM	BN	APNIC	NEPAL	NP	APNIC
BULGARIA	BG	RIPE	NETHERLANDS	NL	RIPE
BURKINA FASO	BF	RIPE	NETHERLANDS ANTILLES	AN	ARIN
BURUNDI	BI	ARIN	NEW CALEDONIA	NC	APNIC
CAMBODIA	KH	APNIC	NEW ZEALAND	NZ	APNIC
CAMEROON	CM	RIPE	NICARAGUA	NI	ARIN
CANADA	CA	ARIN	NIGER	NE	RIPE
CAPE VERDE	CV	RIPE	NIGERIA	NG	RIPE
CAYMAN ISLANDS	KY	ARIN	NIUE	NU	APNIC
CENTRAL AFRICAN REPUBLIC	CF	RIPE	NORFOLK ISLAND	NF	APNIC
CHAD	TD	RIPE	NORTHERN MARIANA ISLANDS	MP	APNIC
CHILE	CL	ARIN	NORWAY	NO	RIPE
CHINA	CN	APNIC	OMAN	OM	RIPE
CHRISTMAS ISLAND	CX	APNIC	PAKISTAN	PK	APNIC
COCOS (KEELING) ISLANDS	CC	APNIC	PALAU	PW	APNIC
COLOMBIA	CO	ARIN	PANAMA	PA	ARIN

COMOROS	KM	APNIC	PAPUA NEW GUINEA	PG	APNIC
CONGO	CG	ARIN	PARAGUAY	PY	ARIN
CONGO, THE DEMOCRATIC REPUBLIC OF THE	CD	ARIN	PERU	PE	ARIN
COOK ISLANDS	CK	APNIC	PHILIPPINES	PH	APNIC
COSTA RICA	CR	ARIN	PITCAIRN	PN	APNIC
COTE D'IVOIRE	CI	RIPE	POLAND	PL	RIPE
CROATIA (local name: Hrvatska)	HR	RIPE	PORTUGAL	PT	RIPE
CUBA	CU	ARIN	PUERTO RICO	PR	ARIN
CYPRUS	CY	RIPE	QATAR	QA	RIPE
CZECH REPUBLIC	CZ	RIPE	REUNION	RE	APNIC
DENMARK	DK	RIPE	ROMANIA	RO	RIPE
DJIBOUTI	DJ	RIPE	RUSSIAN FEDERATION	RU	RIPE
DOMINICA	DM	ARIN	RWANDA	RW	ARIN
DOMINICAN REPUBLIC	DO	ARIN	SAINT KITTS AND NEVIS	KN	ARIN
EAST TIMOR	TP	APNIC	SAINT LUCIA	LC	ARIN
ECUADOR	EC	ARIN	SAINT VINCENT AND THE GRENADINES	VC	ARIN
EGYPT	EG	RIPE	SAMOA	WS	APNIC
EL SALVADOR	SV	ARIN	SAN MARINO	SM	RIPE
EQUATORIAL GUINEA	GQ	RIPE	SAO TOME AND PRINCIPE	ST	RIPE
ERITREA	ER	RIPE	SAUDI ARABIA	SA	RIPE
ESTONIA	EE	RIPE	SENEGAL	SN	RIPE
ETHIOPIA	ET	RIPE	SEYCHELLES	SC	APNIC
FALKLAND ISLANDS (MALVINAS)	FK	ARIN	SIERRA LEONE	SL	RIPE
FAROE ISLANDS	FO	RIPE	SINGAPORE	SG	APNIC
FIJI	FJ	APNIC	SLOVAKIA (Slovak Republic)	SK	RIPE
FINLAND	FI	RIPE	SLOVENIA	SI	RIPE
FRANCE	FR	RIPE	SOLOMON ISLANDS	SB	APNIC
FRANCE, METROPOLITAN	FX	RIPE	SOMALIA	SO	RIPE
FRENCH GUIANA	GF	ARIN	SOUTH AFRICA	ZA	ARIN
FRENCH POLYNESIA	F	APNIC	SOUTH GEORGIA AND THE SOUTH SANDWICH ISLANDS	GS	ARIN
FRENCH SOUTHERN TERRITORIES	TF	APNIC	SPAIN	ES	RIPE
GABON	GA	RIPE	SRI LANKA	LK	APNIC
GAMBIA	GM	RIPE	ST. HELENA	SH	ARIN
GEORGIA	GE	RIPE	ST. PIERRE AND MIQUELON	PM	ARIN
GERMANY	DE	RIPE	SUDAN	SD	RIPE
GHANA	GH	RIPE	SURINAME	SR	ARIN
GIBRALTAR	GI	RIPE	SVALBARD AND JAN MAYEN ISLANDS	SJ	RIPE
GREECE	GR	RIPE	SWAZILAND	SZ	ARIN
GREENLAND	GL	RIPE	SWEDEN	SE	RIPE
GRENADA	GD	ARIN	SWITZERLAND	CH	RIPE
GUADELOUPE	GP	ARIN	SYRIAN ARAB REPUBLIC	SY	RIPE
GUAM	GU	APNIC	TAIWAN, PROVINCE OF CHINA	TW	APNIC
GUATEMALA	GT	ARIN	TAJIKISTAN	TJ	RIPE
GUINEA	GN	RIPE	TANZANIA, UNITED REPUBLIC OF	TZ	ARIN
GUINEA-BISSAU	GW	RIPE	THAILAND	TH	APNIC
GUYANA	GY	ARIN	TOGO	TG	RIPE
HAITI	HT	ARIN	TOKELAU	TK	APNIC
HEARD AND MC DONALD ISLANDS	HM	ARIN	TONGA	TO	APNIC
HOLY SEE (VATICAN CITY STATE)	VA	RIPE	TRINIDAD AND TOBAGO	TT	ARIN
HONDURAS	HN	ARIN	TUNISIA	TN	RIPE
HONG KONG	HK	APNIC	TURKEY	TR	RIPE
HUNGARY	HU	RIPE	TURKMENISTAN	TM	RIPE

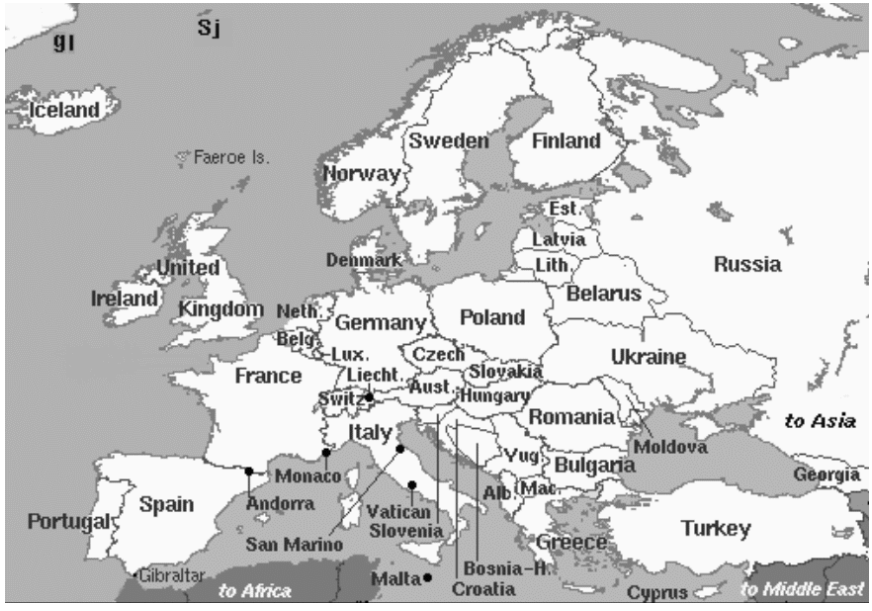
ICELAND	IS	RIPE	TURKS AND CAICOS ISLANDS	TC	ARIN
INDIA	IN	APNIC	TUVALU	TV	APNIC
INDONESIA	ID	APNIC	UGANDA	UG	RIPE
IRAN (ISLAMIC REPUBLIC OF)	IR	RIPE	UKRAINE	UA	RIPE
IRAQ	IQ	RIPE	UNITED ARAB EMIRATES	AE	RIPE
IRELAND	IE	RIPE	UNITED KINGDOM	GB	RIPE
ISRAEL	IL	RIPE	UNITED STATES	US	ARIN
ITALY	IT	RIPE	UNITED STATES MINOR OUTLYING ISLANDS	UM	ARIN
JAMAICA	JM	ARIN	URUGUAY	UY	ARIN
JAPAN	JP	APNIC	UZBEKISTAN	UZ	RIPE
JORDAN	JO	RIPE	VANUATU	VU	APNIC
KAZAKHSTAN	KZ	RIPE	VENEZUELA	VE	ARIN
KENYA	KE	RIPE	VIET NAM	VN	APNIC
KIRIBATI	KI	APNIC	VIRGIN ISLANDS (BRITISH)	VG	ARIN
KOREA, DEMOCRATIC PEOPLE'S REPUBLIC OF	KP	APNIC	VIRGIN ISLANDS (U.S.)	VI	ARIN
KOREA, REPUBLIC OF	KR	APNIC	WALLIS AND FUTUNA ISLANDS	WF	APNIC
KUWAIT	KW	RIPE	WESTERN SAHARA	EH	RIPE
KYRGYZSTAN	KG	RIPE	YEMEN	YE	RIPE
LAO PEOPLE'S DEMOCRATIC REPUBLIC	LA	APNIC	YUGOSLAVIA	YU	RIPE
LATVIA	LV	RIPE	ZAMBIA	ZM	ARIN
			ZIMBABWE	ZW	ARIN

Zdroj informací: http://www.ripe.net.

Obr. 17.2
Asie

Obr. 17.3
Afrika

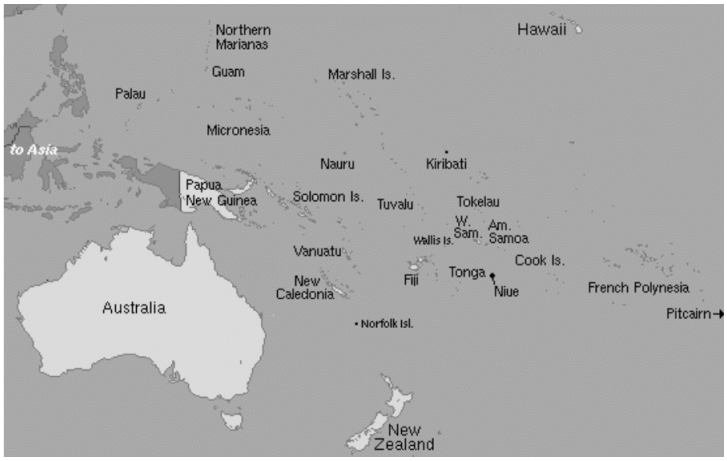
Obr. 17.3
Evropa



Obr. 17.5
Severní Amerika

Obr. 17.6
Jižní Amerika

Obr. 17.7
Austrálie



Obr. 17.8
Karibská oblast

Evropští IR spravující TLD vytvořili společný orgán pod názvem CENTR *(COUNCIL OF EUROPEAN NATIONAL TOP-LEVEL DOMAIN REGISTRIES)*. Bližší informace viz http://www.centr.org

17.4 IP-adresy

17.4.1 RFC1466 – bloky adres třídy C

Rozdělování IP-adres mezi regiony řeší RFC1466. Novější informace o rozdělení IP adres mezi jednotlivé regionální IR, tj. mezi jednotlivé části světa můžeme najít na ftp://ftp.isi.edu/in-notes/iana/assignments/ipv4-address-space (resp. přes http://www.iana.org). První blok byl prakticky vyčerpán. Ostatní bloky pak přiděluje IANA jednotlivým regionům, aby bylo mj. možné agregovat IP-adresy i z hlediska směrování následovně:

```
Multi-regional          192.0.0.0 - 192.255.255.255
Europe                  193.0.0.0 - 195.255.255.255
Various Registeries     196.0.0.0 - 196.255.255.255
IANA-Reserved           197.0.0.0 - 197.255.255.255
Various Registeries     198.0.0.0 - 198.255.255.255
```

```
North America           199.0.0.0 - 199.255.255.255
Central/South America   200.0.0.0 - 201.255.255.255
Pacific Rim             202.0.0.0 - 203.255.255.255
North America           204.0.0.0 - 209.255.255.255
Pacific Rim             210.0.0.0 - 211.255.255.255
Europe                  212.0.0.0 - 213.255.255.255
US-DOD                  214.0.0.0 - 215.255.255.255
North America           216.0.0.0 - 216.255.255.255
Reserved                217.0.0.0 - 223.255.255.255
```

IP-adresy určené pro uzavřené sítě (Intranet)

Tyto adresy specifikuje RFC-1918:

```
10.0.0.0        -       10.255.255.255
172.16.0.0      -       172.31.255.255
192.168.0.0     -       192.168.255.255
```

17.4.2 AS

Pro uzavřené sítě jsou rezervovány podle RFC-1930 čísla AS v intervalu 64512 až 65535.

17.5 RIPE

RIPE přiděluje v Evropě čísla AS a IP-adresy, spravuje reverzní domény k přiděleným IP-adresám a v neposlední řadě vede příslušné databáze. Formuláře, normy a další užitečné informace pro komunikaci s RIPE je možné stáhnout z anonymního ftp-serveru ftp.ripe.net.

Poskytovatel Internetu se nejprve musí stát lokálním IR, pak může žádat RIPE o přidělování IP-adres, čísel AS, registraci svých reverzních domén na jmenném serveru RIPE pro sebe a své zákazníky. Je povinen udržovat aktuální data v databázích RIPE. Norma RIPE-185 tvoří příručku popisující spolupráci s RIPE.

17.5.1 Internet Registry

Abyste se mohli stát poskytovatelem Internetu, budete potřebovat mít možnost komunikovat s RIPE. RIPE však přijímá požadavky pouze od lokálních IR. Je třeba se tedy stát lokálním IR (vše pochopitelně až po získání licence pro podnikání v příslušném oboru v České republice).

17.5.2 Registrace lokálního IR

Jak se stát lokálním IR popisuje dokument RIPE-160. Ustavení nového lokálního IR sestává ze tří kroků:

- **Založení položky o lokálním IR v seznamu lokálních IR v RIPE.**
- **Seznámení se s registračními procedurami.**
- **Uzavření hospodářské smlouvy, aby RIPE začalo zasílat faktury.**

Komunikace s RIPE spojená s ustavením nového lokálního IR probíhá pomocí e-mailu.

17.5.3 Založení položky o lokálním IR v seznamu lokálních IR

Tuto problematiku řeší norma RIPE-160. Nejprve musíte shromáždit informace o své firmě, která se chce stál lokálním IR, vyplnit formulář a odeslat ho na adresu new-lir@ripe.net.

Formulář pro registraci lokálního IR:

```
———————— cut here ————————

                regid:
                org:
                type:
                address:
                country:
                admin-c:
                tech-c:
                phone:
                fax-no:
                e-mail:
                remark:
                lst-localir:
                lst-provs:
                lst-contrib:
                bill-addr:
                bill-mail:
                bill-ref:
                bill-vatno:
                bill-proto:
                bill-categ:
                bill-scheme:
                bill-remark:

                Questions:

                1. Does the organisation already operate a Local
                Internet Registry in the RIPE NCC region? If yes,
                please give us the regid(s) of the Registry and a
                short explanation why an additional Registry needs
                to be openned.


                2. Does the organisation have another registry in a
                different region (APNIC or ARIN region)? If yes,
                please specify the region and explain why you also
                need to set up a Local Registry in the RIPE NCC
                region.


                3. Does the organisation already have address space?
                Please list it here. Please only list address space
                being used for the organisation's own internal net-
                work, and not address space being used for customers
                (other than dial-up or web servers). Please include
                address space for the entire organisation, not just
                this subsidiary.


                ———————— cut here ————————-
```

Formulář obsahuje následující klíčová slova:

```
regid:          Identifikace IR. Skládá se z kódu státu a identifikace IR.  Např. cz.pvtnet.
org:            Jméno organizace, která se chce stát lokálním IR.
type:           Typ lokálního IR, např. PROVIDER nebo SPONSOR atp.
address:Adresa šnečí (snail) pošty.
country:Stát.
admin-c:Kontaktní osoba pro administrativní kontakt. Preferuje se identifikační číslo (han-
        dle).
                Při prvním výskytu kontaktní osoby se uvede jméno a příjmeni. V takovém přípa-
                dě musí za popisem objektu následovat popis objektu kontaktní osoby.
tech-c:         Kontaktní osoba pro technický kontakt (blíže viz admin-c).
phone:          Telefonní číslo. Skládá se z čísla státu uvozeného znakem plus, mezery, kódu
                města, mezery a místního čísla. Při volbě pomocí operátorky dodáme mezeru, ře-
                tězec ext, mezeru a klapku. Příklad +42 38 827111 ext 361
fax-no:         Číslo faxu - viz phone.
mail:           e-mail lokálního IR.
remarks:Poznámka
lst-localir:    Mailová adresa, která bude přidána do konference zabývající se technickými
                a procedurálními problémy lokálních IR.
lst-provs:      Mailová adresa, která bude přidána do konference zabývající požadavky poskyto-
                vatelů na RIPE.
lst-contrib:    Mailová adresa, která bude přidána do konference přispěvatelů RIPE.
bill-addr:      Šnečí adresa, na kterou se budou posílat účty.
bill-mail:      Mailová adresa pro případ, že se faktury posílají mailem.
bill-ref:       Text, který má být přidán na fakturu.
bill-vatno:     DIČ (v ES).
bill-proto:     E-MAIL (faktura mailem) nebo SNAIL-MAIL (fakturace šnečí poštou).
bill-categ:     Velikost lokálního IR - na ní závisí velikost příspěvku (LARGE, MEDIUM, SMALL,
                atp).
bill-scheme:    Interval platby (YEARLY,HALF-YEARLY, QUARTERLY)
bill-remarks:   Poznámky týkající se platby.
```

Je-li v položce admin-c nebo tech-c uvedeno jméno a příjmení (nikoliv identifikační číslo – handle), pak následuje pro každou takovou osobu popis objektu této osoby. Takový objekt má následující položky (blíže viz RIPE-119):

```
person:         Jméno (i více) a příjmení. Neuvádí se tituly. Nesmí se vyskytovat tečky:
                Příklad:        Daniel F Karrenberg
address:Šnečí adresa.
phone:          Telefonní číslo.
fax-no:         Fax.
e-mail:         Elektronická pošta.
nic-hdl:Identifikační číslo (handle).
remarks:Poznámka.
changed:Změnil ve formátu mail osoby, která změnu provedla a datum změny tvaru RRMMDD.
                Příklad:changed:       dostalek@pvt.cz   960401
source:         RIPE (tento řádek je povinný a nemá jinou alternativu).
```

Místo komunikace elektronickou poštou lze použít i formulář http://www.ripe.net/ripencc/new-mem/newlir-form.html.

Seznam lokálních IR a základní informace o nich jsou vystaveny na: http://www.ripe.net/lir/registries/indices/index.html.

RIPE provedlo registraci lokálního IR.

O registraci lokálního IR se dozvíte mailem, jehož příklad následuje:

```
From billing@ripe.net Mon Nov 20 11:56:03 1995
…

We have added

        cz.pvtnet

to the list of Local Internet Registries with the following particulars:

regid:       cz.pvtnet
org:         PVT a.s.
type:        PROVIDER
community:   „We will serve customers of PVT a.s., an ISP in CZ.“
community:   „We are ready to serve to all requests from Czech Republic.“
address:     PVT a.s
address:     Kovanecka 2124
address:     190 00 Praha
address:     Czech Republic
country:     CZ
admin-c:     Boris Belousov
admin-c:     Frantisek Drdak
tech-c:      Boris Belousov
tech-c:      Libor Dostalek
phone:       +42 2 6849 122 ext 246
fax-no:      +42 2 6849 279
e-mail:      registr@pvt.cz
remark:
lst-localir: registr@pvt.cz
lst-provs:   provider@pvt.cz
lst-contrib: belousov@pvt.cz
bill-addr:   PVT a.s.
bill-addr:   PVT a.s
bill-addr:   Kovanecka 2124
bill-addr:   190 00 Praha
bill-addr:   Czech Republic
bill-mail:   belousov@pvt.cz
bill-ref:
bill-vatno:
bill-proto:  E-MAIL
bill-categ:  SMALL
bill-scheme: YEARLY
bill-remark:
reg-ack:

The public part of this has been published in …
```

Získali jsme tak identifikaci *cz.pvtnet*. Tato identifikace je ve tvaru *nn.provider*, kde *nn* je kód země a *provider* je zkratka specifikující lokálního IR. Při komunikaci s RIPE je v některých případech nutné

tuto identifikaci uvádět (např. vždy při komunikaci s hostmaster@ripe.net). Provede se to tak, že na počátek zprávy se uvede hlavička:

```
X-NCC-RegID: nn.provider
```

Uzavření hospodářské smlouvy

Aktuální vzor hospodářské smlouvy je uveřejněn v normě RIPE. Tč. je aktuální normou RIPE-191 pod názvem „The Standard RIPE NCC Service Agreement“. Tento vzor si stáhněte z: *ftp://ftp.ripe.net/ripe/docs/ripe-191.ps* a vytiskněte.

Hospodářská smlouva se musí vyhotovit dvakrát a poslat běžnou poštou do Amsterodamu. Jeden podepsaný exemplář vám RIPE stejným způsobem vrátí. Po obdržení podepsané hospodářské smlouvy vám RIPE zašle fakturu. Jakmile ji zaplatíte, můžete začít fungovat jako lokální IR.

Poplatky jsou různé podle billing kategorie (velikosti poskytovatele Internetu), kterou jste si zvolili. Výše poplatků a jejich stanovení je popsáno v dokumentu příslušné normě RIPE, např. pro rok 1999 je v normě RIPE-188 mimo jiné uvedeno:

```
1.   The 1999 fee schedule is as follows:


Enterprise registries:         ECU 2650.- / year

Service Provider Registries:

Small                          ECU 2650.- / year
Medium                         ECU 3700.- / year
Large                          ECU 4900.- / year
Supernational                  Multiple times large registry
                               Please contact <billing@ripe.net>

Start-up fee for registries
established during 1999        ECU  2100.-


2.   Registries established during the course of the
     year are charged as follows:

     +-------------------------------------------------------+
     |established during   charge                            |
     |                                                       |
     |1st quarter          start-up fee + full yearly fee    |
     |2nd quarter          start-up fee + 3/4 yearly fee     |
     |3rd quarter          start-up fee + 1/2 yearly fee     |
     |4th quarter          start-up fee + 1/4 yearly fee     |
     +-------------------------------------------------------+

3.   The start-up fee is due with the first invoice.

…

If you have any questions, please contact
<billing@ripe.net>.
```

Otázky k placení a účtům můžete posílat na adresu billing@ripe.net. Startujte vždy jako malý (small) poskytovatel.

Jako nový poskytovatel Internetu se seznamte zejména s normou popisující jednotlivé pracovní postupy, tč. je aktuální normou norma RIPE-185.

Jako nový lokální IR potřebujete nejprve:

- Získat (alokovat) interval IP pro svou vlastní síť a své zákazníky.
- Dostat přiděleno číslo autonomního systému.
- Registrovat objekt Route, který popisuje routing mezi jednotlivými AS a sítí.

17.5.4 Databáze RIPE

Databáze RIPE jsou určeny pro uschovávání informací o evropských objektech Internetu. V databázi jsou tedy informace o přidělených intervalech IP adres, doménách, route objektech atd. Všechny objekty spravují nějací lidé, takže jsou v databázi informace i o těchto lidech, o tzv. kontaktních osobách. Informace jsou v databázi RIPE udržovány ve tvaru objektů různých typů. Mezi nejčastěji používané objekty patří objekt inetnum, objekt domain, objekt person, objekt role a objekt aut-num.

Objekt inetnum

Objekt inetnum popisuje interval přidělených IP adres. V objektu je uveden rozsah IP adres, jméno uživatele nebo název organizace, které byly IP adresy přiděleny, označení kontaktní osoby, tj. osoby, kterou je možno kontaktovat v případě problému s IP adresou z daného rozsahu. Kontaktní osoby jsou uvedeny pomocí jednoznačného identifikátoru, který mají přidělený. Tento jednoznačný identifikátor se nazývá RIPE-handle.

Příklad:

```
inetnum:     194.149.96.0 - 194.149.111.255
netname:     PVTNET
descr:       PVT Czech Internet Provider Network
country:     CZ
admin-c:     HP1171-RIPE
tech-c:      HP1171-RIPE
rev-srv:     ns.pvt.net
rev-srv:     ns1.pvt.net
status:      ASSIGNED PA
changed:     dostalek@pvt.cz 19960524
changed:     kabelova@pvt.cz 19990111
source:      RIPE
```

Je třeba si uvědomit, že obdobné objekty jsou i v databázích jiných regionálních IR. Např.:

```
inetnum:     203.0.0.0 - 203.63.255.255
netname:     AUNIC-AU
descr:       Telstra Internet
descr:       5/490 North Bourne Av.
descr:       Dixson
descr:       Canberra ACT 2903
country:     AU
admin-c:     GH105
```

```
tech-c:      GH105
remarks:     national nic
notify:      dbmon@apnic.net
mnt-by:      MAINT-APNIC-AP
changed:     paulg@apnic.net 19981029
source:      APNIC
```

Objekt domain

Objekt domain popisuje doménu. Objekt obsahuje jméno domény, informace o uživateli, který registroval doménu, včetně kontaktních osob uživatele a jmenných serverů pro doménu.

Příklad:

```
domain:      pvt.cz
descr:       PVT, Ltd.
descr:       Ceske Budejovice
admin-c:     LD3-RIPE
tech-c:      PB25-RIPE
zone-c:      LD3-RIPE
nserver:     ns.pvt.net ns1.pvt.net snmp0.pvt.net ns0.pipex.net ns1.pipex.net
mnt-by:      CZ-DOMREG
changed:     ors@Czechia.EU.net 981114
source:      RIPE
```


Objekty person a role

Kontaktní osoby se registrují pomocí objektu person. Objekt person popisuje konkrétní kontaktní osobu. Objekt obsahuje adresu, telefonní číslo, faxové číslo a e-mail kontaktní osoby.

Objekt role definuje jednu nebo více kontaktních osob, které zastávají určitou funkci v organizaci, např. funkci správce. V objektu role jsou uvedeny konkrétní osoby, které funkci zastávají, pomocí RIPE-handlu. Pokud danou funkci začne zastávat nový pracovník, stačí zaktualizovat objekt role.

V ostatních objektech je možné uvádět kontaktní osoby registrované jak pomocí objektu person, tak pomocí objektu role.

Příklad:

```
role:        Hostmaster PVTNET
address:     Praha
e-mail:      hostmaster@pvt.net
admin-c:     PS2771-RIPE
admin-c:     JD1802-RIPE
admin-c:     AK291-RIPE
tech-c:      PS2771-RIPE
nic-hdl:     HP1171-RIPE
notify:      hm-dbm-msgs@ripe.net
changed:     kabelova@pvt.cz 19990122
source:      RIPE

person:      Petr Svihalek
address:     PVT a.s.
```

```
address:     Podvinny mlyn 2178/6
address:     Praha 9
address:     190 00
address:     Czech Republic
phone:       +420 2 66198432
fax-no:     +420 2 66198678
e-mail:     svihalek@pvt.net
nic-hdl:    PS2771-RIPE
changed:    svihalek@pvt.net 19990211
source:     RIPE
```

Objekt aut-num

Objekt aut-num popisuje autonomní systém a zejména popisuje směrování do sousedních autonomních systémů.

Příklad:

```
aut-num:    AS5490
descr:      PVTNET
descr:      PVT public IP network
descr:      Czech Republic
as-in:      from AS701 100 accept ANY
as-in:      from AS702 100 accept ANY
as-in:      from AS8593 100 accept ANY
as-in:      from AS2819 100 accept AS2819
as-in:      from AS6881 100 accept AS1902 AS2605 AS2686 AS2852 AS5407
as-in:      from AS6881 100 accept AS5578
as-in:      from AS6881 100 accept AS5588 AS5620 AS6706 AS6721 AS6740
as-in:      from AS6881 100 accept AS6881
as-in:      from AS6881 100 accept AS8248 AS8316 AS8593 AS8679 AS8747
as-in:      from AS6881 100 accept AS8913
as-out:     to AS2819 announce AS5490
as-out:     to AS701 announce AS5490
as-out:     to AS702 announce AS5490
as-out:     to AS8593 announce AS5490
as-out:     to AS6881 announce AS5490
default:    AS701 100
admin-c:    JD1802-RIPE
admin-c:    PS2771-RIPE
tech-c:     MM107-RIPE
tech-c:     IV15-RIPE
notify:     ripe-notify@pvt.net
mnt-by:     ASPVT-MNT
changed:    hostmaster@pvt.net 19990127
source:     RIPE
```

Objekt route

Objekt route sděluje, do jakého autonomního systému patří určitý interval IP-adres.

Příklad:

```
route:      194.149.96.0/19
descr:      PVTNET
origin:     AS5490
mnt-by:     ASPVT-MNT
changed:    drdak@pvt.net 970403
source:     RIPE
```

Objekt mntner

Objekt mntner umožňuje autentizaci při správě objektu, tj. umožňuje, aby objekt v databázi RIPE nemohl opravovat kdokoliv, ale pouze autentizovaná osoba.

Příklad:

```
mntner:     DOSTALEK
descr:      Libor Dostalek
admin-c:    LD3-RIPE
tech-c:     LD3-RIPE
upd-to:     dostalek@pvt.cz
mnt-nfy:    dostalek@pvt.cz
auth:       CRYPT-PW dCGQH1DfjU4to
mnt-by:     DOSTALEK
changed:    dostalek@pvt.cz 960410
changed:    kabelova@pvt.cz 980604
source:     RIPE
```


17.5.4.1 Jak si prohlížet obsah databáze

K prohlížení databází je určen program whois, který je možné získat na adrese *ftp://ftp.ripe.net/tools* (má několik parametrů, které neuvádíme).

V příkladu požaduji informace z RIPE databáze na adrese whois.ripe.net o objektu inetnum 194.149.101.0. Parametr -h uvozuje jméno počítače s whois-serverem, který má přístup do RIPE databáze.

Příklad:

```
whois -h whois.ripe.net 194.149.101.0
...

inetnum:     194.149.96.0 - 194.149.111.255
netname:     PVTNET
descr:       PVT Czech Internet Provider Network
country:     CZ
…
```

Příkaz vypisuje s objektem i informace o kontaktních osobách, které jsou vlastně samostatnými objekty. Kontaktní osoby jsou v objektech ostatních typů uvedeny zkratkou ve tvaru XXn-RIPE, kde XX je monogram kontaktní osoby a n je jednoznačné číslo v rámci stejných monogramů. Tato zkratka se nazývá RIPE-handle. Jednoznačnost zajišťuje sama databáze. Uživatel vkládající objekt person do databáze použije v novém objektu ripe-handle AUTO-1XX a vlastní databáze pak tento manipulátor (handle) nahradí jednoznačným manipulátorem. V objektu person je ripe-handle uveden v řádku nic-hdl.

Příkazem whois můžeme získat informace o všech objektech, které mají za klíčový údaj zadaný řetězec. Klíčovým údajem je: aut-num, inetnum, netname, handle, jméno kontaktní osoby, příjmení kontaktní osoby, jméno a příjmení současně, jméno domény nebo jméno subdomény (pouze domény prvního a druhého řádu, tj. např. cz nebo pvt.cz).

Příklad:

```
whois -h whois.ripe.net pvt.cz
...

domain:      pvt.cz
descr:       PVT, Ltd.
descr:       Ceske Budejovice
admin-c:     LD3-RIPE
tech-c:      PB25-RIPE
zone-c:      LD3-RIPE
nserver:     mh.pvt.cz ns.eunet.cz
changed:     ors@Czechia.EU.net 960402
source:      RIPE

... dále následují informace o kontaktních osobách
```

Program whois není příliš rozšířen, ale jelikož používá protokol telnet na portu 43, tak se lze bez něj i obejít a použít příkaz telnet přímo:

```
telnet whois.ripe.net 43
řetězec
```

Jako řetězec je dobré si zadat help a dostaneme popis celé syntaxe. Běžně se však ani tato možnost nepoužívá.

Používá se gateway do www.

Gateway do www lze nalézt na *http://www.ripe.net*. Obdobně se komunikuje i s InterNIC.

17.5.5 Komunikace s RIPE

S RIPE komunikujeme zásadně mailem. Své požadavky odesíláme na příslušné mailové adresy do RIPE. Na počátku subjectu odpovědi RIPE vrací identifikaci požadavku ve tvaru NCC#číslo (např. NCC#1234567). Toto číslo pak opakuje ve všech dalších odpovědích na tento požadavek. Pomocí této identifikace lze pak i požadavek reklamovat u personálu RIPE. Při reklamaci opět tuto identifikaci uvádíme v předmětu reklamačního mailu.

Více odpovědí je běžných zejména v případě, kdy požadavek nejprve kontroluje robot a pak vyřizuje personál RIPE. V takovém případě pak dostaneme první odpověď od robota a následnou od personálu.

17.5.5.1 Oprava databáze

Požadavky na vložení objektu do databáze, jeho opravu nebo zrušení se posílají e-mailem. Objekty může vkládat, opravovat a rušit pouze oprávněná osoba. Požadavky na opravu databáze RIPE se posílají na adresu auto-dbm@ripe.net. Na této adrese čeká robot. Pokud je váš požadavek formálně správně a jste oprávněni modifikovat databázi, pak robot váš požadavek vyřídí a zprávu o vyřízení pošle zpět na vaši adresu. Pokud se vám podaří v požadavku vyrobit chybu, dostanete mailem zprávu o odmít-

nutí požadavku. Výhodné je v mailu s požadavkem uvést jako subjekt řetězec LONGACK, který zajistí podrobný výpis v odpovědi důležitý hlavně při výskytu chyby.

17.5.5.2 Jak vložit objekt do databáze

Nejprve si obstaráte prázdný formulář popisující objekt. Prázdné formuláře jsou popsány v jednotlivých normách RIPE (např. RIPE-159 obsahuje formulář pro registraci reverzní domény).

Formulář se skládá z klíčových slov ukončených dvojtečkou. Za klíčové slovo doplníme hodnotu.

Jako příklad uvedu formulář pro vložení objektu inetnum:

```
inetnum:
netname:
descr:
descr:
country:
admin-c:

tech-c:
notify:
changed:
source:      RIPE
```

Inspiraci pro vyplnění najdete v některém existujícím objektu v databázi.

Prázdný formulář vyplníte a odešlete ke zpracování na adresu *auto-dbm@ripe.net*. Odpověď se vrátí během několika minut. Odpověď je buď záporná, pokud jste udělali nějakou chybu, nebo kladná, pak máte úspěch a databáze je aktualizována.

Příklad komunikace:

Požadavek:

```
Subject:  LONGACK
   Date: Thu, 10 Sep 1998 09:32:06 +0100
   From: Alena Kabelova <kabelova@pvt.cz>
    To: auto-dbm@ripe.net


inetnum:  195.47.38.16 - 195.47.38.31
netname:  DIGITIS
descr:    Digitis a.s.
descr:    Skalova 2490
descr:    Tabor
descr:    390 02
country:  cz
admin-c:  AUTO-1LJ
tech-c:   AUTO-1LJ
status:   ASSIGNED PA
changed:  kabelova@pvt.cz 980910
source:   RIPE

person:   Libor Jinda
address:  Skalova 2490
address:  390 02 Tabor
```

```
phone:    + 420 361 281685
fax-no:   + 420 361 281634
nic-hdl:  AUTO-1LJ
changed:  kabelova@pvt.cz 980910
source:   RIPE

person:   Tomas Sevcik
address:   Hroznova 2
address:   656 06 Brno
phone:     +420 05 43321304
fax-no:    +420 05 43211148
e-mail:    sevcik@zeus.cz
nic-hdl:   AUTO-2TS
changed:   kabelova@pvt.cz 980910
source:    RIPE
```

Odpověď:

```
Subject: SUCCEEDED: LONGACK
   Date: 10 Sep 1998 07:33:21 -0000
   From:  RIPE Database Management <ripe-dbm@ripe.net>
    To:  Alena Kabelova <kabelova@pvt.cz>


Your update was SUCCESSFUL, but it can take
10 minutes before your update is visible
using a whois look-up.

>   From:    Alena Kabelova <kabelova@pvt.cz>
>   Subject: LONGACK
>   Date:    Thu, 10 Sep 1998 09:32:06 +0100
>   Msg-Id:  <35F78E86.A60E23A7@pvt.cz>

The following objects were processed.

New OK: [person] LJ460-RIPE (Libor Jinda)

New OK: [person] TS2088-RIPE (Tomas Sevcik)

New OK: [inetnum] 195.47.38.16 - 195.47.38.31
```

V příkladu si všimněte dvou věcí. Jedním mailem je možné vkládat, opravovat nebo rušit několik objektů. Pokud jedním mailem vkládám více než jednu kontaktní osobu, zvyšuji číslo v dočasném nic-handle. Použil jsem tedy AUTO-1LJ a AUTO-2TS.

17.5.5.3 Jak modifikovat objekt v databázi

Pokud chcete modifikovat objekt v databázi, pak je výhodné nejprve vypsat z databáze objekt původní, tento objekt opravit a opravený opět odeslat na *auto-dbm@ripe.net* se subjektem LONGACK. A zase čekáte na odpověď. Je třeba si uvědomit, že se nedají opravovat klíčové údaje. Např. si firma změní jméno domény, není možné objekt stávající domény opravovat. Při změně jmen objektů musíte postupovat tak,

že nejprve zrušíte stávající objekt a poté vložíte objekt nový. Nicméně k sestavení nového objektu můžete výhodně použít starý objekt vypsaný z databáze.

17.5.5.4 Jak zrušit objekt v databázi

Zrušení objektu v databázi je obdobné modifikaci. Nejprve si vypíšeme do souboru popis starého objektu. A za popis připíšeme řádek:

```
delete: moje_mailová_adresa [proč]
```

kde napíšeme svou e-mailovou adresu a krátce můžeme popsat důvod rušení. No a soubor opět pošleme na auto-dbm@ripe.net.

17.5.6 Přidělování IP adres, autonomních systémů a registrace reverzních domén

V těchto případech je komunikace s RIPE zpravidla dvoustupňová:

1. Zašlete vyplněný formulář s požadavkem, který je zkontrolován robotem (programem).
2. Po formální kontrole jsou tyto požadavky předány ke zpracování personálu. O provedené (nebo zamítnuté) akci jste opět mailem informováni. Je třeba zdůraznit, že robot odpovídá řádově v minutách nebo hodinách, odpověď od personálu obdržíte v několika dnech.

17.5.7 Přidělení čísla autonomního systému

O přidělení čísla AS se žádá podle RIPE-147 na formuláři (popisuji jen položky, které se nevyskytly v předchozím formuláři):

```
X-NCC-RegID:


        #[AUT-NUM TEMPLATE]#

        aut-num:O kolik AS žádáme
        descr: Krátký popis AS
        descr:
        as-in:Popis směrovací informace, která se přijímá od souseda (blíže viz RIPE-147)
        as-in:
        as-out:Popis směrovací informace, kterou předáváme sousedovi (blíže viz RIPE-147)
        as-out:
        default:Default routing
        admin-c:
        tech-c:
        remarks:
        mnt-by:
        changed:
        source: RIPE
        #[MAINTAINER TEMPLATE]#

        mntner:
        descr:
        admin-c:
```

```
tech-c:
upd-to:
auth:
mnt-by:
changed:
source: RIPE
```

Je-li v položce admin-c nebo tech-c uvedeno jméno a příjmení (nikoliv identifikační číslo – handle), pak následuje pro každou takovou osobu popis objektu této osoby.

```
#[PERSON TEMPLATE]#

   person:
   address:
   address:
   phone:
   fax-no:
   e-mail:
   nic-hdl:
   remarks:
   notify:
   changed:
   source:   RIPE

   #[TEMPLATE END]#
```

Vyplněný formulář se zašle e-mailem na adresu hostmaster@ripe.net.

Příklad požadavku následuje (položka aut-num vyjadřuje počet požadovaných AS):

```
From: „Ing. Boris Belousov“ <belousov@pvt.cz
To: hostmaster@ripe.net
Subject: cz.pvtnet AS request

X-NCC-RegID: cz.pvtnet

aut-num: 1
descr:   PVTNET
descr:   PVT public IP network
descr:   Czech Republic
tech-c:  LD11-RIPE
tech-c:  BB31-RIPE
admin-c: BB31-RIPE
admin-c: FD14-RIPE
notify:  registr@pvt.cz
mnt-by:  ASPVT-MNT
changed: belousov@pvt.cz 951124
source:  RIPE

as-in:    AS1849 100 accept ANY
as-out:   AS1849 announce ANY
default:  AS1849 100
```

Jelikož objekt AS musí být chráněn pomocí autorizace, tak je nutné v případě zavedení objektu mntner jej i popsat (blíže viz dále):

```
mntner:     ASPVT-MNT
descr:      PVT a.s.
descr:      Internet Service Provider
admin-c:    BB31-RIPE
admin-c:    FD14-RIPE
tech-c:     LD11-RIPE
tech-c:     MM107-RIPE
upd-to:     registr@pvt.cz
mnt-nfy:    registr@pvt.cz
auth:       MAIL-FROM .*@pvt.cz
notify:     registr@pvt.cz
mnt-by:     ASPVT-MNT
changed:    belousov@pvt.cz 951128
source:     RIPE
```

RIPE v odpovědi v položce aut-num vrátí číslo přiděleného AS:

```
To: „“ Ing. Boris Belousov „“ <belousov@pvt.cz>
From: RIPE NCC Hostmaster <hostmaster@ripe.net>
Subject: NCC#951712 Internet Autonomous System Number 5490 Assigned

Dear Ing. Boris Belousov,

This is to inform you that the Autonomous System Number 5490
has been issued by the RIPE NCC today. The RIPE database shows the
following information:

aut-num:      AS5490
descr:        PVTNET
descr:        PVT public IP network
descr:        Czech Republic
as-in:        from AS1849 100 accept ANY
as-out:       to AS1849 announce ANY
default:      AS1849 100
admin-c:      BB31-RIPE
admin-c:      FD14-RIPE
tech-c:       LD11-RIPE
tech-c:       BB31-RIPE
notify:       registr@pvt.cz
mnt-by:       ASPVT-MNT
changed:      hostmaster@ripe.net 951207
source:       RIPE

mntner:       ASPVT-MNT
descr:        PVT a.s.
descr:        Internet Service Provider
admin-c:      BB31-RIPE
admin-c:      FD14-RIPE
tech-c:       LD11-RIPE
tech-c:       MM107-RIPE
```

```
upd-to:      registr@pvt.cz
mnt-nfy:     registr@pvt.cz
auth:        NONE
notify:      registr@pvt.cz
mnt-by:      ASPVT-MNT
changed:     hostmaster@ripe.net 951207
source:      RIPE
```

Práci s databází RIPE upravuje dokument RIPE-189.

Přidělení (alokace) adresního prostoru

Před napsáním žádosti o přidělení adresního prostoru je třeba zvážit, o kolik adres třídy C žádat, protože je výhodné si nechat přidělit adresy tak, aby mohly být agregovatelné do jedné supersítě (CIDR). Ve směrovacích tabulkách ostatních poskytovatelů pak celý prostor vystupuje jako jedna položka – jedna supersíť.

Otázkou je, zdali se mají agregovat adresy zákazníků s adresami poskytovatele. V drtivé většině tato strategie vyhovuje. Tyto adresy se označují jako PA. Naopak ve výjimečných případech je možné žádat o adresy PI (Provider Independent). V takovém případě se agregují pouze sítě konkrétního zákazníka do jedné supersítě. Tato strategie je zdůvodnitelná zejména v případě, že se jedná o velmi velkého zákazníka, který chce být připojen k několika poskytovatelům současně. Zejména je-li zákazník připojen k poskytovatelům, kteří patří do různých regionálních IR, tj. je-li připojen k Internetu současně např. v Evropě a v Austrálii.

Rozlišujeme tři druhy adresních prostorů:

- Privátní adresní prostor, kde se přidělují IP-adresy podle RFC-1918, o takové adresy netřeba žádat.
- PA
- PI

O přidělení IP-adres se žádá na formuláři podle RIPE-141 (komenář k formuláři obsahuje RIPE-142). Formulář může odeslat pouze lokální IR, a to v ASCII-tvaru na adresu hostmaster@ripe.net.

```
#[OVERVIEW OF ORGANIZATION TEMPLATE]#
```

Informace o organizaci, kde bude adresní prostor použit.

Zde se anglicky popíše struktura organizace, tj. z jakých podřízených a nadřízených složek se skládá. Dále se popíše informace, jak bude adresní prostor využit jednotlivými odděleními organizace.

```
#[REQUESTER TEMPLATE]#
```

Informace o IR, který o přidělení žádá:

```
reg-ID: Identifikace lokálního IR (např. cz.pvtnet)
name: Plné jméno kontaktní osoby (neuvádí se tituly, za iniciálami se nepíše tečka)
organisation: Plný název organizace
country: cz
phone: Telefon do zaměstnání osoby uvedené v name
fax: (optional)
e-mail:
```

```
#[USER TEMPLATE]#
Kontaktní osoba organizace, kde bude prostor využit:

name:
organisation:
country:
phone:
fax: (optional)
e-mail:

#[CURRENT ADDRESS SPACE USAGE TEMPLATE]#
Používaný adresní prostor:
                                      Addresses Used
    Prefix        Subnet Mask      Size  Current  1-yr  2-yr  Description
```

Kde:

- *Prefix*: Adresa používané subsítě
- *Subnet Mask*: Síťová maska
- *Size*: Max. síťových interface na subsíti

Použití:

- *Current*: počet nyní používaných adres
- *1-yr*: počet interface za rok
- *2-yr*: počet interface za 2 roky
- *Description*: Slovní popis subsítě

```
#[REQUEST OVERVIEW TEMPLATE]#

request-size: Počet požadovaných adres celkem
addresses-immediate: Počet požadovaných adres okamžitě
addresses-year-1: Počet požadovaných adres do 12 měsíců
addresses-year-2: Počet požadovaných adres do 24 měsíců
subnets-immediate: Počet subsítí použitých okamžitě
subnets-year-1: Počet subsítí použitých do 12 měsíců
subnets-year-2: Počet subsítí použitých do 24 měsíců
inet-connect: Jméno providera nebo datum a jméno providera od kdy bude síť připojena
country-net: cz
private-considered: yes/no Vzal žadatel v úvahu použití adres pro privátní sítě?
request-refused: Uvede se v případě, že byl požadavek v minulosti odmítnut a proč
PI-requested: Yes (Provider independent space) jinak No
address-space-returned: Jestliže uživatel vrací nějaké adresy, pak se uvede interval.

#[ADDRESSING PLAN TEMPLATE]#
```

Adresní plán. Vyplní se obdobně jako používané adresy, pouze u žádaných adres není známo jejich číslo, tak se uvedou nuly (viz dále příklad).

```
Relative                              Addresses Used
Prefix#   Subnet Mask      Size  Immediate   1yr   2yr   Description



#[NETWORK TEMPLATE]#
Informace pro databázi RIPE:

netname: Zde uveďte krátký řetězec popisující adresní prostor
descr: Název a sídlo organizace
descr:
descr:
descr:
country: cz
admin-c: Osoba pověřená administrativním kontaktem
tech-c: Osoba pověřená technickým kontaktem
changed:
source: RIPE

#[PERSON TEMPLATE]#
```

V případě, že některá z osob uvedených v předchozí template nemá přidělen NIC-handle, pak se uvedou její osobní data:

```
person:
address:
address:
address:
address:
address:
address:
phone:
fax:

nic-hdl:
changed:

source: RIPE

#[TEMPLATES END]#
```

Na závěr je možné uvést ještě další údaje.

17.5.7.1 Příklad (podle RIPE-138)

```
#[OVERVIEW OF ORGANIZATION TEMPLATE]#

Santa's Workshop Inc. is divided into a Development Division
which includes departments for candy cane construction, toy man-
ufacturing and reindeer training ; and a Surveillance Division
that oversees investigations of naughty and nice children.  The
Development Division is located in in Christmastown, and each
department in the division has its own subnet.  The Surveillance
Division is an international operation with numerous branch
offices operating around the globe.
```

```
#[REQUESTER TEMPLATE]#

reg-id: nn.antarctic
name: Johnny B Good
organisation: Antarctic Internet Company
country: NN (or Northern Nowhere)
phone: +12 21 111 2222
fax: +12 21 222 1111
e-mail: jbgood@antarctic.nn

#[USER TEMPLATE]#

name: Santa A Claus
organisation: Santa's Workshop Inc.
country: NN
phone: +12 12 122 2121
fax: +12 12 221 1212
e-mail: santa@antarctic.nn


#[CURRENT ADDRESS SPACE USAGE TEMPLATE]#

                                      Addresses Used
Prefix          Subnet Mask      Size Current 1-yr  2-yr  Description

193.1.193.0     255.255.255.192   64    28    34    60    Candy Cane Dept.
193.1.193.64    255.255.255.224   32    10    21    28    Toy Design
193.1.193.96    255.255.255.224   32    8     13    27    Toy Manufacture
193.1.193.128                     128                     Unused
193.1.194.0     255.255.254       512   317   429   480   Reindeer Training
193.1.196.0     255.255.255       256   132   200   230   Naughty or Nice

                                  1024  495   697   825   Totals

#[REQUEST OVERVIEW TEMPLATE]#

request-size: 256
addresses-immediate: 120
addresses-year-1: 171
addresses-year-2: 202
subnets-immediate: 6
subnets-year-1: 7
subnets-year-2: 7
inet-connect: Already connected through Antarctic
country-net: NN
private-considered: yes
request-refused: yes - North Pole Inet refused us service
because we believe reindeer can fly.
PI-requested: no
addresses-returned: 192.0.2.0 - 192.0.2.256 to Antarctic Inet on
960623
```

```
#[ADDRESSING PLAN]#

Relative                                Addresses Used
Prefix#    Subnet Mask      Size  Immediate  1yr    2yr   Description

0.0.0.0     255.255.255.128   128     80     95     100   Toy Design
0.0.0.128   255.255.255.224   32      12     17     25    Toy Manufacture
0.0.0.160   255.255.255.224   32      0      15     28    Elf Personnel
0.0.0.192   255.255.255.240   16      7      8      10    Naughty or Nice North
0.0.0.208   255.255.255.240   16      10     14     14    Naughty or Nice South
0.0.0.224   255.255.255.240   16      10     12     12    Naughty or Nice East
0.0.0.240   255.255.255.240   16      8      10     13    Naughty or Nice West

                              256     120    171    202   Totals


#[NETWORK TEMPLATE]#

netname: XMAS
descr: Santa's Workshop Inc.
descr: Christmas Toys Manufacturing Facility
descr: Northern Nowhere
country: NN
admin-c: SC12-RIPE
tech-c: RR212-RIPE
changed: jbgood@antarctic.nn 960412
source: RIPE

#[PERSON TEMPLATE]#

person: Santa A Claus
address: Santa's Workshop Inc.
address: Jingle Bell Lane 12
address: 1224CH Christmastown
address: Northern Nowhere
phone: +12 12 122 1122
phone: +12 12 122 2211 ext. 1221
fax: +12 12 122 121
nic-handle: SC12-RIPE
changed: jbgood@antarctic.nn 960412
source: RIPE

#[PERSON TEMPLATE]#

person: Rudolf R N Reindeer
address: Santa's Workshop Inc.
address: Jingle Bell Lane 21
address: 1224CH Christmastown
address: Northern Nowhere
phone: +12 12 122 1111
fax: +12 12 122 2222
nic-handle: RD212-RIPE
```

```
changed: jbgood@antarctic.nn 960412
source: RIPE

#[TEMPLATES END]#
```

Nyní má poskytovatel alokován adresní prostor a přiděleny IP-adresy pro své sítě. Poskytovatel může začít s přidělováním IP-adres svým zákazníkům.

Poskytovatel musí se svými zákazníky vyplňovat stejné formuláře a odesílat je do RIPE ke schválení. Po schválení žádosti RIPE může teprve poskytovatel oficiálně přidělit IP-adresy zákazníkovi a zanést příslušný objekt do databáze RIPE.

Tento postup je velice zdlouhavý, proto po čase, kdy je RIPE s kvalitou práce poskytovatele spokojeno, může poskytovateli přidělit tzv. okno. Okno je počet adres, které mohou být celkově přiděleny zákazníkovi bez předběžného souhlasu RIPE. Vyjadřuje se většinou ve tvaru prefixu, tj. pokud poskytovatel může přidělit bez předchozího posvěcení až 256 adres, pak je přiděleno okno /24.

Je třeba zdůraznit, že se jedná o celkový počet přidělených adres. Má-li zákazník již přiděleno 128 adres a požaduje po poskytovateli přidělit dalších 192 adres, pak pokud má poskytovatel přiděleno okno /24, tak musí o těchto dalších 192 adres žádat RIPE, protože 128+192>256.

Rozeznáváme tak dvojí adresní prostor:

- Alokované adresy, které jsou poskytovateli rezervovány pro jeho další přidělování zákazníkům.
- Přidělené adresy, které jsou již z alokovaného prostoru přiděleny konkrétním zákazníkům poskytovatele.

Přidělení IP-adresy zákazníkovi

Zákazníkovi zpravidla přiděluje poskytovatel IP-adresy z intervalu, který má od RIPE alokován. Zvlášť provider žádá o přidělení jen v případě, že zákazník vyžaduje IP adresy nebo množství adres, které přesahuje aktuální Assignements Windows poskytovatele.

Po přidělení adres je nutné o zákazníkovi udržovat informace v databázi RIPE. Objekt prostor IP-adres přidělených zákazníkovi se spravuje ve dvou fázích. Nejprve se takový objekt vytvoří při přidělování adres. Pak se „odladí“ jmenný server pro reverzní domény a informace o tomto jmenném serveru se doplní do databáze RIPE.

```
inetnum:Interval IP adres
netname:Název adresního prostoru přidělovaného zákazníkovi.
descr:          Název a popis organizace, které je prostor přidělován.
admin-c:

tech-c:
rev-srv:Doplní se až při registraci reverzní domény.
remarks:

changed:
source: RIPE
```

Je-li jako admin-c nebo tech-c uvedena osoba, která ještě nemá registrovaný RIPE-handle, použije se se pro tuto osobu dočasný identifikátor ve tvaru AUTO-nXX (a – je pořadové číslo, XX jsou iniciály osoby). Pro každou takovouto osobu pak následuje popis objektu této osoby.

Při registraci objektu do RIPE databáze je automaticky dočasný RIPE-handle nahrazen jednoznačným vygenerovaným identifikátorem.

Příklad:

```
inetnum:194.149.100.0 - 194.149.100.15
netname:MOJE
descr:          Soukromy podnikatel Krvavy Pepa
admin-c:AB23-RIPE
tech-c:         CD98-RIPE
changed:hostmaster@pepa.cz
source:         RIPE
```

17.5.8 Notifikace a autorizace objektů

Asi vás napadlo, že měnit objekty v databázi by mohl každý, takže by někdo mohl snadno jiného uživatele poškozovat. Norma RIPE-120 popisuje zabezpečení proti takovýmto situacím, takže objekty mohou měnit jen oprávněné osoby.

Notifikace je velice jednoduchá metoda. Libovolnému objektu můžeme přidat položku:

notify: e-mailová adresa, na kterou je posílána zpráva o aktualizaci objektu

Když kdokoliv aktualizuje příslušný objekt, je na uvedou adresu posláno upozornění.

Složitější metodou je autorizace. Nejprve je nutné definovat objekt mntner (zavést jej může pouze personál RIPE). Libovolnému objektu je pak možné přidat položku mnt-by, která odkazuje na objekt mntner. Objekt pak může aktualizovat pouze osoba splňující podmínky v objektu mntner.

```
Položky objektu mntner:

mntner:         Název objektu (velká písmena a pomlčka).
descr:
admin-c:
tech-c:
upd-to:         Mailová adresa, na kterou se posílá zpráva v případě, že
                se pokusil objekt aktualizovat neautorizovaný uživatel.
mnt-ntf:Obdoba notify.
auth:           Typ autorizace, jsou zde tři možnosti:
                NONE    (žádná autorizace)
                CRYPT-PW zašifrované-heslo
                        MAIL-FROM .*@pvtnet.cz
remarks:
changed:
source: RIPE
```

Máme tedy tři typy autorizace:

1. NONE – bez autorizace
2. CRYPT-PW – pomocí hesla, které je v zašifrované formě uloženo v objektu mntner.
3. MAIL-FROM – pomocí jména počítače, ze kterého je požadavek odeslán.

Autorizace CRYPT-PW umožnuje aktualizaci jen v případě, že na začátek aktualizačního mailu napíšete položku:

password: heslo

Zatímco do položky passwd se heslo zadává nešifrované, tak do položky auth musíte zašifrovat heslo příkazem crypt (zašifrované heslo je např. v druhém poli souboru /etc/passwd v Unixu).

Autorizace MAIL-FROM umožňuje aktualizovat objekty pomocí mailů, které mají v hlavičce v položce FROM uvedenou specifikovanou doménu jako druhý parametr (v našem případě doménu pvtnet.cz).

Příklad

```
mntner:     ASPVT-MNT
descr:      PVT a.s.
descr:      Internet Service Provider
admin-c:    BB31-RIPE
admin-c:    FD14-RIPE
tech-c:     LD11-RIPE
tech-c:     MM107-RIPE
upd-to:     registr@pvt.cz
mnt-nfy:    registr@pvt.cz
auth:       MAIL-FROM .*@pvt.cz
notify:     registr@pvt.cz
mnt-by:     ASPVT-MNT
changed:    belousov@pvt.cz 951128
source:     RIPE
```


17.6 Registrace subdomén domén .com, .net a .org

I české firmy si někdy přejí registrovat subdomény těchto domén. Dříve se touto registraci zabýval InterNIC.

Bylo založeno sdružení ICANN (*The Internet Corporation for Assigned Names and Numbers*), které akreditovalo řadu organizací, jež mohou tuto registraci provádět (blíže viz http://www.icann.org). Váš poskytovatel Internetu je buď jednou z těchto organizací, nebo je s některou z těchto organizací v kontaktu a příslušnou registraci provede.

Jednotlivé akreditované organizace mají různé postupy pro registraci (elektronickou poštou, web-formulářem či dokonce telefonem). Vždy jde o zjištění, zdali doména není již registrována, uvedení údajů nutných pro registraci domény a zaplacení za registraci. Subdoména je po zaplacení registrována na dva roky. Poté se zpravidla po roce registrace obnovuje.

18

DNS v uzavřených podnikových sítích

Uzavřenou podnikovou sítí rozumíme síť, která nemá spojení do Internetu. V kap. 19 se pak budeme věnovat sítím, které mají částečné spojení do Internetu omezené firewallem.

Na první pohled se může zdát, že konfigurace DNS pro uzavřené podnikové sítě musí být úplně bezproblémová. V čem tedy tkví problém?

Ve vaší firmě je používána doména *firma.cz*. Nakonfigurujete vcelku bez problémů jmenný server pro doménu *firma.cz*. Jste dokonce pečliví a nakonfigurujete primární jmenný server a na jiném počítači sekundární jmenný server pro doménu *firma.cz*.

Všem klientům nasměrujete resolver na tyto jmenné servery. Na obr. 18.1 je znázorněno, jak klient žádá vámi nakonfigurovaný jmenný server o překlad jména server.firma.cz na IP-adresu.

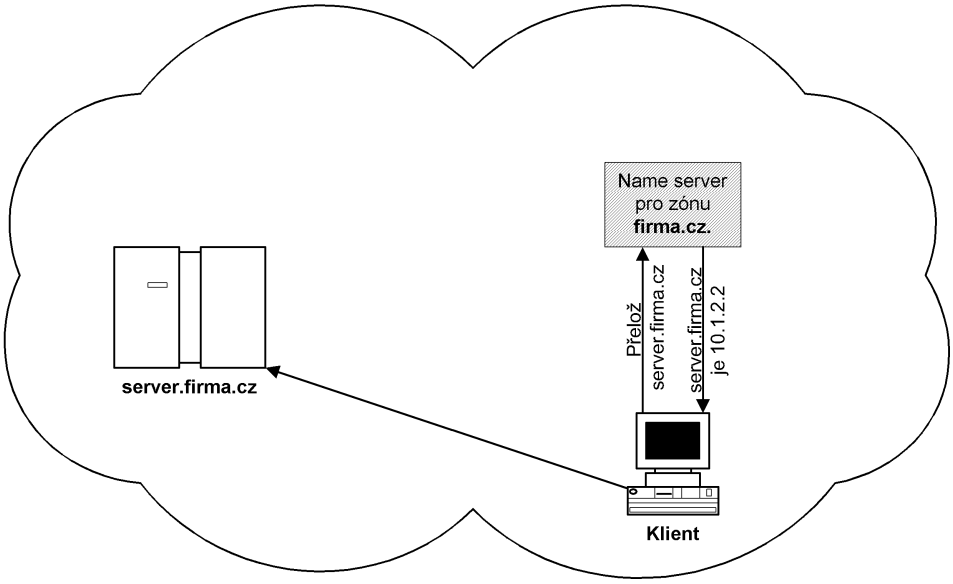
Vše pracuje korektně až do okamžiku, kdy se klient splete a místo *server.firma.cz* napíše *server.irma.cz* (splete se v jednom znaku).

Taková vcelku banální chyba ve jméně počítače způsobí, že nám aplikace zmrzne na několik desítek vteřin. To mnohé uživatele přivádí přímo k šílenství. Co se stalo? Na obr. 18.2 je zobrazena podstata problému. Nakonfigurovaný podnikový jmenný server je autoritou pro doménu *firma.cz*. Není však autoritou pro doménu *irma.cz*. Jelikož není autoritou pro doménu *irma.cz*, pak o překlad musí požádat kořenové jmenné servery, které mu pomohou nalézt autoritativní jmenný server, který jediný může sdělit, že počítač server v doméně *irma.cz* neexistuje (nebo existuje a je to úplně jiný počítač).

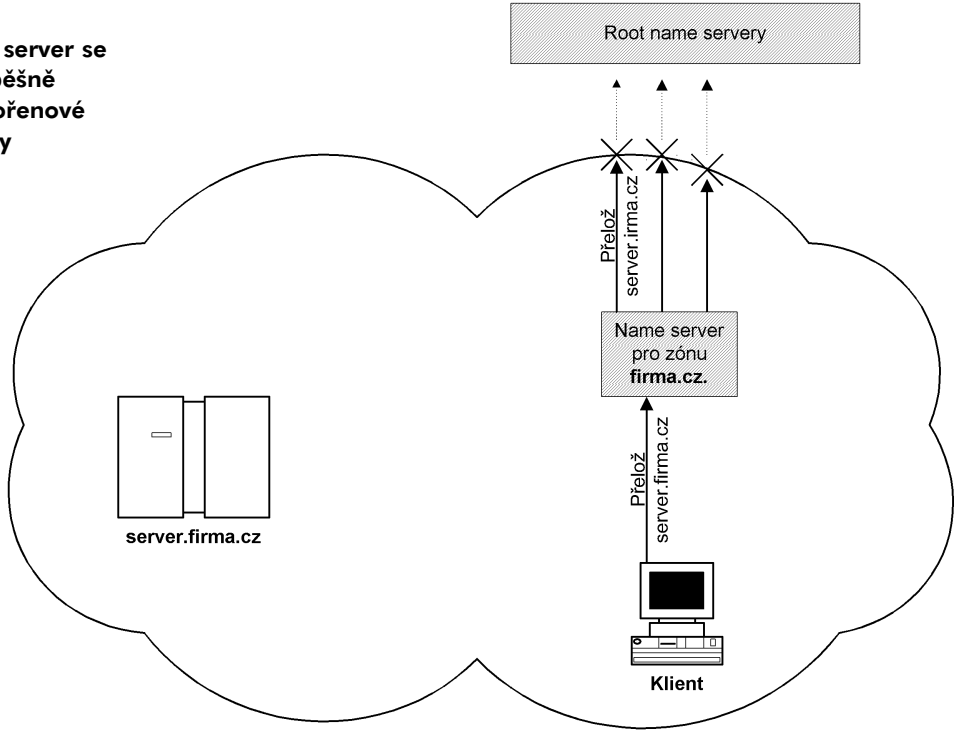
Jenže jsme v uzavřené podnikové síti, která nemá spojení do Internetu. Takže datagramy nesoucí dotazy na kořenové jmenné servery jsou nejpozději na hranici sítě zahozeny. Podnikový jmenný server nemůže dostat odpověď a nechá klienta na holičkách.

Resolver po několika desítkách vteřin když nedostane odpověď, tak si domyslí, že někde musí být nějaká chyba a uživateli vrátí chybovou hlášku. Ovšem chybová hláška se uživateli zobrazí jen v případě, že uživatel byl dostatečně trpělivý a mezi tím např. nezavedl znovu operační systém nebo …

Obr. 18.1
Překlad jména server.firma.cz na IP-adresu



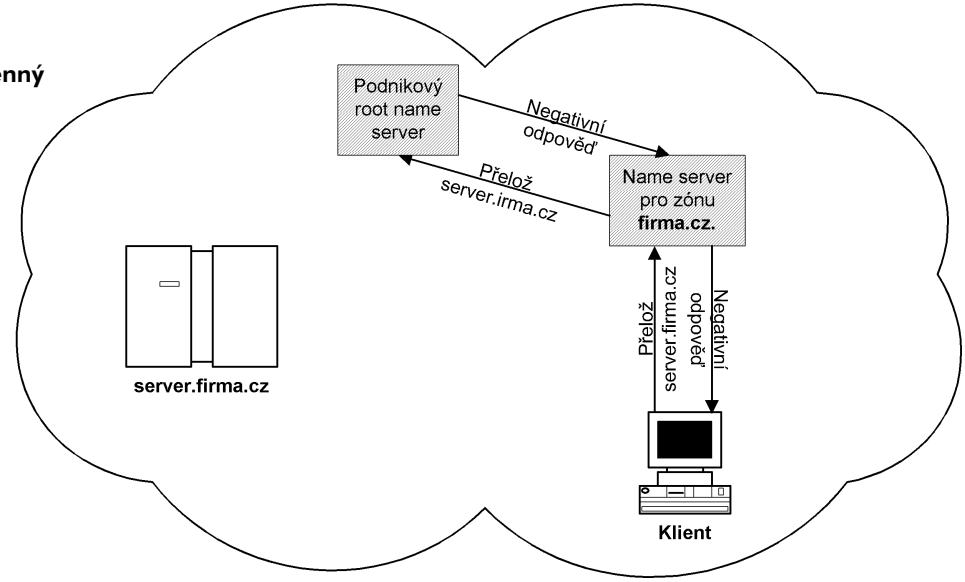
Obr. 18.2
Místní jmenný server se pokouší neúspěšně kontaktovat kořenové jmenné servery v Internetu



První reakcí správce je, že pochopí, že kořenové jmenné servery z uzavřené podnikové sítě kontaktovat nelze. Vzpomene si, že při startu jmenného serveru se do paměti jmenného serveru nahrávají data o kořenových jmenných serverech (soubor se zpravidla jmenuje *cache* a natahuje se pomocí řádku *cache* v */etc/named.boot*). Správce se domnívá, že příčinou jeho problémů je tento soubor, a proto jej zruší. Je však nemile překvapen, že se vůbec nic nezměnilo. Vysvětlení je prosté, pokud jmenný server nenajde vůbec žádné informace o kořenových jmenných serverech, tak přímo v kódu programu jmenného serveru jsou pro takový případ uvedeny autorem programu IP-adresy některých jmenných serverů.

Na obr. 18.3 je nakresleno, jak je třeba správně postupovat. Vytvoří se podnikový kořenový jmenný server (může jich být i více). A soubor s informacemi o kořenových jmenných serverech se nesmaže, ale opraví tak, aby vše bylo nasměrováno na náš podnikový kořenový jmenný server.

Pro kořenový jmenný server ani nemusí být nějaký speciální počítač. Můžeme jej nakonfigurovat i na stávajícím jmenném serveru tím, že zde vytvoříme primární jmenný server pro kořenovou doménu.



Obr. 18.3 Podnikový kořenový jmenný server vrací negativní odpověď: "V našem podnikovém Internetu neexistuje žádná doména irma.cz"

18.1 Konfigurace kořenového jmenného serveru na témž serveru (BIND 4)

Budeme předpokládat, že v naší podnikové síti máme dva jmenné servery:

- *ns1.firma.cz.* s IP-adresou 10.1.1.1
- *ns2.firma.cz* s IP-adresou 10.2.2.2

Oba jmenné servery budeme konfigurovat jako kořenové jmenné servery i jako jmenné servery pro zónu *firma.cz.*

V souboru `/etc/named.boot` serverů *ns1.firma.cz* a *ns2.firma.cz* musíme doplnit řádek deklarující, že náš jmenný server je také primárním jmenným serverem pro kořenovou doménu (pro tečku).

```
…
primary firma.czsoubor1
primary .               soubor2
…
```

Tj. není zde řádek s příkazem *cache*.

Důležité je, jak bude vypadat soubor *soubor2* specifikující kořenovou doménu:

```
@                       IN      SOA      …
.                       IN      NS      ns1.firma.cz.
ns1.firma.cz.           IN      A       10.1.1.1
.                       IN      NS      ns2.firma.cz.
ns2.firma.cz.           IN      A       10.1.1.1
```

V tomto souboru jsme neuvedli pro jinou doménu než *firma.cz* záznam typu NS, proto v našem podnikovém Internetu nejsou jiné domény. Pokud bychom chtěli mít v síti ještě nějaké jiné domény, pak je zde můžeme bez problému také specifikovat. Každopádně žádná doména *irma.cz* tam není. Zároveň delegujeme pravomoc pro doménu *firma.cz* na jmenné servery *ns1.firma.cz* a *ns2.firma.cz* (je jen shoda okolností, že jsou to ty stejné počítače).

Zónový soubor pro doménu *firma.cz* (*soubor1*) je pak již zcela shodný s tím, co očekáváme:

```
@                       IN      SOA      …
                        IN      NS      ns2.firma.cz.
                        IN      NS      ns1.firma.cz.
ns1                     IN      A       10.1.1.1
ns2                     IN      A       10.2.2.2
…
```

18.2 Konfigurace kořenového jmenného serveru na samostatném serveru (BIND 4)

Budeme předpokládat, že v naší podnikové síti máme dva jmenné servery pro doménu *firma.cz*:

- *ns1.firma.cz* s IP-adresou 10.1.1.1
- *ns2.firma.cz* s IP-adresou 10.2.2.2

A dále třetí jmenný server pro kořenovou doménu (pro tečku):

- *ns-root.firma.cz.* s IP-adresou 10.3.3.3

18.2.1 Konfigurace jmenného serveru pro kořenovou doménu

V souboru `/etc/named.boot` serveru *ns-root.firma.cz* musíme doplnit řádek deklarující, že náš jmenný server je primárním jmenným serverem pro kořenovou doménu (pro tečku).

```
…
primary .               soubor2
…
```

Tj. není zde řádek s příkazem *cache*.

Soubor *soubor2* specifikující kořenovou doménu bude delegovat pravomoc pro doménu *firma.cz* na jmenné servery *ns1.firma.cz* a *ns2.firma.cz*:

```
@                       IN      SOA      …
.                       IN      NS      ns1.firma.cz.
ns1.firma.cz.           IN      A       10.1.1.1
.                       IN      NS      ns2.firma.cz.
ns2.firma.cz.           IN      A       10.1.1.1
```

V tomto souboru jsme neuvedli pro jinou doménu než *firma.cz* větu typu NS, proto v našem podnikovém Internetu nejsou jiné domény. Pokud bychom chtěli mít v síti ještě nějaké jiné domény, pak je zde můžeme bez problému také specifikovat. Každopádně žádná doména *irma.cz* tam není. Zároveň delegujeme pravomoc pro doménu *firma.cz* na jmenné servery *ns1.firma.cz* a *ns2.firma.cz*.

18.2.2 Konfigurace jmenných serverů pro doménu *firma.cz*

V souboru `/etc/named.boot` serverů *ns1.firma.cz* a *ns2.firma.cz* musíme doplnit řádek s příkazem cache specifikující z jakého souboru se mají nahrát informace o kořenových jmenných serverech:

```
…
primary firma.cz  soubor1
cache .           soubor3
…
```

Soubor *soubor3* bude obsahovat neutoritativní informace o kořenových jmenných serverech (je uveden pouze jeden, ale můžeme jich mít i více):

```
.                       99999   IN      NS      ns-root.firma.cz.
ns-root.firma.cz.       99999   IN      A       10.3.3.3
```

Tento soubor nemůže obsahovat záznam SOA, protože ta uvozuje pouze autoritativní údaje (SOA=*Start of Autority*). Další zajímavostí je druhý sloupec obsahující číslo 99999. S tímto sloupcem se v jiných souborech zpravidla nesetkáváme. Sloupec specifikuje dobu života věty v paměti (TTL). Proč zde musí být uveden? Odpověď je prostá. V ostatních databázích nebývá tato hodnota uvedena, protože když není uvedena, tak se tato hodnota vezme ze záznamu SOA. Avšak zde nemůže být záznam SOA, proto se tato hodnota musí explicitně uvést. Pokud bychom ji neuvedli, pak při nahrání by data okamžitě vypršela (TTL=0), tj. data by byla prohlášena za neplatná. Jmenný server by neměl informace o žádných kořenových jmenných serverech, a tak by přišly ke slovu IP-adresy uvedené přímo v programu – efekt by byl stejný jako na obr. 18.2.

Zónový soubor pro doménu *firma.cz* (*soubor1*) je pak již zcela shodný s tím, co očekáváme:

```
@               IN      SOA      …
                IN      NS      ns2.firma.cz.
                IN      NS      ns1.firma.cz.
ns1             IN      A       10.1.1.1
ns2             IN      A       10.2.2.2
ns3             IN      A       10.3.3.3
…
```

19

DNS a firewall

Firewall odděluje vnitřní podnikovou síť (intranet) od Internetu. Umožňuje klientům vnitřní sítě čerpat informace z Internetu a znemožňuje útočníkům z Internetu útočit proti počítačům ve vnitřní síti.

Firma má přidělenu doménu *firma.cz*. Avšak tuto doménu chce používat v Internetu i v intranetu. V Internetu bude v doméně *firma.cz* nejspíš pouze: *www.firma.cz*, *mail.firma.cz* a několik málo dalších záznamů (MX záznamy pro *firma.cz* ukazující na *mail.firma.cz* atd.). Kdežto v intranetu mohou být v doméně *firma.cz* desítky, stovky či tisíce počítačů.

Jinými slovy: budou dvě domény *firma.cz*, každá bude obsahovat jiné záznamy, ale potíž je v tom, že se jmenují stejně – *firma.cz*. V Internetu nemohou existovat dvě různé domény stejného jména. Avšak tato situace nenastává v samotném Internetu, ale jedna doména je v Internetu a druhá v intranetu.

Problém je s firewallem jako takovým. Na firewallu běží aplikace (např. proxy), které potřebují pracovat s doménou *firma.cz* vnitřní sítě, ale s ostatními doménami z Internetu. Navíc firewall jako jmenný server, který se z hlediska klientů v Internetu musí tvářit, že pracuje s doménou *firma.cz* z Internetu.

Aplikace běžící na firewallu (např. proxy) využívají resolver, kdežto firewall jako jmenný server bude poskytovat informace jako server. Jako nástroj se využívá skutečnost, že resolver nemusí být nasměrován na jmenný server běžící na místním počítači (na 127.0.0.1).

Jedním problémem je vlastní zapojení firewallu a přidělení IP-adres (viz obr. 6.9). Jiným problémem je konfigurace firewallu z hlediska DNS. Oba problémy jsou na sobě v podstatě nezávislé.

Z hlediska konfigurace DNS na firewallu může nastat celá řada eventualit. Probereme několik modelových situací. V praxi však nejspíš použijete kombinaci těchto situací.

19.1 Společné DNS pro Internet i intranet

Nejjednodušším řešením je udělat společné DNS pro Internet i intranet. Toto řešení je považováno za nedokonalé zejména ze dvou příčin:

1. Na Internetu se zveřejňují překlady počítačů, které mají nesměrovatelné adresy (síť 10/8, 172.16/12 či 192.168/16).
2. Zveřejňují se informace o struktuře firmy (IP-adresy počítačů vnitřní sítě). Tyto informace jsou zpravidla utajovány.

Kardinální otázkou při konfiguraci DNS na firewallu vždy je, zdali se mají překládat všechny jména Internetu ve vnitřní síti. Nebo zdali klienti vnitřní sítě mají mít možnost překládat pouze jména z domény *firma.cz* ve vnitřní síti.

19.1.1 Ve vnitřní síti se překládá celý Internet

Pokud se překládá celý Internet ve vnitřní síti, pak se i ve vnitřní síti musí směrovat (dopravovat) IP-adresy celého Internetu. To má opět dva nepříjemné důsledky:

1. Směrování vnitřní sítě musí být na tuto skutečnost připraveno. Tj. veškeré IP-adresy, které nejsou z vnitřní sítě, musí směrování dopravit na firewall. Zpravidla se to provádí pomocí položky *default* ve směrovacích tabulkách. Zejména v případě rozsáhlých intranetů nemusí být triviálním problémem takové směrování stabilně udržet.
2. Bezpečnostní manažeři sledují vnitřní síť, zdali v ní nedochází k útokům z jiných sítí. Pokud se v síti dopravují pouze IP-datagramy adresované v sítích 10/8, 172.16/12 či 192.168/16, pak při výskytu jiné adresy se snadno může signalizovat možnost bezpečnostního incidentu. Pokud se však mohou ve vnitřní síti vyskytovat legálně zcela libovolné adresy Internetu, pak bezpečnostním manažerům bereme tento nástroj.

Překlad celého Internetu ve vnitřní síti se používá zejména ve dvou případech:

- Pokud jsou na firewallu provozovány transparentní proxy. Transparentní proxy je příjemná pro uživatele používajícího telnet nebo ftp. Pokud se chce uživatel přihlásit programem telnet např. na server *www.playboy.com*, pak se nemusí hlásit nejprve k proxy (firewallu) a teprve z něj dále na cílový server. Prostě ve vnitřní síti napíše jen *„telnet www.playboy.com*“. Transparentní proxy na firewallu akceptuje spojení jakoby byla sama cílovým serverem a předá požadavek dále jménem klienta na cílový server do Internetu.

  Jenže na *www.playboy.com* se většinou uživatelé nepřihlašují programem telnet, ale pomocí webového prohlížeče. A webový prohlížeč žádné transparentní proxy nepotřebuje.

  Závěr je takový, že běžní uživatelé telnet a ftp nepoužívají a správcům a vývojářům použití klasické proxy pro programy telnet a ftp příliš nevadí.

- Pokud se nepoužívá klasický firewall s proxy, ale pouze ochrana vnitřní sítě pomocí filtrace.

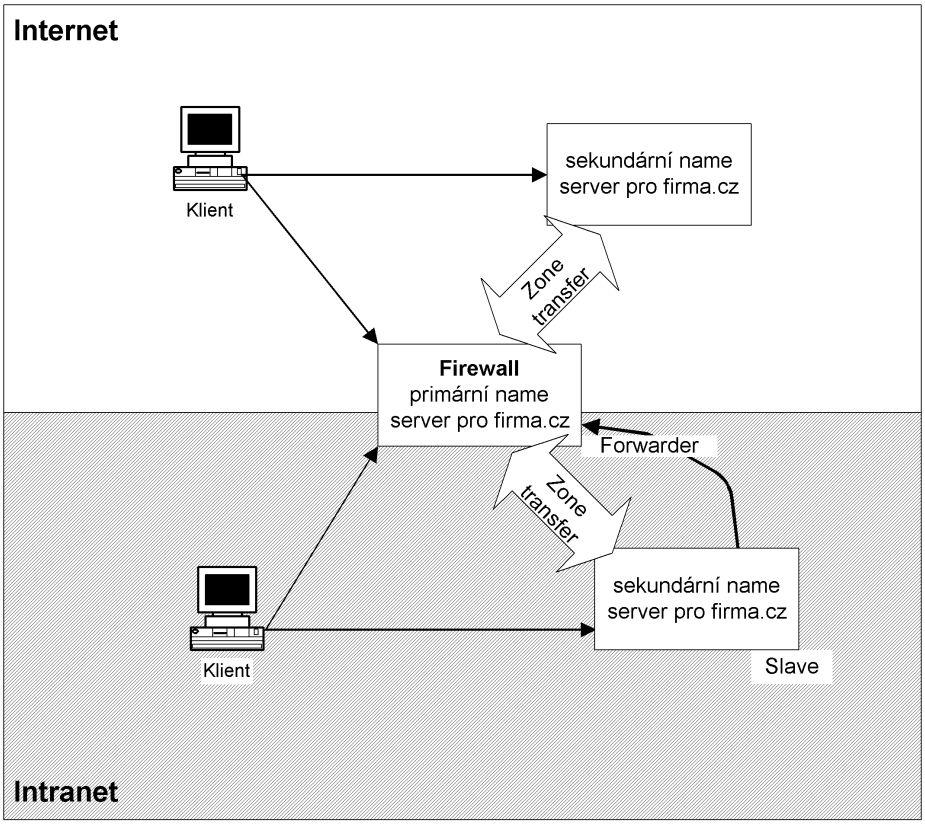
Firewall zpravidla pracuje jako primární jmenný server pro doménu *firma.cz* (viz obr. 19.1), která je společná jak pro intranet, tak i pro Internet.

Je rozumné mít alespoň dva jmenné servery pro doménu (primární a sekundární). Avšak je třeba mít dva jmenné servery dostupné v Internetu a dva v intranetu. Jelikož firewall je dostupný z obou sítí, tak stačí nakonfigurovat jeden sekundární jmenný server pro Internet a druhý pro intranet.

Sekundární jmenný server v Internetu pro naši doménu budeme nejpravděpodobněji provozovat u našeho poskytovatele Internetu.

Sekundární jmenný server v intranetu zřídíme na nějakém dalším počítači. Pokud bude klient intranetu požadovat překlad jména z jiné domény přímo na firewallu, pak to není potíž. Firewall může o pomoc požádat kořenové jmenné servery Internetu a klienta uspokojí. Pokud však klient požádá o překlad jména z jiné domény sekundární jmenný server v Intranetu, pak je problém v tom, že tento sekundární jmenný server nemá spojení do internetu a nemohl by proto o takový překlad požádat kořenové jmenné servery. Aby i takové překlady provádět mohl, tak se sekundární jmenný server

Obr. 19.1



intranetu nakonfiguruje jako podřízený server vůči firewallu jako forwarderovi. Firewall překlad provede a předá jej podřízenému serveru, který jej obratem předá klientovi.

19.1.2 Ve vnitřní síti se Internet nepřekládá

Překlad adres Internetu ve vnitřní síti většinou není vůbec nutný. Ve vnitřní síti je nutné umět přeložit pouze jméno firewallu (proxy), protože klient navazuje spojení s proxy a ta teprve navazuje jménem klienta spojení s cílovým serverem v Internetu. Tj. až proxy musí umět přeložit jméno cílového serveru v Internetu na IP-adresu.

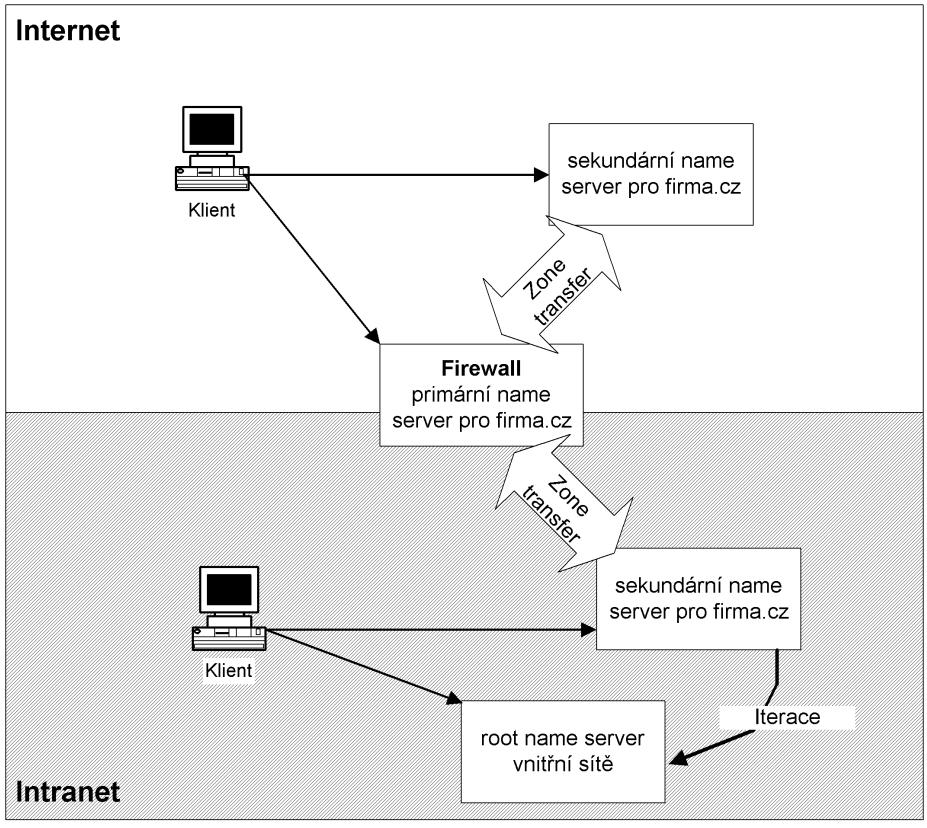
Pokud si to chcete ověřit, pak si procvičte následující dva příklady.

1. **Stažení webové stránky *www.playboy.com* programem telnet klientem v Internetu**. Využijte program telnet, ale vždy volte port („zásuvku" pokud máte klienta Microsoft) 80 a nikoliv 23. Nyní již můžete navázat spojení programem telnet na portu 80 a zadat příkaz (někdy bývá praktické mít v programu telnet nastavenu místní odezvu – lokální echo):

   ```
   GET / HTTP/1.0
   <Enter><Enter>
   ```

   a bude vám vrácena titulní stránka, tj. nejspíše soubor *index.html*. (`<Enter>` znamená zmáčknout klávesu `<Enter>`.)

Obr. 19.2



2. **Stažení webové stránky *www.playboy.com* programem telnet klientem ve vnitřní síti.** Pokud jste v intranetu za firewallem (a máte skrze firewall přístup bez další autentizace do Internetu), pak navažte spojení s proxy na portu, na kterém běží proxy (bývá to zpravidla port 8080) a vložte příkaz:

```
GET http://www.playboy.com HTTP/1.0
<Enter><Enter>
```

a bude vám vrácena titulní stránka, tj. nejspíše soubor *index.html*. Všimněte si, že proxy jste předávali jméno cílového serveru *www.playboy.com* jako textový řetězec – nikoliv IP-adresu.

Na obr. 19.2 je znázorněno, jak je nakonfigurováno DNS ve vnitřní síti, aby neprovádělo překlady Internetu. Ve vnitřní síti se zřídí kořenový jmenný server. Sekundární jmenný server pro firma.cz se nasměruje na tento kořenový jmenný server. Pokud je požadavek na jinou doménu než na doménu *firma.cz*, pak kořenový jmenný server odpoví negativně.

Jelikož je praktické mít ve vnitřní síti dva jmenné servery, tak se prakticky zřizují dva počítače. Na obou běží sekundární jmenné servery pro *firma.cz* a současně na nich běží kořenové jmenné servery.

Je třeba ještě upozornit, že pokud klient vnitřní sítě nasměruje svůj resolver přímo na firewall, pak mu firewall přeloží libovolnou adresu Internetu. Klientův resolver proto musí být nasměrován na jmenný servery ve vnitřní síti.

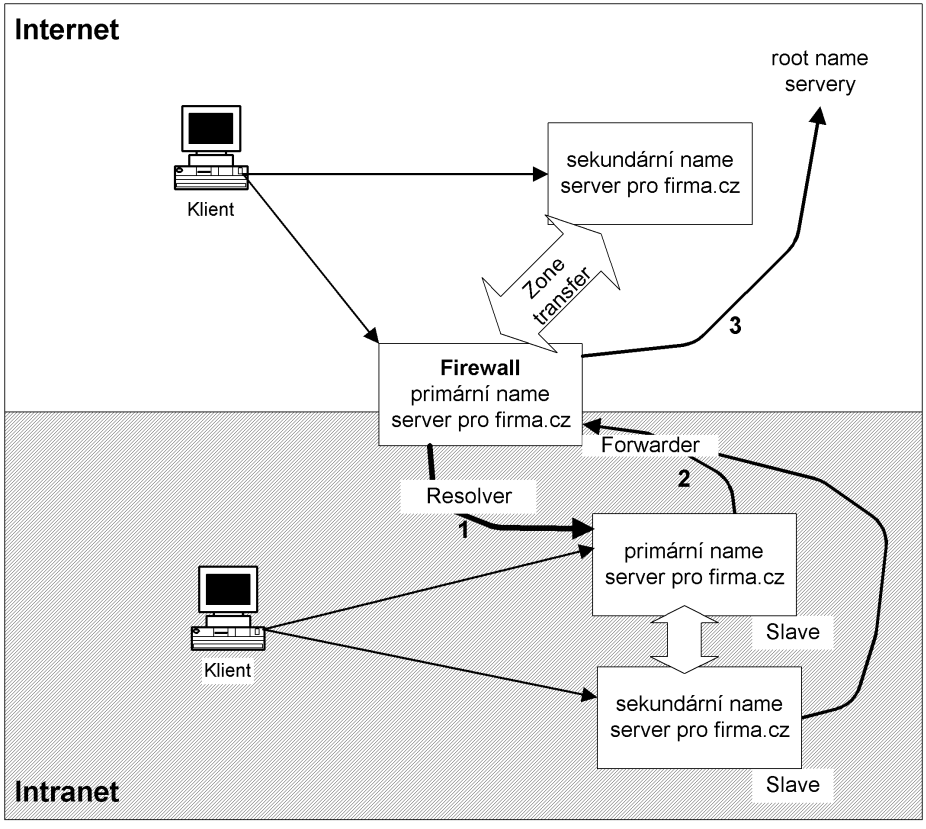
19.2 Na firewallu je jmenný server Internetu

Pokud chceme mít dvě oddělené zóny pro doménu *firma.cz*, pak zpravidla bývá primární jmenný server pro Internet na firewallu a sekundární jmenný server pro Internet na počítači poskytovatele Internetu. Pro vnitřní síť se zřídí samostatná dvojice primární/sekundární server na serverech ve vnitřní síti.

Opět jsou tu dvě možnosti. První možnost dovoluje překládat ve vnitřní síti celý Internet a druhá možnost překládat ve vnitřní síti pouze zónu vnitřní sítě.

19.2.1 Ve vnitřní síti se překládá celý Internet

Obr. 19.3



Máme dvě samostatné dvojice jmenných; serverů (viz obr. 19.3). Jedna dvojice je ve vnitřní síti a druhá dvojice v Internetu.

Prvotním problémem je, že aplikace běžící na firewallu (např. proxy) potřebuje informace o zóně *firma.cz* vnitřní sítě, avšak potřebuje také informace o všech ostatních doménách Internetu. Provede se to tak, že resolver firewallu není nasměrován na jmenný server firewallu, ale na jmenný server vnitřní sítě, který má k dispozici zónu vnitřní sítě.

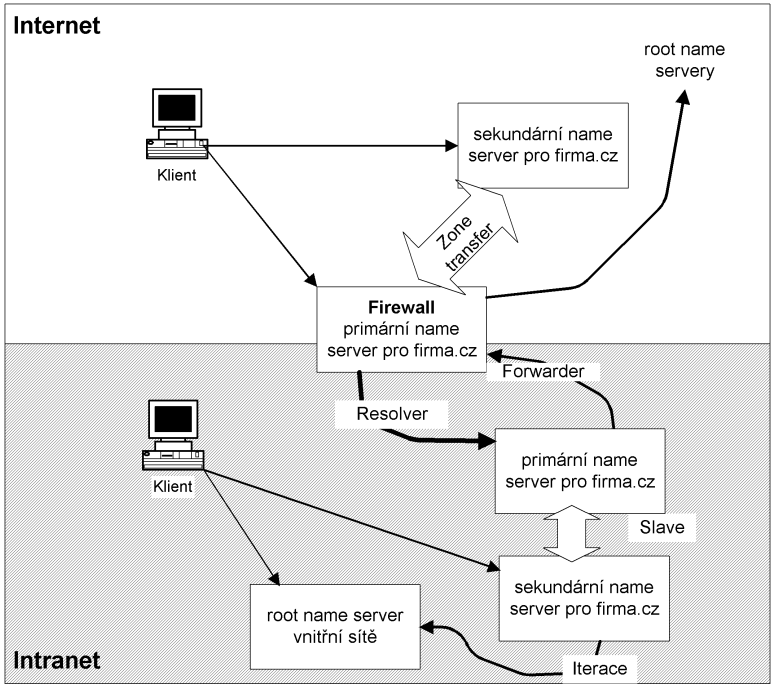
Avšak v případě, že aplikace na firewallu potřebuje přeložit např. *www.playboy.com*, pak o tento překlad také požádá jmenný server vnitřní sítě (šipka 1 na obrázku 19.3). Jmenný server vnitřní sítě je podřízeným serverem, který všechny požadavky, které neumí vyřešit sám, předá předávači – firewallu (šipka 2). Jmenný server firewallu pak již má přístup ke kořenovým jmenným serverům (šipka 3) a tak může provést překlad. Výsledek předá zpět podřízenému serveru, který obratem vše předá klientu na firewallu.

Klient vnitřní sítě požaduje překlady od jmenných serverů vnitřní sítě. Pokud se jedná o překlad místní domény, pak mu vrátí odpověď; pokud se však jedná o překlad domény z Internetu, pak takový požadavek předá předávači, tj. firewallu.

19.2.2 Ve vnitřní síti se Internet nepřekládá

Varianta, kdy se ve vnitřní síti Internet nepřekládá znamená vytvořit ve vnitřní síti kořenový jmenný server (viz obr. 19.4). Zajímavé je, že ve vnitřní síti jsou minimálně dva jmenné servery a každý má jinou funkci. První (na obr. 19.4 označený jako primární jmenný server) slouží firewallu. Pokud by si klient vnitřní sítě na něj nasměroval svůj resolver, pak by mu tento jmenný server přeložil cokoliv z Internetu a ještě zónu *firma.cz* ve vnitřní síti. To však nechceme, proto jsou resolvery klientů vnitřní sítě nasměrovány na další jmenné servery ve vnitřní síti, které používají kořenový jmenný server vnitřní sítě. Kořenové jmenné servery vnitřní sítě pak zamezí překladu jiných domén.

Obr. 19.4



19.3 Duální DNS

Pokud chceme mít samostatnou zónu pro vnitřní síť a samostatnou zónu pro Internet, pak díky tomu, že se stejně jmenují, tak každou zónu musíme mít na samostatném počítači. Cílem duálního DNS je provozovat primární jmenný server pro zónu *firma.cz* vnitřní sítě i Internetu na jednom počítači. Důvod je ekonomický. Zatímco u velkých firem se na vnitřní síti provozuje celá řada serverů, na kterých mohou běžet jmenné servery, tak u menších firem by bylo často nutné instalovat další počítač jen proto, aby na něm mohl běžet jmenný server.

Princip duálního DNS spočívá v tom, že na firewallu poběží dva jmenné servery (dva procesy). Každý však běží na jiném portu. Na obr. 19.5 jmenný server pro Internet běží na portu 7053 a jmenný server pro Intranet běží na portu 8053.

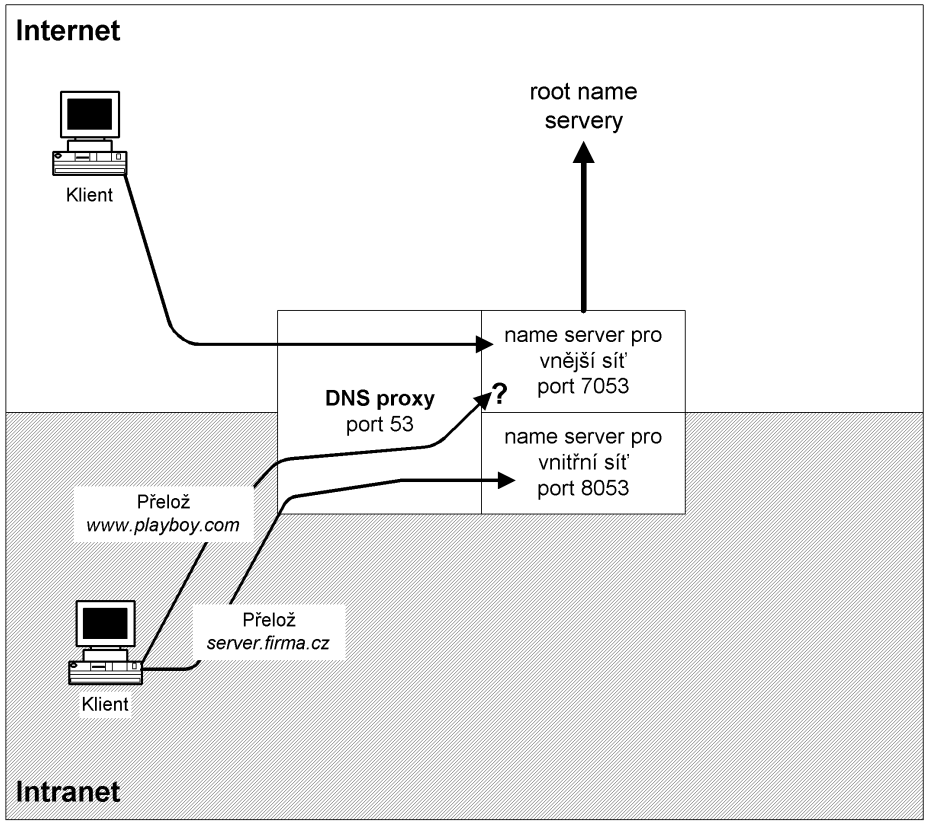
Běžní klienti mohou těžko využít tyto jmenné servery na jiném portu než na portu 53, protože o portech 7053 a 8053 se nedoví.

Na firewallu běží na standardním portu pro jmenný server tzv. DNS proxy. DNS proxy zjišťuje odkud požadavek přichází. Podle toho odkud požadavek přichází, tak jej buď zamítne, předá jmennému serveru na portu 7053 nebo jmennému serveru na portu 8053.

Požadavek může přicházet od:

- Klienta Internetu, pak jej předá jmennému serveru pro Internet (na obrázku port 7053).
- Klienta intranetu, pak se jedná o dva případy:
  1. Jedná se o požadavek na překlad z domény *firma.cz*, pak jej předá jmennému serveru pro vnitřní síť (na obrázku port 8053).
  2. Jedná se o požadavek na překlad jiné domény Internetu. Pak se DNS proxy rozhoduje:
     - Chceme-li ve vnitřní síti překládat Internet, pak požadavek předá jmennému serveru pro Internet (port 7053).
     - Nechceme-li ve vnitřní síti překládat jiné domény Internetu, pak odpoví negativně. Je zajímavé, že pokud ve vnitřní síti nemáme další (např. sekundární) jmenné servery, pak nepotřebujete na vnitřní síti kořenový jmenný server. Negativní odpověď vydá přímo DNS-proxy.
- Aplikace běžící na firewallu (např. proxy) pak se zkoumá, zdali se jedná o požadavek na doménu *firma.cz*, ten je předán jmennému serveru pro vnitřní síť (port 8053), nebo jestli se jedná o jiný požadavek, ten je předán jmennému serveru pro Internet (port 5073).

Obr. 19.5
Duální DNS



R

Rejstřík

N

O

P

R

Ř

S

T

U

V

W

Z

Ž