UNIVERZITA PALACKÉHO V OLOMOUCI
KATEDRA INFORMATIKY

Počítačové sítě

Martin Kopka

září 1996 TR-CS-96-04

Předmluva

Skriptum je psáno jako pomocný text pro studenty kursu *Počítačové sítě* prvního ročníku rozšiřujícího studia informatiky, využití však najde i u studentů denního studia na katedře matematické informatiky. Cílem textu je uvést do problematiky propojování počítačů, seznámit s teoretickými i praktickými aspekty počítačových sítí a ukázat problémy, které jsou aktuálně v oboru řešeny.

Úvodní kapitola obsahuje charakteristiky jednotlivých typů sítí a jejich možných topologií a základní pojmy z oblasti standardizace počítačových sítí. Vedle referenčního modelu OSI je popsáno v současnosti zřejmě nejpoužívanější schéma – architektura TCP/IP.
Následují kapitoly o některých pojmech používaných v dalším výkladu (teorie informace, kódování, komprese dat).

Čtvrtá kapitola se zabývá metodami přenosu dat, rozebírá základní typy přenosu a zabezpečení dat.

Kapitola o lokálních počítačových sítích podává přehled používaných přenosových médií a přístupových metod, navíc popisuje tři základní typy lokálních sítí, a to *Token–Ring*, *Arcnet* a *Ethernet*.

Speciální kapitoly jsou věnovány problematice směrování a propojování sítí včetně vysokorychlostních sítí. V kapitole o rozlehlých počítačových sítí je kromě základního popisu technických vlastností čtenář seznámen se standardy a doporučeními, na základě kterých dané služby pracují. Pozornost je věnována digitálnímu přenosu dat.

Poslední kapitola pojednává o rodině protokolů TCP/IP a některých vlastnostech sítě Internet. Příloha nabízí krátké seznámení s prací v operačním systému UNIX.

Tento text lze získat na anonymním FTP serveru `ftp.inf.upol.cz` v adresáři `/pub/techrep/` jako soubor `TR-CS-96-04` .

V seznamu literatury je uvedena kromě citované literatury také další dostupná literatura, která byla v poslední době vydána. Jedná se o několik vysokoškolských skript a odkazy na časopisy, zabývající se aktuálními tématy z počítačových sítí.

Text byl napsán za pomoci textového editoru Emacs a vysázen typografickým systémem LaTeX2ε v operačním systému UNIX.

V Olomouci, 22. září 1996 autor

Obsah

Seznam obrázků

1 Počítačové sítě (*Computer networks*)

Počátek vývoje počítačů spadá do 50. let tohoto století. Uveďme několik etap ve vývoji počítačových sítí, které měly zásadní význam pro jejich rozvoj.

V 60. letech dochází ke spojení počítače s komunikačními prostředky a tím ke vzniku **terminálové sítě**. Ta obsahuje jeden výkonný uzel - **počítač**, který vykonává veškerou výpočetní práci a k němu připojená koncová zařízení - **terminály**, které slouží pouze pro vstup a výstup dat. Z tohoto jednoznačného rozdělení kompetencí a přísné centralizace zpracování vyplývá i relativní jednoduchost terminálové sítě. Všechny obtížné problémy synchronizace procesů, řízení přístupu k datům a ostatním prostředkům i ochrana integrity dat se odehrávají a řeší v jediném uzlu známými technikami multiprogramových operačních systémů. Později bylo umožněno připojení terminálů i pomocí telefonního vedení, tj. na větší vzdálenosti.

Počátkem 70. let začínají vznikat požadavky na vzájemné propojení od sebe vzdálených počítačů. Nestačí již, že pomocí terminálové sítě má více uživatelů přístup k jedinému počítači, ale účastník u terminálu potřebuje mít přístup k více počítačům tak, aby mohl používat jejich databáze, případně i velký výpočetní výkon. **Počítačová síť** [3] je skupina počítačů popř. periferií, které jsou mezi sebou propojeny tak, aby zajistily vzájemnou komunikaci libovolného uživatele s programem na libovolném počítači, dvou programů mezi sebou nebo dvou libovolných uživatelů mezi sebou, a to při vysoké spolehlivosti komunikace.

V 80. létech, kdy se začínají objevovat výkonné sítě LAN, se počítačová síť mění z prostředku komunikace na komplexní výpočetní systém zajišťující různé služby.

Existuje několik základních důvodů pro použití počítačových sítí. Prvním důvodem je **sdílení zdrojů** (počítačů, programů a dat) kterýmkoliv uživatelem sítě bez ohledu na fyzické umístění zdroje a uživatele. Dalším důvodem je **zvýšení spolehlivosti**, ke kterému dochází v důsledku přístupu k alternativním zdrojům (hardware nebo software). Nezanedbatelným důvodem je také **úspora finančních prostředků**. Malé počítače mají mnohem příznivější poměr cena/výkon, než je tomu u střediskových počítačů (*mainframe*). Zavádění relativně samostatných personálních počítačů s operačními systémy Windows vyžaduje jejich propojení do sítě a sdílení společných dat nebo programů.

Dalším kvalitativním krokem ve využívání vypočetní techniky je zavádění **distribuovaných systémů** [1] (distribuovaný souborový nebo výpočetní systém). Zatímco v počítačové síti se musí uživatel explicitně přihlásit na určitý počítač, spouštět explicitně vzdálené úlohy s vědomím jejich přesného uložení, explicitně přesouvat soubory a ovládat veškeré zařízení sítě osobně, v distribuovaném systému není třeba nic dělat explicitně. Veškerý přístup ke službám poskytuje systém automaticky bez vědomí uživatele o tom, kdo službu poskytuje nebo kde se nacházejí data, se kterými služba pracuje.

Uživatel distribuovaného systému může napsat příkaz pro vykonání procesu a tento proces začne být zpracováván. Výběr nejvhodnějšího procesoru, nalezení a přepravení všech vstupních souborů k tomuto procesoru a vrácení výsledků na požadovaná místa je záležitostí distribuovaného systému. Jinými slovy distribuovaný systém (rozprostřený mezi několik počítačů v síti) se uživateli jeví jako jednoprocesorový počítač, ke kterému se jednou přihlásí a tím získá možnost vyžívat jeho služby.

Takto chápaný distribuovaný systém je zvláštním případem počítačové sítě. Jde o síť, jejíž programové vybavení jí dává vysoký stupeň soudržnosti a transparentnosti. Rozdíl mezi počítačovou sítí a distribuovaným systémem je tedy dán více programovým vybavením (především operačním systémem) než technickým vybavením. Počítačová síť se většinou chápe jako komunikační podsíť distribuovaného systému.

1.1 Rozlehlost počítačových sítí

V souvislosti s počítačovými sítěmi hovoříme o třech typech sítí podle jejich územní rozlehlosti.

[1] Distribuované systémy a jejich vlastnosti jsou náplní stejnojmenného kursu vyučovaného ve 4. ročníku studia.

1.1.1 Lokální počítačová síť (*Local Area Network - LAN*)

Prvky takovéto sítě jsou rozmístěny v určitém ohraničeném objektu, který se rozprostírá v okruhu stovek metrů. Celá síť je pod logickou i fyzickou kontrolou jednoho pracovníka, označovaného jako správce sítě (*supervisor*). Síť se skládá obvykle z osobních mikropočítačů a pracovních stanic doplněných o potřebné hardwarové prostředky (síťové adaptéry, konektory) spojené síťovými kabely. Přenosová média jsou různá (od levné kroucené dvojlinky přes koaxiální kabel až po vysokorychlostní optické kabely).

Lokální počítačová síť zajišťuje tyto služby :

- sdílení nákladných periferií (laserové tiskárny, velkokapacitní diskové systémy, mechaniky CD-ROM apod.),
- sdílení společných dat a aplikací (aktuální data, úspora diskového prostoru, snadná záloha, bezproblémový přechod na vyšší verzi produktu apod.),
- jednoduchá vzájemná komunikace uživatelů (lokální pošta),
- provozování lokálního informačního systému (INTRANET)

Příkladem takovéto sítě je LAN Hejčín, která se nachází na katedře informatiky v Olomouci - Hejčíně, lokální sítě na školách nebo v podnicích.

1.1.2 Městská počítačová síť (*Metropolitan Area Network - MAN*)

Jak už sám název napovídá, jednotlivé počítače jsou rozmístěny v rozsahu města, tedy několika kilometrů. Se zlepšováním komunikačních prostředků a tím zkvalitňováním poskytovaných služeb by mohly být městské sítě zařazeny mezi LAN, avšak na druhé straně se při realizaci městských sítí používají principy uplatňované ve WAN.

Příkladem této sítě je síť Univerzity Palackého v Olomouci UPONET.

1.1.3 Rozlehlá počítačová síť (*Wide Area Network - WAN*)

Počítače rozlehlé sítě jsou umístěny ve více městech, dokonce i ve více státech či kontinentech. Sítě WAN jsou tvořeny řídícími počítači (tzv. uzlovými počítači), které jsou propojeny mezi sebou prostřednictvím komunikační podsítě. Komunikační podsíť tvoří většinou speciální datové spoje organizací poskytujících telekomunikační služby. Jedná se nejčastěji o pevné telefonní linky nebo optické kabely, existují však i možnosti mikrovlného a družicového spojení. Uzly WAN jsou obvykle výkonné střediskové počítače (*mainframes*) pracující pod operačními systémy UNIX, VMS, Windows NT. Jde tedy o techniku schopnou obsloužit větší počet uživatelů současně a pracující nepřetržitě. V poslední době se za uzly WAN považují i jednotlivé LAN, které mezi sebou komunikují právě prostřednictvím rozlehlé sítě. U rozlehlých sítí není prakticky možné propojit každý počítač s každým a vzájemné propojení tedy probíhá zprostředkovaně. Zpráva je předávána od jednoho počítače ke druhému, a to až k cílovému místu.

Rozlehlá počítačová síť poskytuje tyto služby :

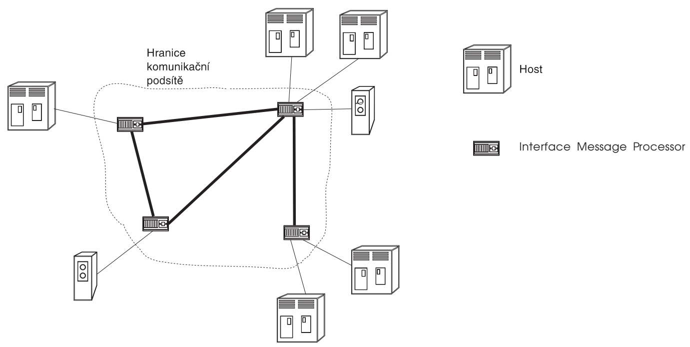
- práce na vzdálených počítačích (*remote login*),
- přenos dat,
- elektronická pošta (*e-mail*),
- přístup do rozsáhlých informačních databází, konference, diskusní kluby, WWW.

Příkladem může být globální celosvětová síť Internet, která propojuje sitě prakticky všech druhů a dnes se stává jedním ze základních komunikačních médií na Zemi.

1.2 Struktura počítačové sítě

V každé síti existuje skupina počítačů určených pro běh uživatelských (aplikačních) programů. Tyto uzly jsou označovány jako hostitelské počítače (*hosts*). Hostitelské počítače jsou propojeny komunikační podsítí (*communication subnet*), jejíž funkcí je přenos zpráv mezi jednotlivými hostitelskými počítači. Oddělení čistě komunikačního hlediska sítě (komunikační podsíť) od aplikačních hledisek (hostitelské počítače) velice zjednodušuje celkový návrh a strukturu počítačové sítě.

Komunikační podsíť je většinou ještě tvořena dvěma komponentami - spoji a propojovacími prvky. **Spoje** jsou vlastní okruhy nebo kanály (v případě WAN) nebo jiná přenosová média jako kroucená dvojlinka, koaxiální kabel popř. optické vlákno (jako je tomu u LAN). **Propojovací prvky** jsou specializovaná zařízení použitá k propojení dvou nebo více spojů (viz obr 1). Když se data objeví na vstupní spoji, musí propojovací prvek zvolit výstupní spoj, po kterém pošle data dál. Tento propojovací prvek se obvykle nazývá **IMP** (*Interface Message Processor*), i když se běžně používají pojmy uzel přepojování paketů (*paket switch node*), mezilehlý systém (*intermediate system*) nebo datová ústředna (*data switching exchange*). Veškerý přenos do a z hostitelského počítače probíhá přes jeho IMP.



Obrázek 1: Struktura počítačové sítě

Existují dva klíčové typy návrhu komunikační podsítě:

- **dvoubodové kanály** (*point-to-point channels*),
- kanály se **všesměrovým šířením** (*broadcast channels*).

V prvním případě (u dvoubodových kanálů) síť obsahuje řadu kabelů nebo pronajatých telefonních spojů, každý z nich propojuje dvojici IMP. Jestliže dva IMP, které spolu nejsou propojeny přímo, spolu chtějí komunikovat, musí být informace přenesena nepřímo přes jiný IMP. Komunikační podsíť pracující na tomto principu se nazývá podsíť s přepojováním paketů a používá se v převážné většině rozlehlých počítačových sítí. Na jeden IMP je možno připojit i více hostitelských počítačů.

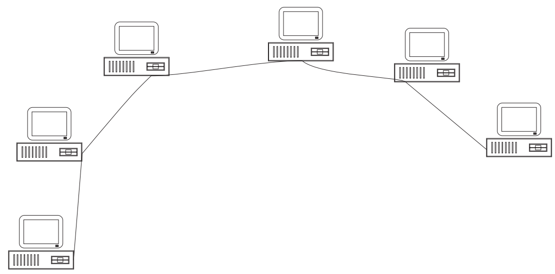
Druhý typ architektury komunikační podsítě je používán převážně u lokálních počítačových sítí. IMP je v tomto případě zredukován na jeden čip vestavěný do hostitelského počítače (resp. síťového adaptéru), takže na jeden IMP je připojen pouze jeden hostitelský počítač.

1.3 Topologie

Topologie sítě je dána způsobem fyzického uspořádání a propojení všech komponent sítě, tj. počítačů, speciálních komunikačních zařízení apod. Používají se tři základní topologie sítí: sběrnicová, hvězdicová, kruhová. Pro úplnost je třeba se zmínit i existenci dalších topologií jako jsou např. smyčka, úplná síť, stromová struktura a páteřní síť.

1.3.1 Sběrnicová topologie (*bus*)

Všechny stanice jsou připojeny na pasivní společné médium, které sdílejí.



Obrázek 2: Sběrnicová topologie

Výhody:

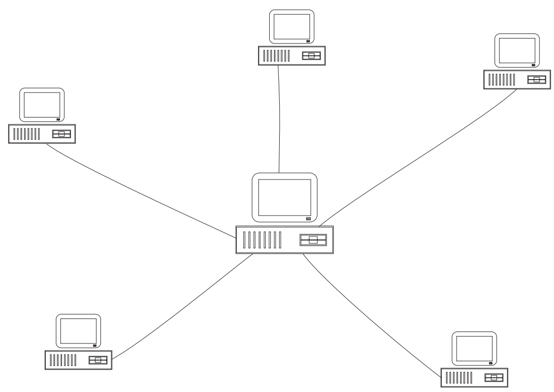
- pasivní médium bez aktivních prvků
- nezávislost stanic na výpadku libovolné jiné stanice
- připojení další stanice bez nákladů (na spojení)
- snadné všesměrové vysílání (*broadcasting*)

Nevýhody:

- úplný výpadek sítě při přerušení kabelu (navíc nutnost zakončení kabelu)
- nutnost realizace vzájemného vyloučenosti na médiu (současně může vysílat pouze jeden uzel)

1.3.2 Hvězdicová topologie (*star*)

Z historického hlediska jde o nejstarší topologii (analogie terminálové sítě). Jedna stanice je centrálním uzlem sítě a přes tuto stanici probíhá věškerá komunikace v síti.



Obrázek 3: Hvězdicová topologie

Výhody:

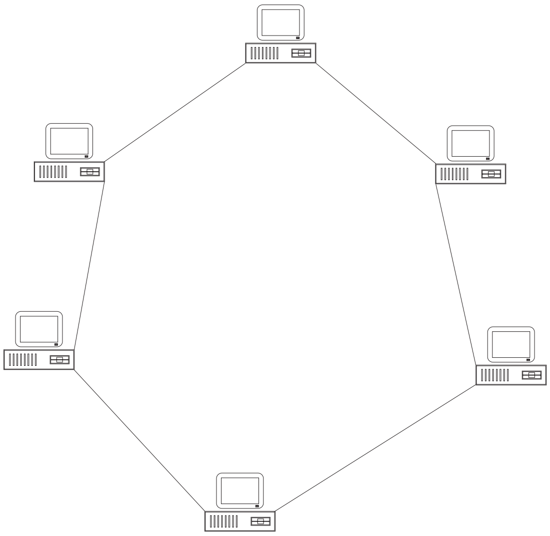
- lehce rozšiřitelná struktura
- výpadek stanice nezpůsobí problémy v síti

Nevýhody:

- výpadek centrálního uzlu znamená výpadek celé sítě
- výkonnost centrálního uzlu je kritickým místem celé sítě
- nesnadná instalace do komplikovaných budov

1.3.3 Kruhová topologie (*ring*)

Každý počítač je propojen přímo s předchozím a následujícím počítačem v kruhu. Zpráva se pohybuje od odesílatele postupně přes všechny sousední uzly ke svému příjemci.



Obrázek 4: Kruhová topologie

Výhody:

- lehce rozšiřitelná struktura
- malý počet spojů
- není centrální uzel

Nevýhody:

- výpadek jedné stanice způsobí rozpad celé sítě (k odstranění se používá princip dvojího kruhu)

1.3.4 Stromová topologie (*tree*)

Jde o rozšíření sběrnicové topologie, kdy pomocí speciálních prvků lze síť rozvětvit a uzly hierarchicky rozčlenit například podle míry vzájemné komunikace (často vzájemně komunikující uzly budou v jedné větvi a jiná skupina bude v jiné větvi). Potom je možné oba segmenty oddělit a propojit speciálním zařízením, které nebude propouštět z jedné větve do druhé informace určené pouze lokální větvi. Takto lze zvýšit nejvyšší možnou komunikaci v síti.

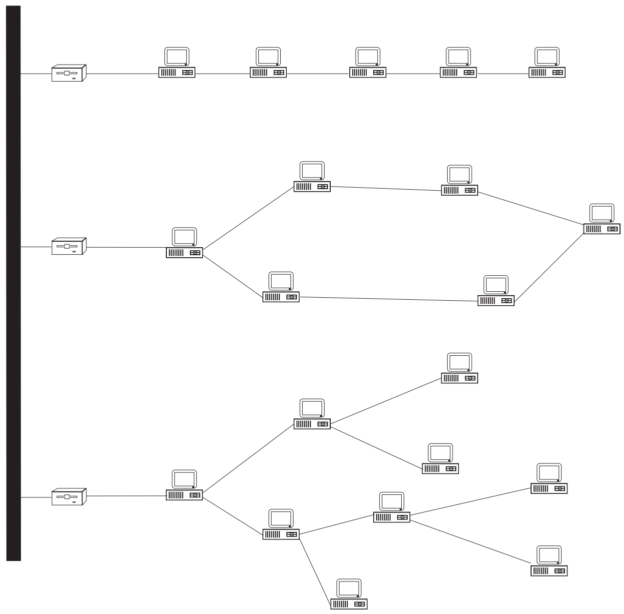
1.3.5 Úplná síť

Každá stanice je propojena s každou jinou. Jde spíše o teoretickou možnost vzhledem k obtížné instalaci a nákladům.

1.3.6 Páteřní síť (*backbone*)

Síť s topologií *backbone* se používá při propojování lokálních sítí nebo při propojování různých typů topologií. Celá síť obsahuje nosnou část (obvykle s rychlým přenosovým médiem), na kterou jsou pak připojeny ostatní části s různou topologií.

Páteřní síť by měla mít o řád vyšší přenosovou rychlost než jednotlivé segmenty. V současné době se používá k implementaci páteřních sítí technologie ATM. Páteřní síť se používá pro spojení strategických uzlů počítačové sítě (důležité komunikační uzly, centrální uzly v zapojení do hvězdy) mezi sebou, přičemž je odstíněna vzájemná komunikace uzlů v podsítích od komunikace zmíněných strategických uzlů.



Obrázek 5: Páteřní síť

Výhody:

- zatížení lokálních větví nezatěžuje páteř
- rychlé spojení počítačů na páteřní síti
- výpadek jedné lokální větve neohrozí zbytek sítě

Nevýhody:

- náklady na páteř a komunikační zařízení na připojení k páteři

1.4 Standardizace počítačových sítí

Původní strategie významných výrobců, nyní označovaná jako orientace na **uzavřené systémy**, spočívala ve vytváření a udržování zpočátku neveřejných firemních specifikací, zajišťujících uživatelům kompatibilitu výhradně vlastních výrobků firmy. Uzavřeným systémům se museli přizpůsobovat drobní producenti technického a programového vybavení, určeného pro zákazníky orientované na určitého výrobce. V některých případech se staly široce respektovanými i produkty vzniklé v rámci univerzitního a vojenského výzkumu. Neřízeným všeobecným rozšířením tak vznikají standardy *de facto*. Standardy *de jure* jsou vytvářeny v organizacích k tomu určených skupinami odborníků z různých firem, vysokých škol a dalších institucí. Někdy vznikají standardy *de jure* i převzetím či přepracováním úspěšných *de facto* standardů.

Otevřený systém je úplný počítačový systém nebo jeho technická či programová část, pro který platí, že je navržený v souladu s uznávanými mezinárodními standardy a lze jej získat od více nezávislých výrobců. Užití otevřených systémů umožňuje, aby programy pracující v lokálních nebo vzdálených systémech sítě mohly navzájem spolupracovat a aby komunikace uživatele s těmito aplikacemi byla jednotná. Otevřené systémy plně respektují otevřené specifikace, což jsou veřejně přístupné popisy dílčích činností, z nichž každá se provádí v souladu s některým mezinárodním standardem. Jinak řečeno, otevřená specifikace je souhrn vybraných mezinárodních standardů. Tento souhrn je úplný v tom smyslu, že činí architekturu specifikovaného systému zjevnou. Přímým důsledkem je modularita otevřeného systému s jasně specifikovanými rozhraními.

Pojem **propojení otevřených systémů** (*OSI, Open Systems Interconnection*) označuje technické a programové vybavení počítačové sítě, resp. systému vzájemně propojených sítí, vyhovující otevřeným specifikacím. Referenční model propojení otevřených systémů, standardizovaný jako ISO 749, je vrstvový model síťové architektury poskytující prostředí pro konkrétní standardy propojení otevřených systémů. Jeho význam spočívá mimo jiné i v zavedení standardní terminologie, která umožňuje popisovat a porovnávat také produkty vytvořené mimo rámec otevřených systémů.

V oblasti počítačových sítí jsou CCITT a ISO nejdůležitějšími mezinárodními organizacemi, které se zabývají tvorbou *de jure* standardů. **CCITT** (*Comité Consultatif International de Télégraphique et Téléphonique*) je jednou ze tří stálých komisí ITU (*International Telecomunications Union*), což je odborná mezinárodní organizace zřízená v rámci OSN. Členy CCITT jsou především vládní instituce nebo spojové organizace, zodpovídající v jednotlivých členských zemích za telekomunikace. CCITT vydává doporučení, která se po schválení ITU stávají mezinárodními standardy. **ISO** (*International Organization for Standardization*) je dobrovolnou nevládní organizací. Jejími členy jsou národní normalizační instituce. Právní závaznosti nabývají až národní normy vypracované obvykle právě na základě mezinárodních standardů či doporučení.

Důležitými jsou kromě národních standardizačních organizací (např. **ANSI** - *American National Standards Institute*) také některé profesní organizace. Příkladem zde může být prestižní americká společnost elektrotechnických a elekronických inženýrů **IEEE** (*Institute of Electrical and Electronics Engineers*). Tato organizace vydává odborné časopisy, pořádá konference a má vlastní orgán zabývající se tvorbou standardů. Pro oblast lokálních počítačových sítí je z nich nejvýznamější skupina standardů IEEE 802, které převzala i organizace ISO jako normu ISO 8802.

1.4.1 Vrstvová struktura (*Layer Architecture*)

Jako motivační příklad vrstvového modelu využijeme klasickou úlohu pana Tanenbauma. Jedná se o posílání filozofických traktátů mezi dvěma filozofy, z nichž jeden žije v Keni ve věži ze slonoviny a druhý v jeskyni v Indii. První mluví pouze svahilsky a druhý ovládá jen jazyk telugu. Oba filozofové využívají pro předávání svých pojednání překladatelů a zprostředkovaně poštovní úředníků, zatímco překladatelé využívají pouze služeb poštovních úředníků.

Filozofové rozumí významu filozofických traktátů. Překladatelé se dohodnou na jazyce, do kterého (resp. ze kterého) budou překládat, a poštovní úředníci se dohodnou na médiu pro přenos. V jednotlivých vrstvách spolu komunikují filozofové na úrovni filozofických traktátů, překladatelé např. anglickými texty a poštovní úředníci např. telegrafem prostřednictvím posloupnosti telegrafických značek.

Informace je však ve skutečnosti předávána vždy mezi filozofem a překladatelem, překladatelem a poštovním úředníkem a mezi poštovním úředníkem a komunikačním médiem. Teprve komunikační médium přenáší signál z Keni do Indie a naopak. Zřejmá je zde relativní nezávislost vrstev. Překladatelů se netýká originalita filozofických úvah a poštovním úředníkům by nemělo záležet na tom, jaký společný jazyk si dohodli překladatelé. Nevhodná volba jazyka může snad snížit kvalitu výsledného filozofického traktátu a volba komunikačního média poštovními úředníky ovlivní z hlediska vyšších dvou vrstev nanejvýš rychlost

Obrázek 6: Model vrstvové architektury

a spolehlivost předávání zprávy. Nabízí se využití tohoto komunikačního systému i pro jiné uživatele, než jsou filozofové, a jeho rozšíření i do jiných míst či zemí, aniž by bylo nutné měnit volbu a uspořádání vrstev.

1.4.2 Referenční model propojování otevřených systémů

V okamžiku, kdy se objevily první významné rozlehlé počítačové sítě, ve kterých se vyskytovaly počítače různých typů od různých výrobců, vyvstala naléhavá potřeba existence jednotného standardu pro vzájemné propojení těchto systémů. V roce 1977 začala ISO vytvářet standard, který měl vycházet z vrstvové struktury. Koncem roku 1979 byl tento standard přijat pod označením *Reference Model of Open Systems Interconnection (RM OSI).*

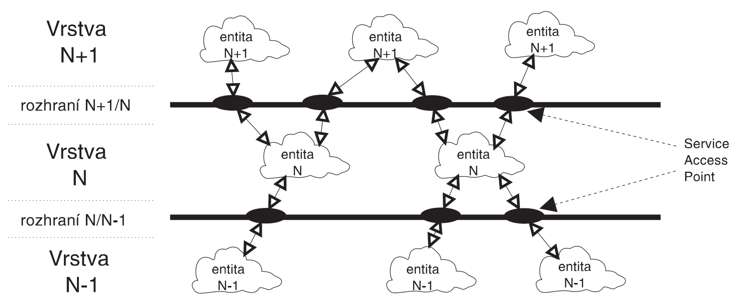
Základní pojmy RM OSI:

Vrstva (*Layer*) je dána přesným vymezením funkcí. Ke každé vrstvě, kromě nejvyšší a nejnižší, přísluší rozhraní s bezprostředně vyšší vrstvou a bezprostředně nižší vrstvou. Nejvyšší vrstvě náleží rozhraní s aplikačním procesem resp. uživatelským programem. Nejnižší vrstva pak sdílí rozhraní s fyzickým médiem.

Entita (*Entity*) je objekt, který v určité vrstvě něco aktivního vykonává. Obvykle se jedná o programový celek. U nižších úrovní může jít i o hardwarový celek (I/O port). Bezprostřední poskytovatel a uživatel služby není vrstva, ale právě určitá entita vrstvy.

Protokol (*Protocol*). Entity ležící ve stejných vrstvách různých otevřených systémů spolu komunikují pomocí předem dohodnutého souboru pravidel s využitím služeb nižších vrstev. Tento soubor pravidel se nazývá protokol.

Služba (*Service*). Entity určité vrstvy vykonáním svých funkcí poskytují služby bezprostředně vyšší vrstvě. Pro realizaci těchto funkcí využívají služeb vrstvy bezprostředně nižší. Jednotlivé služby jsou nabízeny v tzv. **bodech poskytování služby** (*Service Access Point, SAP*), které jsou identifikovány prostřednictvím svých adres.



Obrázek 7: Entity a SAP

Služby, poskytované jednotlivými vrstvami, můžeme rozdělit do dvou skupin:

Spojované služby (*Connection-oriented Services*) pracují následujícím způsobem. Dvě entity na stejných úrovních, které spolu chtějí komunikovat, musí nejprve mezi sebou navázat spojení. Jakmile je toto spojení navázáno, chová se v jistém smyslu jako roura - vysílající do ní na jedné straně vkládá to, co si přeje odeslat, a příjemce si na druhé straně ve stejném pořadí zprávy odebírá.
Jde o obdobu navázání telefonního spojení.

Nespojované služby (*Connectionless Services*) na rozdíl od spojovaných nepočítají se zřízením stálého spojení mezi odesílatelem a příjemcem. Místo toho se považují jednotlivé části přenášených dat (zprávy) za samostatné celky opatřené adresou svého konečného příjemce a doručují se nezávisle na ostatních zprávách. Jednotlivé zprávy mohou tedy být v principu přenášeny různými cestami, což vede nutně k tomu, že nemusí být zachováno pořadí mezi vysílanými a přijímanými zprávami.
Jde o analogii listovní pošty.

Spojovaný charakter mají telefonní sítě a většina veřejných datových sítí a jiných telekomunikačních systémů, zatímco nespojované služby jsou využívány v lokálních počítačových sítích. Původní verze RM OSI vůbec s nespojovanými službami nepočítala a jejich zavedení bylo provedeno až dodatečně.

Spojované služby jsou vhodnější pro přenos větších objemů dat. Mají sice větší jednorázovou počáteční režii (navázání spojení), na druhé straně vykazují menší režii na vlastní přenos dat. Naopak nespojované služby nemají žádnou počáteční režii, mají však relativně vyšší režii na vlastní přenos jednotlivých přenášených částí.

Kromě rozdělování služeb na spojované a nespojované služby je možné i jiné dělení:

Spolehlivá služba (*Reliable Service*) je taková, která nikdy neztrácí žádná data. Obvykle je tato služba realizována prostřednictvím vhodného mechanismu potvrzování (příjemce potvrzuje úspěšné přijetí, resp. znovu žádá o vyslání dat, která byla přijata chybně). Samozřejmě, že s tímto potvrzováním je spojena určitá režie, která nemusí být vždy žádoucí.

Nespolehlivá služba (*Unreliable Service*) není zatížena potvrzováním zpráv. Název je však poněkud zavádějící, jedná se totiž o služby s velmi vysokou spolehlivostí, které však neposkytují stoprocentní záruku úspěšnosti přenosu.

Základní obecné principy, ze kterých se vycházelo při návrhu referenčního modelu byly následující (uvádíme je zde vzhledem k jejich velkému přínosu k pochopení celé koncepce vrstvového modelu otevřeného systému).

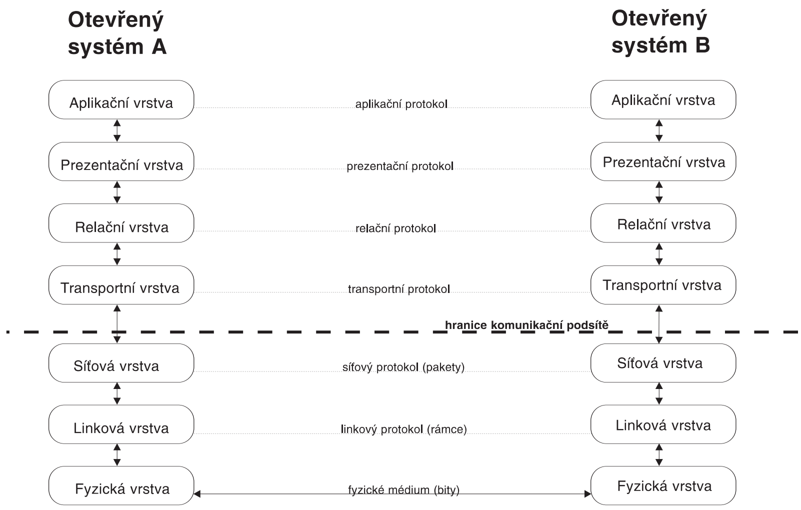
- Samostatná vrstva by měla vzniknout všude tam, kde je zapotřebí jiný stupeň abstrakce.
- Všechny vrstvy by měly provádět přesně definované funkce.
- Funkce všech vrstev by měly být voleny s ohledem na definice mezinárodně normalizovaných protokolů.
- Rozhraní mezi vrstvami by měla být volena tak, aby byl minimalizován tok dat přes tato rozhraní.
- Počet vrstev by měl být tak velký, aby vzájemně odlišné funkce nemusely být zařazovány do stejné vrstvy, a zároveň tak malý, aby celá architektura zůstala dostatečně přehledná.

Výsledkem aplikace těchto principů bylo vymezení sedmi vrstev a specifikace úkolů, které by tyto vrstvy měly zajišťovat.

Fyzická vrstva (*Physical layer*)

Úkol této vrstvy je zdánlivě velmi jednoduchý - zajistit přenos jednotlivých **bitů** mezi příjemcem a odesílatelem prostřednictvím fyzické přenosové cesty, kterou tato vrstva bezprostředně ovládá. K tomu je ovšem třeba vyřešit mnoho úkolů převážně technického charakteru - např. ujasnit, jakou úrovní napětí bude reprezentována logická jednička a jakou logická nula, jak dlouhou „trvá" jeden bit, kolik kontaktů a jaký tvar mají mít konektory kabelů, jaké signály jsou těmito kabely přenášeny, jaký je jejich význam, časový průběh apod.

Problematika fyzické vrstvy proto spadá spíše do působnosti elektroinženýrů a techniků.



Obrázek 8: Referenční model OSI

Linková vrstva (*Data Link Layer*)

Fyzická vrstva poskytuje jako své služby prostředky pro přenos jednotlivých bitů. Bezprostředně vyšší linková vrstva (někdy nazývaná též spojová vrstva nebo vrstva datového spoje) pak má za úkol zajistit pomocí těchto služeb bezchybný přenos bloků dat (velikosti řádově stovek Bytů) označovaných jako **rámce** (*frames*). Jelikož fyzická vrstva nijak neinterpretuje jednotlivé přenášené bity, je na linkové vrstvě, aby správně rozpoznala začátek a konec rámce i jeho jednotlivé části.

Na přenosové cestě může docházet k nejrůznějším poruchám a rušením, v jejichž důsledku jsou přijaté bity jiné než jaké byly původně vyslány. Jelikož se fyzická vrstva nezabývá významem jednotlivých bitů, rozpozná tento druh chyb až linková vrstva. Ta kontroluje, zda byly přeneseny celé rámce správně (posílání různých kontrolních součtů apod.). Odesílateli potvrzuje linková vrstva přijetí bezchybně přenesených rámců, zatímco v případě poškozených rámců si vyžádá jejich opětovné vyslání.

Síťová vrstva (*Network layer*)

Linková vrstva zajišťuje přenos celých rámců, ovšem pouze mezi dvěma uzly, mezi kterými existuje přímé spojení. Jak ale komunikovat, když spojení mezi příjemcem a odesílatelem není přímé a vede přes jeden nebo více mezilehlých uzlů? Pak musí nastoupit síťová vrstva, která zajistí potřebné **směrování** (*routing*) přenášených rámců označovaných nyní **pakety** (*packets*). Síťová vrstva tedy zajišťuje volbu vhodné trasy přes mezilehlé uzly a také postupné předávání paketů po této trase od původního odesílatele ke konečnému příjemci. Síťová vrstva si tedy musí „uvědomovat" konkrétní topologii sítě (tj. způsob vzájemného přímého propojení jednotlivých uzlů).

Transportní vrstva (*Transport Layer*)

Síťová vrstva poskytuje bezprostředně vyšší vrstvě služby zajišťující přenos paketů mezi libovolnými dvěma uzly sítě. Transportní vrstvu tedy zcela odstiňuje od skutečné topologie sítě a vytváří ji tak iluzi, že každý uzel sítě má přímé spojení s kterýmkoli jiným uzlem sítě. Transportní vrstvě se stačí zabývat komunikací koncových účastníků. Při odesílání dat zajišťuje transportní vrstva sestavování jednotlivých paketů, do kterých rozděluje přenášená data, a při příjmu je zase z paketů vyjímá a skládá do původního tvaru. Dokáže tak zajistit přenos libovolně velkých zpráv, přestože jednotlivé pakety mají omezenou velikost.

Relační vrstva (*Session Layer*)

Úkolem této vrstvy je navazovat, udržovat a rušit **relace** (*sessions*) mezi koncovými účastníky (uživatelem a cílovým počítačem). V rámci navazování relace si tato vrstva vyžádá na transportní vrstvě vytvoření spojení, jehož prostřednictvím pak probíhá komunikace mezi oběma účastníky relace. Pokud je třeba tuto komunikaci nějak řídit (např. určovat, kdo má kdy vysílat, nemohou-li to dělat oba účastníci současně) zajišťuje to právě tato vrstva, která má také na starosti vše, co je potřeba k ukončení relace a zrušení existujícího spojení.

Prezentační vrstva (*Presentation Layer*)
Data, která se prostřednictvím sítě přenášejí, mohou mít povahu textů, čísel či obecnějších datových struktur. Jedním úkolem prezentační vrstvy je zavést datové typy, které se budou používat pro komunikaci. Jednotlivé uzlové počítače však mohou používat odlišnou vnitřní reprezentaci těchto dat - např. střediskové počítače firmy IBM používají znakový kód EBCDIC, zatímco většina ostatních pracuje s kódem ASCII. Podobně jeden počítač může zobrazovat celá čísla v doplňkovém kódu, zatímco jiný v kódu přímém. Rozdílný je také přístup k chápání víceslabikové hodnoty (malý a velký endian) - jestli jsou nižší bity uloženy v paměťové buňce s nižší nebo vyšší adresou. Potřebné konverze dat má na starosti právě prezentační vrstva. V rámci této vrstvy bývá realizována také komprese dat, případně i jejich šifrování a zajištění některých dalších služeb bezpečnosti.

Aplikační vrstva (*Application Layer*)
Koncový uživatelé využívají počítačové sítě prostřednictvím nejrůznějších síťových aplikací: systém elektronické pošty, přenos souborů, vzdálené přihlašování (*remote login*) apod. Začleňovat všechny tyto různorodé aplikace přímo do aplikační vrstvy by nebylo rozumné. Proto se do aplikační vrstvy začleňují ty části aplikací, které realizují společné, resp. obecně použitelné mechanismy. Názorným příkladem může být emulace terminálů potřebná pro vzdálené přihlašování. Ve světě dnes existuje nepřeberné množství různých terminálů. Realizovat potřebné přizpůsobení mezi každými dvěma typy je v podstatě nemožné. Proto se zavádí jediný virtuální terminál a pro každý konkrétní terminál se potom vytvoří jen jediné přizpůsobení mezi tímto virtuálním terminálem a terminálem skutečným. Prostředky pro práci s virtuálním terminálem jsou přitom součástí aplikační vrstvy (neboť jsou všude stejné), zatímco prostředky pro přizpůsobení konkrétnímu terminálu již součástí aplikační vrstvy nejsou.

Jiným příkladem může být dobře známý a často používaný systém elektronické pošty.

Zřejmě největší nevýhodou takto předloženého referenčního modelu je, že nespecifikuje žádné konkrétní protokoly, s jejichž pomocí by funkce jednotlivých vrstev měly být realizovány. Nedefinuje dokonce ani přesná rozhraní mezi jednotlivými vrstvami - tedy konkrétní služby, které vrstvy bezprostřeně nižší poskytují vrstvám bezprostředně vyšším. Protokoly a služby jednotlivých vrstev začaly vznikat až dodatečně, a to jako samostatné standardy ISO nebo jako převzaté standardy jiných organizací (např. CCITT, IEEE).

1.4.3 Architektura TCP/IP

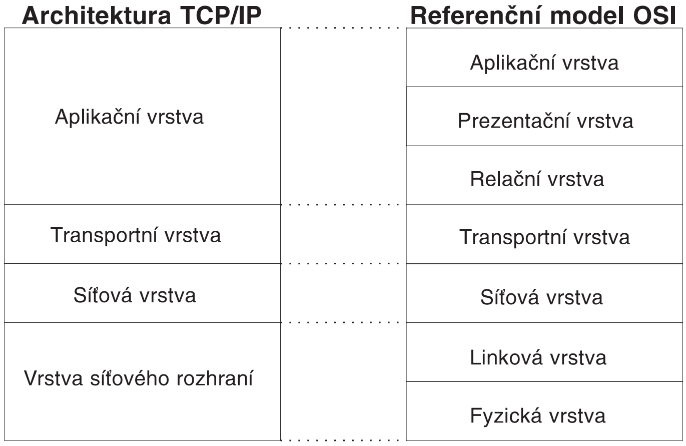
RM OSI je všeobecně považován za síťovou architekturu, která v budoucnosti převládne a stane se dominantní. Není ale zdaleka architekturou jedinou. V současné době má zřejmě největšího konkurenta v soustavě protokolů TCP/IP, které jsou základním kamenem sítě Internet.

Zkratka TCP/IP je obvykle chápána pouze jako označení dvou přenosových protokolů používaných v sítích na bázi operačního systému UNIX. Jsou to protokoly TCP (*Transmission Control Protocol*) a IP (*Internet Protocol*). Ve skutečnosti tato zkratka označuje celou soustavu protokolů ne nutně svázaných se systémem UNIX, přičemž protokoly TCP a IP jsou nejznámější protokoly této soustavy, zdaleka ne však protokoly jediné.

Na rozdíl od RM OSI má architektura TCP/IP pouze čtyři vrstvy:

Vrstva síťového rozhraní (*Network Interface Layer*)
Tato vrstva má na starosti vše, co je spojeno s ovládáním konkrétní přenosové cesty a s přímým vysíláním a příjmem datových paketů. V rámci architektury TCP/IP není tato vrstva blíže specifikována, neboť je závislá na konkrétní přenosové technologii.

Síťová vrstva (*Internet Layer*)
Někdy je nazývána také IP vrstva, vzhledem k použití protokolu IP. Úkol této vrstvy je stejný jako úkol síťové vrstvy RM OSI - stará se o to, aby se jednotlivé pakety dostaly od odesílatele až ke svému skutečnému příjemci bez ohledu na to, zda mezi nimi existuje přímé spojení nebo ne. Vzhledem k nespojovanému charakteru přenosu (protokol IP) je na úrovni této vrstvy zajišťována jednoduchá datagramová služba.



Obrázek 9: Vztah architektury TCP/IP a RM-OSI

Transportní vrstva (*Transport Layer*)
Často též TCP vrstva opět podle používaného protokolu. Hlavním úkolem této vrstvy je zajistit přenos mezi dvěma koncovými účastníky, kterými jsou v případě architektury TCP/IP aplikační programy. Podle jejich požadavků může transportní vrstva regulovat tok dat oběma směry, zajišťovat spolehlivost přenosu a také měnit nespojovaný charakter přenosu (v síťové vrstvě) na spojovaný. Přestože je transportní vrstva zajišťována především protokolem TCP, existují i další způsoby komunikace. Například protokol UDP (*User Datagram Protocol*), který nezajišťuje spolehlivost přenosu (na rozdíl od TCP) pro aplikace, které si to na úrovni transportní vrstvy nepřejí.

Aplikační vrstva (*Aplication Layer*)
Nejvyšší vrstvou architektury TCP/IP je aplikační vrstva, jejímiž entitami jsou jednotlivé aplikační programy. Ty, na rozdíl od RM OSI, komunikují přímo s transportní vrstvou. Případné prezentační a relační služby si aplikační programy zajišťují samy.

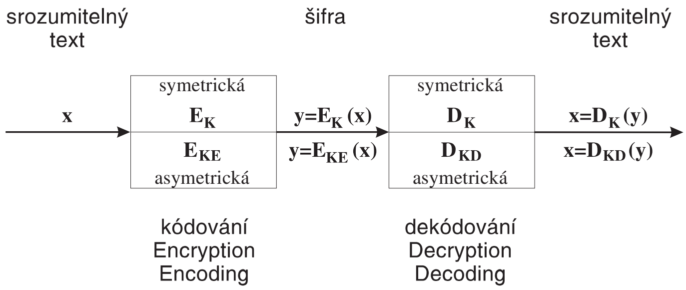
2 Kódování informací

Přenos dat prostřednictvím počítačových sítí musí být chráněn před neoprávněným přístupem cizích osob nebo programů stejně jako přenos dat klasickou cestou (dopis). Přitom je nutné zajistit podmínky bezpečnosti spojené se samotným přenosem (přenášet informace v takovém tvaru, aby nebyly běžně čitelné) i vyloučit takzvaný zprostředkovaný přístup k datům (tím myslíme zamezit klasickým útokům na bezpečnost známým z nepočítačového světa - trezory, utajení existence dat). Na tomto místě se zaměříme na to, jak lze data zabezpečit na úrovni jejich zdrojového tvaru - jak data ,,znečitelnit".

Ochranou informací na výpočtové úrovni se zabývá **kryptografie**. Hledáme-li vrstvu v referenčním modelu OSI, která se zabývá šifrováním přenášených informací, pak to bude prakticky vždy některá vrstva nad relační vrstvou (klasicky prezentační vrstva).

Uveďme základní pojmy, se kterými kryptografie pracuje. Úkolem kryptografické služby je převést srozumitelný text **M** pomocí operace **šifrování** (kódovací funkce encode **E**) na šifrovaný text **E(M)**. Ze šifrovaného textu lze obvykle získat operací **dešifrování** (dekódovací funkce decode **D**) zpět srozumitelný text. [2] Platí tedy implikace $C = E(M) \Rightarrow M = D(C)$. Velmi často se používají kódovací a dekódovací funkce, které jsou parametrizované snadno zaměnitelným **klíčem**, který prohlásíme za tajný a tím velmi znatelně znesnadníme odhalení srozumitelného textu ze šifry. Přitom algoritmy kódovacích a dekódovacích šifer jsou většinou veřejně známy a dlouhodobě používány. Bezpečnost použité metody je pak dána především bezpečností použitého klíče.

Kryptografie může být založena na dvou různých principech - **symetrickém** a **asymetrickém šifrování**. Při symetrické kryptografii je ve funkci E i D použit stejný klíč, zatímco asymetrická kryptografie používá v obou funkcích různé klíče. Na obrázku 10 jsou schematicky ukázány významy používaných označení.



Obrázek 10: Princip kryptografie

- Symetrická kryptografie používá klíč K jako tajný klíč a pro její šifru y a srozumitelný text x platí $y = E_K(x)$, $x = D_K(E_K(x))$.
- Asymetrická kryptografie používá dva klíče: KE (pro kódování) a KD (pro dekódování). Jestliže je výpočetně složité odvodit ze známého klíče KE klíč KD, hovoříme o šifrování s *veřejným klíčem KE*, který potom označujeme K_V a dešifrování *privátním klíčem KD*, který označujeme K_P.
  Jestliže je naopak výpočetně složité odvodit z KD klíč KE, hovoříme o mechanismu digitálního podepisování a $K_V = KD$, $K_P = KE$.

2.1 Teorie informace

Pro pochopení podstaty kódování zprávy je důležité pochopit, co vlastně ukrýváme. Ne každá zpráva pevné délky v sobě totiž nese stejné množství informace. Hodně záleží na jazyku, ve kterém je zpráva zapsána.

[2] Slovo ,,obvykle" zde vyjadřuje možnost tzv. jednocestného kódování, které se používá pro ukládání hesel nebo dalších tajných informací. U jednocestných kódování je prakticky nemožné z šifry vypočítat srozumitelný text.

Množství informace ve zprávě je minimální potřebný počet bitů potřebných k zakódování této zprávy. Co si pod tímto pojmem představujeme jistě pochopí návrháři databázových položek. Například dny v týdny můžeme reprezentovat buď řetězcovými zkratkami {Po, Út, St, Čt, Pá, So, Ne} nebo bitovými identifikátory {000, 001, 010, 011, 100, 101, 110}. Zatímco první metoda vyžaduje k vyjádření jednoho dne ze sedmi dnů v týdnu dvou bytů (dva znaky), druhá metoda vystačí se třemi bity. Ve sdělení dne v týdnu jsou tedy obsaženy tři bity informace.

Množství informace ve zprávě jsme tedy převedli na počet bitů, které potřebujeme k zápisu zprávy. Tento počet bývá označován také jako **entropie zprávy** a je definitoricky stanoven jako $H(M) = log_2 n$, kde n označuje počet možných významů zprávy M. Přitom předpokládáme rovnoměrnou distribuci významů zprávy M.

Významnou veličinou při šifrování je tzv. **neurčitost zprávy**, která označuje počet bitů srozumitelného textu, které je nutno obnovit k rozpoznání srozumitelného textu, známe-li jeho šifru.

Objasníme nyní, jak může jazyk zprávy výrazně ovlivnit složitost a výsledek šifrování. Teorie říká, že **vyjadřovací síla jazyka zprávy** je dána poměrem její entropie a délky. Jednotkou vyjadřovací síly je tedy počet bitů na znak. Na základě fundovaných odhadů pro dlouhé zprávy vyplývá, že vyjadřovací síla anglického jazyka je přibližně $1,2$ bitu na znak. Přitom podle klasického výpočtu je absolutní vyjadřovací síla (entropie) jednoho znaku $R_A = log_2 26 = 4,7$bit/znak. Výsledkem těchto úvah je překvapivé zjištění, že každý znak anglické věty nese $1,2$ bitu informace a $3,5$ bitu redundantní informace.

Je pochopitelné, že každá šifra v sobě nese nějaký odraz srozumitelného textu. Proto se snažíme minimalizovat ve srozumitelném textu vše, co není nutné k vyjádření jeho významu. Jinými slovy - - snažíme se minimalizovat redundanci jazyka, kterým je srozumitelný text zapsán a znesnadnit tak práci kryptoanalýze. Proto bývá před samotným šifrováním zpráva komprimována algoritmy, o kterých se zmíníme dále.

2.2 Metody symetrické kryptografie

Existují dvě základní skupiny metod symetrické kryptografie, obě založeny pochopitelně na utajování společného klíče: substituční a transpoziční metody.

2.2.1 Substituční metody

Základní vlastností substituční metody je, že nahrazuje každý symbol srozumitelného textu jiným symbolem nebo skupinou symbolů. Příkladem je klasický *Caesarův šifrovač*, který nahrazuje každý symbol zprávy symbolem o tři pozice vyšším (A→D, B→E, ..., X→A, ...). Klíčem pak může být číslo, o které se má inkrementovat index symbolu zprávy modulo počet symbolů abecedy. Těchto čísel je ale ve znakové abecedě pouze 26, což umožňuje snadný útok hrubou silou (otestování všech možných klíčů).

Obecně lze navrhnout *monoalfabetický šifrovač* založený na překladové tabulce, která stanovuje překlad každého symbolu abecedy. Příkladem může být následující tabulka:

A B C D E F G H I ... O P Q R S T U V W X Y Z
Q A D E W S Z X C ... T Y H N M J U I K L O P

Její druhý řádek je vlastně klíčem šifry. Vzhledem k počtu klíčů je obecný šifrovač mnohem více chráněn proti odhalení klíče metodou hrubé síly (klíčů je 26!). Pokud by útočník potřeboval k otestování jednoho klíče $1\mu s$, pak v nejhorším případě by trvalo odhalení šifry 10^{13} let. Metoda však překládá symboly textu vždy na stejné symboly šifry, takže se nezakrývá například frekvence písmen, která je v daném jazyce známá a může výrazně pomoci při útoku.

Právě odstranění možného útoku z pozice frekvenční analýzy řeši tzv. **polyalfabetický šifrovač**. Ten používá několik abeced současně a tím zastírá identitu jednotlivých symbolů. Známý je například *Vigenérův šifrovač*, který používá pro zobrazení mezi abecedami srozumitelného a šifrovaného textu čtvercovou matici cyklicky posunutých řádků s prvky abecedy srozumitelného textu. Používá tedy tolik různých abeced, kolik je symbolů abecedy srozumitelného textu. Klíčem je libovolné slovo, které napíšeme nad srozumitelný text a svými symboly (znaky) určuje, který řádek tabulky se má použít pro kódování příslušného symbolu zprávy napsaným pod symbolem klíče. Podle následující tabulky zašifrujeme text „TAJNAZPRAVA" s klíčem „XY".

	A	B	C	D	E	F	G	H	I	...	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	...	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	...	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	...	Q	R	S	T	U	V	W	X	Y	Z	A	B
...																						
X	X	Y	Z	A	B	C	D	E	F	...	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	...	M	N	O	P	Q	R	S	T	U	V	W	Y
Z	Z	A	B	C	D	E	F	G	H	...	N	O	P	Q	R	S	T	U	V	W	Y	Z

Napíšeme text zprávy a nad ni opakovaně klíč. Znak klíče nám určí řádek tabulky, znak zprávy sloupec a jejich průsečík v tabulce je přeložený znak šifry. Za zmínku stojí jedna nevýhoda, která vyplývá z pravidelnosti opakování klíče. Ta totiž ve svém důsledku u dlouhé zprávy dává zelenou frekvenční analýze.

klíč →	X	Y	X	Y	X	Y	X	Y	X	Y	X
zpráva →	T	A	J	N	A	Z	P	R	A	V	A
šifra →	Q	Y	G	L	X	Y	M	P	X	T	X

Další šifrovač, který je používán například na hot-line spojení Moskva-Washington, používá tzv. **jednorázový klíč**. Jednorázovým klíčem bývá zpravidla náhodný bitový vzorek delší než samotná zpráva, který se spojí se zprávou funkcí nonekvivalence (*xor*). Na straně příjemce je opětným použitím stejného klíče zpráva rozšifrována. Tento způsob šifrování je bez znalosti klíče prakticky nerozluštitelný.

2.2.2 Transpoziční metody

Transpoziční metody kryptografie mění pořadí symbolů ve zprávě, ale nemění symboly samotné. Příkladem může být zapisování zprávy po znacích do řádků matice, která má počet sloupců roven počtu znaků klíče. Z této matice je pak čtena šifra po sloupcích, a to ještě v jiném pořadí sloupců. Uvedené pořadí se zjistí z klíče (první sloupec je index znaku klíče, který je nejblíže začátku abecedy atd.). Následující tabulka ukazuje způsob zašifrování zprávy „TAJNA ZPRAVA" pomocí klíče „KLIC" na šifru „NPAJZVTARA A".

klíč →	K	L	I	C
pořadí →	3	4	2	1
zpráva →	T	A	J	N
	A		Z	P
	R	A	V	A

2.2.3 Norma DES

Použité techniky kódování založené na úplném utajení klíčů jsou poměrně jednoduché, ovšem většinou existuje možnost jejich pirátského dekódování (s použitím počítače). Pro přenos dat počítačovou sítí musí být zaručeno, že nebude možné dekódovat data ani s použitím výpočetní techniky. Již v roce 1977 byla přijata norma pro kódování dat **DES** (Data Encryption Standard), což je v podstatě monoalfabetický šifrovač, který dělí zprávu na 64bitové úseky. Každý úsek zprávy je kódován použitím 54bitového klíče, přičemž zpracování používá iniciální permutaci bitů zprávy následovanou cyklem šestnácti transpozic a substitucí bitů zprávy. Podrobný popis metody DES může čtenář najít například v knize [6] Pravdou ovšem je, že norma DES nedefinuje způsob distribuce tajných klíčů, uveďme tedy několik známých způsobů distribuce tajných klíčů.

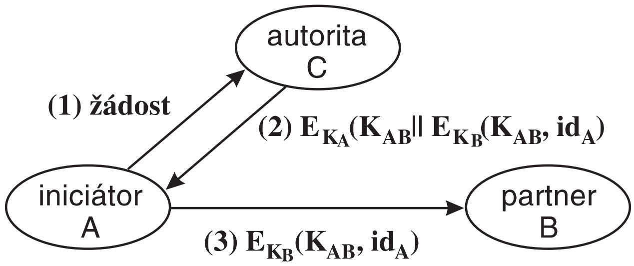
2.3 Distribuce tajných klíčů symetrické kryptografie

Předpokládejme, že v distribuovaném systému existují uzly A a B, které spolu komunikují a používají k šifrování symetrických metod kryptografie. Může se jednat například o banku a jejího klienta, kteří komunikují po modemu nebo počítačové sítí a tajným klíčem šifrují přístupové heslo, příkazy k provedení bankovní operace a výpisy z konta.

V jistém okamžiku vznikne potřeba zvolit nový tajný klíč K pro komunikaci mezi těmito dvěma uzly. Uzly mají k dispozici následující varianty řešení (první dvě vyloučíme, protože předpokládají jinou cestu přepravy klíče, než jakou je dopravována zpráva).

- A zvolí klíč K a dopraví ho osobně B.
- Klíč K zvolí třetí strana a opět osobně ho dopraví oběma zbývajícím.
- Jestliže A a B používají klíč K', zvolí uzel A nový klíč K a pošle B tento nový klíč zakódovaný $E_{K'}(K)$ s použitím starého klíče K'. Při této variantě je nutno mít na vědomí, že na samotném počátku komunikace je potřeba dodat každým dvěma komunikujícím uzlům jeden klíč (tj. $\binom{n}{2}$ klíčů). Navíc v případě jednoho odhalení klíče útočníkem je znám také každý následující klíč.
- Jestliže A a B již používají klíče K_{AC} a K_{BC} pro komunikaci se třetí stranou C, zvolí nový klíč K_{AB} strana C a oznámí ho uzlu A zakódovaný jako $E_{K_{AC}}(K_{AB})$ a uzlu B jako $E_{K_{BC}}(K_{AB})$.
- Existuje-li centrální uzel s evidencí ,,uzlových klíčů" jednotlivých uzlů, lze použít distribuci klíčů podle obrázku 11. Uzlové klíče jsou známy pouze centrální autoritě a samotným uzlům, každý uzel má svůj jedinečný uzlový klíč. Nechť například uzel A iniciuje požadavek vytvořit klíč pro komunikaci s uzlem B. Zašle požadavek (1) autoritě C, ta vytvoří klíč K_{AB}, sestaví zprávu pro A z klíče K_{AB} a $E_{KB}(K_{AB}, idA)$. Operátor || v obrázku znamená zřetězení. Tuto zprávu před odesláním zakóduje (2) uzlovým klíčem A. Uzel A přečte nový klíč, zašle uzlu B (3) položku $E_{KB}(K_{AB}, idA)$, kterou dovede rozšifrovat pouze uzel B se znalostí klíče KB.

  Tím oba získají klíč pro novou komunikaci a mohou provést případné autentizační kroky. Ty spočívají ve výměně svých identifikátorů zakódovaných novým klíčem.
- Používaná je také metoda hierarchických klíčů - pro jednotlivé úrovně hierarchie systému se používají samostatné klíče. V hierarchicky členěných organizacích se tato metoda může osvědčit.



Obrázek 11: Distribuce klíčů symetrické kryptografie

2.4 Asymetrická kryptografie

Výrazně jiný přístup k utajování klíčů přináší myšlenka asymetrické kryptografie. Základní změnou oproti symetrickým metodám je existence různých klíčů pro kódování a dekódování zprávy a odtajnění jednoho z nich. Algoritmy pro kódování (E) a dekódování (D) splňují následující kritéria:

- Dekódováním zprávy E(x) získáme opět zprávu x: x=D(E(x)).
- Algoritmy E a D jsou výpočetně jednoduché.

- Nelze efektivně vypočítat D ani při znalosti E a původní zprávy i šifry.

Komunikující uzly zveřejňují kódovací klíč algoritmu E, který nazýváme veřejný klíč (K_V) a utajují kódovací klíč dekódovacího algoritmu (ten nazýváme tajný-privátní klíč K_P). Libovolný uzel tedy může zjistit veřejný klíč K_{VA} uzlu A, pomocí něho zakódovat svoji zprávu $E_{K_{VA}}(x)$ a zaslat ji uzlu A. Ten pomocí svého privátního klíče K_{PA} zprávu dekóduje. Analogicky zašle zpět odpověď.

Zveřejněním kódovacích klíčů se podstatně zjednoduší kódování zpráv (komunikace s autoritou odpadá), ale zvyšují se paměťové nároky (pamatují se veřejné klíče).

Uveďme nyní jeden z klasických algoritmů používaný k asymetrickému šifrování. Vzhledem k tomu, že tento algoritmus využívá závěrů vyšší matematiky, některé vlastnosti předložíme k uvěření a nedůvěřivého čtenáře odkazujeme na příslušnou literaturu (např. [6]).

Kódovací funkce:		$C = M^e \ (mod\ n)$,
dekódovací funkce:		$M = C^d \ (mod\ n)$,
přitom platí		n = p.q
		m = (p-1)·(q-1)
		p, q jsou volena jako velká prvočísla ($> 10^{100}$)
	d	je číslo nesoudělné s číslem m
	e	je řešením rovnice $e \cdot d = 1\ mod\ m$

Veřejně jsou známy hodnoty kódovacího klíče $\{e, n\}$, utajena je hodnota dekódovacího klíče $\{d, n\}$. Zprávy určené příjemci se správným dekódovacím klíčem tedy může kódovat každý, bez tohoto klíče však dekódovat zprávu prakticky nelze. Důvodem je extrémní složitost výpočtu d ze známých hodnot n a e i za použití počítače. Při pokusu o výpočet d by musel útočník zvládnout rozklad velkého čísla na prvočinitele, což právě způsobuje problém. Například při volbě $p > 10^{100}$ vychází n řádově 10^{200}, což při přepočtu vychází 2^{664}. Protože v podobném řádu zůstává i hodnota m která určuje modulo, může se zpráva rozdělit na úseky dlouhé až 83 slabik. V porovnání s osmi slabikami DES šifry je rozdíl patrný.

2.5 Podpisování

Prostřednictvím asymetrické kryptografie lze zasílat zprávy nejen zakódované, ale také podepsané. Tak je možné ověřit, jestli zprávy přicházejí opravdu od klienta, od kterého jsou očekávány. Potřeba takového aparátu je zřejmá v bankovních operacích, ale s nárůstem použití počítačové pošty k nejrůznějším transakcím je reálná potřeba této kontroly i v dalších oblastech. Náhrada skutečného podpisu v běžném světě je tedy tzv. **digitální podpis**, který vedle funkce ověřovací má také funkci průkaznou (znemožnit popření autorství). Digitální podpis musí být závislý na obsahu zprávy a na identitě odesílatele.

Následující tabulka ukazuje tři základní typy podpisování zpráv. Ve všech třech případech zasílá uzel A (veřejný klíč K_{VA}, privátní klíč K_{PA}) zprávu x uzlu B (veřejný klíč K_{VB}, privátní klíč K_{PB}) zakódovanou jako y.

zakódovaná zpráva	**typ zprávy**	**dekódování**
$E_{K_{VB}}(x)$	tajná	$D_{K_{PB}}(y)$
$E_{K_{PA}}(x)$	podepsaná	$D_{K_{VA}}(y)$
$E_{K_{VB}}(E_{K_{PA}}(x))$	tajná podepsaná	$D_{K_{PB}}(D_{K_{VA}}y)$

3 Komprese dat

Přenos dat počítačovou sítí je technicky zajišťován komunikačními kanály, jejichž jednou vlastností je **propustnost**, tedy množství dat přenesených za jednotku času (například Mbit/s). Propustnost a spolehlivost pak určují reálné množství dat, které je schopna komunikační podsíť přenést. Vzhledem ke stále rostoucímu množství přenášených dat je stále aktuální potřeba minimalizování velikosti zpráv předkládaných komunikačnímu kanálu k přenesení. Služba **komprese dat** tedy kóduje zprávu tak, aby byla minimalizována její délka (měřeno v počtech bitů) při zachování obsahu. S kompresí se setkáváme prakticky v každém přenosu dat - při přenosu sítí, při archivaci a distribuci software. Hledáme-li opět vrstvu v referenčním modelu OSI, která se zabývá kompresí přenášených zpráv, pak je zvykem tyto operace zajišťovat prezentační vrstvou.

Jak již bylo naznačeno, při kompresi dochází k přepisu zprávy do jiného (kratšího) tvaru při zachování původního obsahu. Na otázku, jak mnoho je možné zprávu zakódovat a které informace můžeme komprimovat, odpovíme následující krátkou exkurzí do teorie. S pojmem **neurčitost** jsme se seznámili již v pasáži o kódování. Určuje souvislost mezi počtem bitů, které zpráva v sobě nese, a množstvím informace přenášené zprávou. Jestliže pravděpodobnost výskytu zprávy z je p, potom míra neurčitosti nesené zprávou je $ln\ p$. Vidíme, že míra neurčitosti roste s klesající pravděpodobností výskytu dané zprávy. Míra neurčitosti zprávy, kterou si vyměňují dva procesy, je dána jednak pravděpodobnostními charakteristikami výskytů prvků abecedy a statistickými závislostmi symbolů ve zprávě. Jsou-li tyto symboly ve zprávě na sobě nezávislé a p_i jsou pravděpodobnosti výskytu prvků abecedy, pak průměrné množství informace jedné zprávy je

$$H_n = -n \cdot \sum p_i \cdot ln\ p_i\ .$$

Pokusme odpovědět na otázku: „Kdy dosahují zprávy největší neurčitosti?" Bude to v případě, že všechny symboly se v nich budou vyskytovat se stejnou pravděpodobností (v případě, že abeceda má m znaků) $\frac{1}{m}$. Potom $H_n = n \cdot ln\ m$. Tím je také naznačen jeden ze základních trendů při kompresi - rovnoměrné rozložení prvků abecedy ve zprávách. Je možné například často se opakující zprávy nahradit co nejkratším překladem a naopak málo se vyskytující zprávy kódovat delšími řetězci. Z existence jisté maximální míry neurčitosti (ideální stav) můžeme odvodit **mez možného zkrácení**, ke které se však většinou pouze blížíme.

Rozdílný přístup ke kompresi dat vyžadují dvě skupiny zpráv, a to zprávy konečně dlouhé a potenciálně nekonečné zprávy. V případě konečnosti dat lze optimalizovat datové typy a rozsahy položek, ve kterých budou data zpracovávána, stejně jako přesun dat prostřednictvím sítě (nechat migrovat menší položky k větším a ne naopak). Jestliže vyměňované zprávy tvoří nekonečnou posloupnost zpráv, nelze předem tvrdit nic o jejich počtu ani rozsahu. Mnohdy jsou ale známy například pravděpodobnosti, se kterými se vyskytují ve zprávách jednotlivé symboly (je známa frekvence jednotlivých znaků v abecedách různých jazyků). Lze pak použít s úspěchem **frekvenčně závislé kódování** založené na přepisu častěji používaných prvků abecedy na kratší bitové posloupnosti a řidčeji se vyskytující prvky na delší bitové posloupnosti.

3.0.1 Frekvenčně závislé kódování

Na konkrétním příkladu ukážeme použití této metody. Mějme abecedu $\{A, B, C, D\}$ a pravděpodobnosti výskytu jednotlivých znaků určeny jako $p_A = 0.6$, $p_B = 0.29$, $p_C = 0.1$ a $p_D = 0.01$. Úkolem je zakódovat zprávu „ABAABC". Vypočítáme střední počet bitů nesených jedním znakem jako $\frac{H_n}{n}$, tedy $-0.6 \cdot ln_2 0.6 + (-0.29 \cdot ln_2 0.29) + (-0.1 \cdot ln_2 0.1) + (-0.01 \cdot ln_2 0.01) = 1.36$. Kdybychom kódovali každý znak dvěma bity podle následující tabulky (vlevo) bez ohledu na pravděpodobnosti výskytu, vyšel by střední počet bitů nesených jedním znakem $0.6 \cdot 2 + 0.29 \cdot 2 + 0.1 \cdot 2 + 0.01 \cdot 2 = 2$, a zpráva by byla kódovaná na 12 bitů jako `000100000110`. Podle tabulky vpravo vychází tato veličina $0.6 \cdot 1 + 0.29 \cdot 2 + 0.1 \cdot 3 + 0.01 \cdot 3 = 1.51$ a zpráva by byla kódovaná na 10 bitů jako `0100010110`.

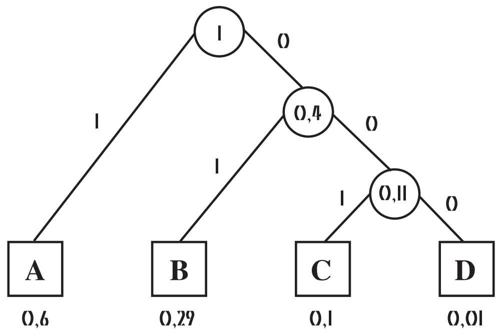
A	B	C	D
00	01	10	11

A	B	C	D
0	10	110	111

3.0.2 Huffmanovo kódování

Jiný způsob zobrazení jednotlivých symbolů do bitových posloupností nabízí Huffmanovo kódování. To je založeno na kódování symbolů nezávisle na sobě. Velmi stručně lze princip Huffmanova kódování shrnout do následujích několika vět. Vypíšeme symboly i s ohodnoceními vedle sebe do posloupnosti setříděné například sestupně. Pak vytváříme binární strom, pro který platí, že v listech budou právě symboly, konstrukce začne u symbolu s nejnižším ohodnocením a vnitřní uzly budou ohodnoceny součtem ohodnocení jejich synů. Odtud plyne, že ohodnocení kořene stromu bude 1. Pak všechny pravé hrany označíme 0 a levé 1. Kód symbolu bude dán ohodnocenou cestou ve stromu od kořene k danému symbolu v listu stromu.

Typickým kompresním poměrem Huffmanova kódování je 80% (na 80% délky). Musíme mít na paměti, že je potřeba přenášet také kódovací strom (cca 1KB). Pro předcházející příklad můžeme vytvořit Huffmanův kódovací strom způsobem zachyceným na obrázku 12.



Obrázek 12: Kódovací strom

3.0.3 Aritmetické kódování

Aritmetické kódování je metoda, která celou zprávu kóduje do reálného čísla z intervalu $< 0, 1)$. Také tato metoda vyžaduje na straně kódovače i dekódovače znalost tabulky pravděpodobností výskytů jednotlivých znaků abecedy ve zprávě. Každý symbol zprávy pak reprezentuje jako číslo na reálné ose. Postupným zjemňováním intervalu $< 0, 1)$ v poměru pravděpodobností výskytů znaků abecedy dospějeme až k výslednému kódu. Celou metodu nejlépe pochopíme z následujícího příkladu.

Mějme dánu abecedu $\{A, B, C, D\}$ a pravděpodobnosti výskytu jednotlivých znaků určeny jako $p_A = 0.4$, $p_B = 0.3$, $p_C = 0.1$ a $p_D = 0.2$. Úkolem je zakódovat zprávu ,,DAC", při jeho řešení postupujeme v následujících krocích. Interval $< 0, 1)$ rozdělíme čtyřmi dělícími body podle pravděpodobností výskytu znaků A, B, C a D. Uvažujeme nyní první znak zprávy – D. Znaku D náleží interval $< 0.8, 1)$, který rozdělíme opět podle pravděpodobností výskytu znaků na čtyři podintervaly. Uvažujme druhý znak zprávy - A, tomu přísluší interval $< 0.8, 0.88)$, který rozdělíme podle pravděpodobností. Poslednímu znaku zprávy (C) náleží interval $< 0.856, 0.864)$, jehož střed udává právě šifrovanou zprávu 0.86.

Dešifrování probíhá opačným směrem. Známe-li šifu (0.86), délku původní zprávy a tabulku pravděpodobností výskytu, jsme schopni dešifrovat takto:

- $0.86 \in< 0.8, 1.0)$, tomuto intervalu náleží znak D
- $0.86 \in< 0.8, 0.88)$, tomuto intervalu náleží znak A, mezivýsledek DA
- $0.86 \in< 0.856, 0.864)$, tomuto intervalu náleží znak C, zpráva DAC

Přitom upozorňujeme na nutnost znát délku původní zprávy. Objasněte proč.

3.0.4 Kontextově závislé kódování

Kontextově závislé kódování je další metoda, která se používá pro kódování předem neznámého počtu zpráv. V principu jde o přepsání zprávy do jiné zprávy, ve které připadá na jeden symbol větší střední množství informace. Přitom se vychází z kontextu zprávy. Obecně lze uvažovat o pravděpodobnostech výskytu každého prvku abecedy po jiném prvku abecedy, ale vzhledem k velikosti pravděpodobnostních tabulek je takto obecně pojatá metoda neakceptovatelná.

3.0.5 RLE - Čítače opakovaných prvků

Nejjednodušší kompresní metodou využívající kontextovou závislost je přepis opakujících se symbolů za tento symbol a specifikaci počtu opakování. Jestliže omezíme počet prvků abecedy na dva (binární abeceda) a převažuje-li výskyt nul ve zprávě, pak lze kódovat například podle následující strategie: místo specifikace počtu opakujících se znaků (nul) mezi jedničkami kódujeme jejich počet pomocí tříbitových symbolů (pro větší počet se použije dohoda, že znak 111 znamená počet $2^3 - 1$ plus hodnota následujícího tříbitového symbolu (opět podle této dohody). Následující zpráva

00010000100000000000010000011

(29 symbolů)
bude zakódována jako

011 100 111 101 101 000

(18 symbolů).
Dosáhneme tím 38% úspory.

3.0.6 Vysílání diferencí zpráv

Jestliže budou zprávy následovat za sebou s minimální změnou obsahu informací (sekvence obrázků, film), je výhodné posílat pouze změny mezi jednotlivými zprávami a po čase celou zprávu z důvodu chyb při přepočtech. Tuto službu může provádět prezentační vrstva, která může rozhodnout o povaze zpráv.

3.0.7 Slovníková komprese (LZW)

Slovníková (LZW=Lempel-Ziv-Welsch) komprese je založena na následující velmi jednoduché strategii. Algoritmus postupně rozpoznává a ukládá do tabulky řetězce znaků a tyto řetězce nahrazuje ve výstupním textu přirozenými čísly z předem definovaného intervalu. Tento interval určuje výslednou abecedu a platí pro něj, že například při kódování řetězce znaků zobrazených v osmibitovém zobrazení je prvních 255 čísel vyhrazeno pro zobrazení samostatných znaků z původní abecedy. Čísla nad 255 se pak přidělují jednotlivým nalezeným řetězcům. Přitom se vytváří tabulka (slovník) již rozeznaných řetězců. Naznačme princip této metody algoritmem zapsaným v pseudojazyce:

```
GET(řetězec)
while not EOF(vstupní_soubor)
  loop
    GET(znak)
    if (řetězec+znak) in tabulka_řetězců
     then řetězec += znak
     else PUT(výstupní_soubor,
              číslo(řetězec, tabulka_řetězců))
          ADD(řetězec, tabulka_řetězců)
          řetězec := znak
    fi
  endloop
PUT(výstupní_soubor, tabulka_řetězců)
```

Uvedená metoda má ovšem několik nevýhod, které vyplývají z rychlého nárůstu prvků nové abecedy, v případě zaplnění tabulky se „zapomínají" některé známé řetězce. V případě větší tabulky však narůstá doba vyhledání řetězce v tabulce.

4 Přenos dat

V počítačových sítích se setkáváme s nejrůznějšími formami přenosu signálů, které mohou být různým způsobem modulovány, kódovány, a k jejich přenosu se mohou používat přenosové kanály různých vlastností. Následující pasáž se bude zabývat právě různými druhy přenosů dat (signálů).

4.1 Směr přenosu

V podstatě existují tři možné směry přenosu. **Simplex** - umožňující přenos pouze jedním směrem, **poloduplex** - umožňující přenos oběma směry (ne však současně) a konečně **duplex** - umožňující obousměrný současný přenos.

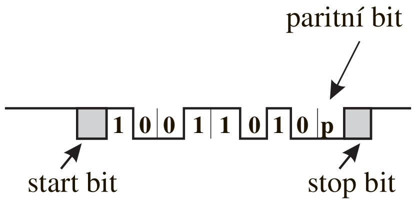
4.2 Paralelní a sériový přenos

Při **paralelním přenosu** (*parallel transmission*) jsou data přenášena po více bitech najednou, většinou po celých bytech. Pro realizaci paralelního přenosu je zapotřebí mít příslušný počet souběžných (paralelních) vodičů, což je únosné jen na krátké vzdálenosti (do 20 metrů). Klasickým místem pro použití paralelního přenosu je komunikace mezi počítačem a tiskárnou (LPT port) nebo na krátkodobé propojení dvou počítačů. Prakticky se paralelní přenos nepoužívá pro propojení sítí.

Při **sériovém přenosu** (*serial transmission*) jsou data přenášena postupně bit po bitu, nejnižší bit jako první. Sériové spojení je, na rozdíl od paralelního, vhodné a používané pro komunikaci v počítačových sítích. Nejmenší jednotkou přenosu je znak, který reprezentuje 7 nebo 8 bitů.

4.3 Asynchronní a synchronní sériový přenos

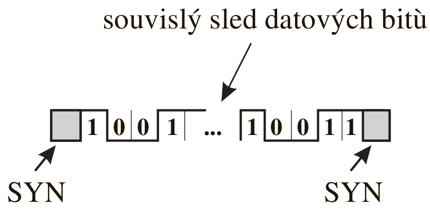
Asynchronní sériový přenos (*serial asynchronous transmission*) je charakteristický obecně různými časovými pauzami mezi zasláním jednotlivých znaků (obrázek 13). Vzhledem k tomu, že adresát musí



Obrázek 13: Asynchronní přenos dat

vědět, kdy je vysílán nový znak, řeší se toto oznámení pomocí **start bitu**. Tento bit je u adresáta významný i pro nastavení dalších informací. Za start bitem následují datové bity, které mohou být následovány paritním bitem (vysvětlíme později) a **stop bitem**, který slouží pouze jako oddělovač dvou znaků.

Při **synchronním sériovém přenosu** (*serial asynchronous transmission*) jsou datové bity zasílány bez prodlení za sebou v celých blocích. Není tedy potřeba zasílat start nebo stop bity. Zasílá se pouze znak začátku bloku, který zajistí časovou synchronizaci, následovaný datovými bity a znakem konec bloku (obrázek 14). Porovnáním synchronního a asynchronního přenosu dospějeme k názoru, že synchronní



Obrázek 14: Synchronní přenos dat

přenos je rychlejší (žádné start a stop bity), ale technicky náročnější.

4.4 Vysvětlení některých pojmů

Přenos v základním pásmu (*baseband transmission*)]
Jde o nejjednodušší typ přenosu. Potřebujeme-li přenášet dvojková data (diskrétní signál) po signálových vodičích, můžeme obě možné hodnoty (0 a 1) reprezentovat pomocí úrovní napětí na vodiči. Například jednou nulovou a jednou nenulovou úrovní nebo jednou kladnou a jednou zápornou úrovní. Používají se i složitější způsoby vyjádření logických hodnot pomocí úrovní napětí. Například tzv. kód Manchester II, který se používá u lokálních sítí Ethernet a zajišťuje určitý minimální počet změn přenášeného signálu i tehdy, má-li být přenášena posloupnost bitů stejné hodnoty. Problém tohoto způsobu přenosu je v tom, že mnohé přenosové cesty (běžné telefonní okruhy) jsou vzhledem ke svým fyzikálním vlastnostem pro přenos v základním pásmu prakticky nepoužitelné. Jiná média (koaxiální kabel) sice přenos v základním pásmu umožňují, ale nevyužívá se přitom s maximální možnou efektivitou.

Širokopásmový přenos
Při tomto přenosu je přenášen takový signál, který se daným přenosovým médiem šíří nejlépe (s nejmenšími ztrátami). Typicky jde o pravidelně se měnící signál průběhu tvaru sinus, tzv. **harmonický signál**. Užitečná informace se potom přenáší prostřednictvím změn v průběhu tohoto signálu. Vzhledem k tomu, že je tento signál jakýmsi nosičem informace, nazývá se také **nosný signál** nebo nosná (*carrier*). Jedná se vždy o analogový signál. Užitečná informace se na něj „nanáší" postupem označovaným jako **modulace**. Existuje několik druhů modulace:

Amplitudová modulace AM (*amplitude modulation*) Jednotlivé logické hodnoty jsou vytvářeny různými hodnotami amplitudy (rozkmitu) harmonického signálu.

Frekvenční modulace FM (*frequency modulation*) Jednotlivé logické hodnoty jsou vytvářeny různými frekvencemi (kmitočty) harmonického signálu.

Fázová modulace PM (*phase modulation*) Jednotlivé logické hodnoty jsou vyjádřeny různou fází (posunutím) harmonického signálu.

Pokud modulací vznikají jen dva navzájem rozlišitelné stavy, hovoříme o modulaci dvoustavové. Kombinací různých druhů modulace lze dosáhnout více rozlišitelných stavů, které potom přenáší vícebitovou informaci.

Multiplex
Pro lepší využití přenosového média při širokopásmovém přenosu se používá tzv. *multiplexing*. Jde o techniku, která umožňuje rozdělit jeden přenosový kanál s velkou šířkou pásma na několik (užších) logických subkanálů. Ty se jeví jako samostatné nezávislé kanály. Existují dva základní způsoby dělení jednoho přenosového kanálu na několik subkanálů:

Frekvenční multiplex FDM (*frequency division multiplexing)* Při FDM je jednotlivým logickým subkanálům přidělena taková část celkové šířky kanálu, kterou potřebují. Přenášený signál je pomocí zařízení nazývaného multiplexor „posunut" do části pásma přiděleného danému subkanálu. Na druhé straně spoje je opět nutné signál „vrátit zpět".

Př.: Šíření rádia. Přenosové médium je vzduch, každá stanice dostane přiděleny určité hranice frekvencí. Vysílač se pak vyhledá tak, že přijímač nastavíme zhruba do středu vysílacího pásma kanálu.

Časový multiplex TDM (*time division multiplexing*) Zde je vlastní přenosový kanál s celou svou šířkou pásma v krátkých časových intervalech pravidelně přidělován jednotlivým subkanálům.

Př.: Přenos číselné informace. Čísla jsou rozdělena na bity a v jednom časovém okamžiku jsou vyslány všechny první bity čísel, v dalším druhé bity, atd.

Modulační rychlost (*modulation speed*)
Vyjadřuje počet změn nosného signálu za jednotku času a měří se v baudech (Bd). Modulační rychlost ještě nic neříká o tom, jaké množství informace nosný signál přenáší.

Přenosová rychlost (*transmission speed*)
Tento parametr udává objem infomace přenesený za jednotku času. Vyjadřuje se v bitech za sekundu

(kbit/s ; *1024 bits per second, bps*). V případě použití vícestavové modulace je přenosová rychlost vyšší než modulační.

Potřebujeme-li přenášet data, můžeme to udělat tak, že je nahrajeme na diskety nebo CD a přeneseme. Tady je lepší hovořit spíše o přepravě dat než přenosu. Vazba mezi zdrojem a cílem je tzv. *off-line*. Existuje-li mezi zdrojem a cílem možnost přenosu elektronickou cestou, pak je na místě mluvit o přenosu dat a používat přívlastek *on-line*. Nutnou podmínkou přenosu elektronickou cestou je vhodný kanál.

Přenosová cesta
Přenosová cesta zajišťuje vlastní přenos signálů. Může jít buď o drátové spoje (koaxialní kabel, optické vlákno) nebo o bezdrátové spoje (krátkovlné, radiové, družicové). Základní charakteristikou přenosové cesty je šířka pásma. Ta pak může být použita pro jeden jediný kanál nebo může být rozdělena pomocí multiplexu na několik subkanálů. Je možný i opačný případ, kdy více přenosových cest (pomocí vhodných technických prostředků) vytvoří kanál s větší šířkou pásma.

Kanál (*channel*), okruh (*circuit*)
Kanál je souhrn prostředků, které vytvářejí telekomunikační spojení mezi dvěma místy. Kanál je charakterizován určitou šířkou pásma, přenosovou rychlostí, úrovní šumu, chybovostí atd. Kanál je obvykle chápán jako jednosměrný, v případě obousměrného kanálu se hovoří spíše o okruhu (dvojice jednosměrných kanálů). Pro vytvoření kanálu (okruhu) musí existovat vhodná přenosová cesta. Pokud vede mezi spojovanými místy vhodná přenosová cesta, lze kanál vytvořit přímo pomocí této jediné přenosové cesty. V opačné případě je nutné kanál vytvořit z více přenosových cest, které na sebe navazují.

Přenosová cesta je tedy pojem, který se týká především struktury existujících spojových prostředků, zatímco pojem kanál (okruh) v sobě zahrnuje jejich organizaci, tj. způsob jejich využívání. Samotné kanály a okruhy však ještě nejsou schopny poskytovat přenosové služby. Například telefonní okruh je sice schopen přenášet telefonní signály, ale k jejich praktickému využití potřebujeme ještě telefonní přístroj, který převádí lidský hlas na telefonní signál a naopak.

Soukromý okruh (*private circuit*)
Uživatel si zřídí potřebnou přenosovou cestu sám (např. položí potřebné vedení mezi dvěma blízkými uzlovými počítači) a na ní si zřídí okruh.

Veřejný okruh (*public circuit*)
V každém státě existuje spojová organizace, která zřizuje veřejné okruhy, které za patřičný poplatek a při dodržení příslušných technických podmínek smí používat každý. Příkladem může být veřejná telefonní síť.

Pronajatý okruh (*leased circuit*)
Provozovatelé počítačových sítí, kteří potřebují mít vhodný okruh trvale k dispozici a jen pro sebe, si jej mohou pronajmout od spojové organizace. Okruh zůstává majetkem spojové organizace, ale výhradní právo na jeho používání má ten, kdo si okruh pronajal.

4.5 Zabezpečení proti chybám

Na fyzické úrovni je možné, že nastane chyba při přenosu a ta má za následek přijetí jiné hodnoty bitu, než byla zaslána. Existuje několik metod, které buď pouze detekují výskyt takové chyby (detekční bezpečnostní kódy), nebo dokonce opraví přijatou zprávu do správného tvaru (samoopravné bezpečnostní kódy). Podstatou bezpečnostního kódu je, že se přenáší větší množství dat, než obsahuje zpráva (klasicky se ke každému slovu nebo skupině slov přidá několik kontrolních bitů). Pomocí těchto redundantně přenesených dat se pak dá kontrolovat nebo opravit přijatá zpráva.

4.5.1 Parita - detekční kód

Nejjednodušší a současně nejméně účinný způsob zabezpečení zprávy, který umožňuje pouze detekci výskytu chyby, je doplnění datových bitů jedním dalším bitem tak, aby celkový počet jedniček ve zprávě byl při odesílání lichý (lichá parita) nebo sudý (sudá parita). Paritní bit lze zasílat například za každým znakem.

Příjemce musí vědět, s jakou paritou byly zprávy odesílány. Jestliže počet přijatých jedničkových bitů nesouhlasí s očekávanou paritou, došlo jistě při přenosu k chybě aspoň jednoho bitu (obecně lichého počtu bitů). Naopak má-li přijatý znak očekávanou paritu, nezaručuje to bezchybnost zprávy, protože nelze odhalit chybu v sudém počtu bitů použitím jednoho paritního bitu.

Zabezpečení pomocí jednoho paritního bitu je tedy vhodné používat jen tam, kde je výskyt chyb v jednotlivých bitech málo pravděpodobný a četnost výskytu chyb ve více bitech současně zanedbatelná.

4.5.2 Hammingův samoopravný kód

Zástupcem samoopravného kódu je například rozšířený Hammingův kód, který přidává ke každému osmibitovému slovu pět bitů kontrolních.

Vzhledem k tomu, že data se vždy přenášejí po větších blocích, při praktickém použití bezpečnostních kódů se zabezpečují vždy bloky dat (ne každé slovo zvlášť). V případě detekované chyby ve zprávě se totiž stejně bude přenášet celý blok, ve kterém je zpráva obsažena.

4.5.3 Kontrolní součet

Jednoduchým detekčním mechanismem na bloku dat je použití kontrolního součtu. Jeho podstatou je součet všech znaků zprávy v binární reprezentaci v aritmetice modulo 2^k, kde $k \in \{8, 16, \ldots\}$. Výsledkem součtu je právě kontrolní součet, který je implementován na k bytech.

4.5.4 Cyklický kód (CRC)

Na základě jednotlivých bitů znaků zprávy se vypočítá hodnota podobně jako u kontrolního součtu. Tato hodnota se připojí na konec zprávy. Jestliže příjemce vypočítá z přijaté zprávy stejnou hodnotu, jakou přečte za zprávou, lze s vysokou pravděpodobností uznat přenos za správný.

5 Lokální počítačové sítě

Pracovní skupina institutu IEEE používá následující definici: „Lokální počítačová síť se odlišuje od ostatních typů počítačových sítí tím, že je komunikace omezena na středně velké geografické území, jako je jedna budova úřadu, jeden závod nebo jeden univerzitní areál a je založena na fyzických přenosových kanálech se střední nebo vysokou přenosovou rychlostí a trvale nízkým výskytem chyb."

Konkrétně se jedná o LAN, jestliže shledáme, že

- síť pokrývá území území v rozsahu do několika čtverečních kilometrů
- používá drátová přenosová média, není třeba používat veřejný telekomunikační systém
- přenosové rychlosti 1–100 Mbit/s (tento parametr s časem zastará)
- komunikační podsíť vlastní a spravuje uživatel LAN

5.1 Vlastnosti, využití

Lokální síť by měla mít tyto vlastnosti:

- levné přenosové prostředky při relativně vysokých přenosových rychlostech
- architekturu otevřených systémů
- možnost rekonfigurace
- možnost připojení různých počítačových zařízení do jednoho systému
- možnost snadné návaznosti na systém dálkového přenosu a veřejných sítí

Lokální počítačová síť zajišťuje tyto služby :

- sdílení nákladných periferií (laserové tiskárny, velkokapacitní diskové systémy, mechaniky CD-ROM apod.)
- sdílení společných dat a aplikací (aktuální data, úspora diskového prostoru, snadná záloha, přechod na vyšší verzi produktu apod.)
- jednoduchá vzájemná komunikace uživatelů (výměna zpráv, lokální pošta)
- ochrana dat (náhodné zničení, mezi uživateli navzájem)

Lokální počítačové sítě můžeme v zásadě rozdělit do dvou základních skupin: sítě **serverového typu** a sítě **peer-to-peer**. Oba systémy se od sebe odlišují počtem počítačů zapojených v síti, poskytovanými síťovými službami, obtížností správy a údržby sítě.

5.1.1 Sítě typu peer-to-peer

Sítě *peer-to-peer* (volně přeloženo rovný s rovným) se vyznačují tím, že všechny do sítě zapojené počítače jsou si alespoň v principu rovny. Síťový operační systém tvoří pouze jakousi nádstavbu běžného operačního systému. Ve formě rezidentních programů umožňuje poskytovat základní síťové služby. Všechny počítače mohou nabízet své služby (disky, tiskárny atd.) všem ostatním do sítě zapojeným počítačům a naopak každý počítač může využívat kterékoli nabízené služby (samozřejmě v závislosti na přidělených právech). Počet počítačů v takovéto síti je omezen pouze na několik kusů. Instalace a správa takovéto sítě bývá obvykle nenáročná a přístupná uživateli, který má dobré znalosti práce s PC. Finanční náročnost zavedení sítě typu peer-to-peer není velká, tyto sítě jsou vhodné pro jednoduché aplikace bez nutnosti (a možnosti) vysokých bezpečnostních opatření a bez požadovaného vysokého výkonu. Typickými představiteli sítí peer-to-peer jsou sítě LANtastic nebo mnohem rozšířenější Windows for Workgroups, Windows 95 a typickým místem nasazení jsou menší firmy. Současná doba také přeje těmto druhům sítí po stránce cenové.

5.1.2 Sítě serverového typu

Tento druh lokální počítačové sítě je obvykle používán pro větší počet počítačů. Služby serverových sítí jsou komplexnější a poskytují lepší možnosti správy sítě. Samozřejmě správa a údržba této sítě je mnohem náročnější a vyžaduje speciální znalosti a práci speciálního pracovníka. Počítače v síti serverového typu můžeme rozdělit do dvou následujících skupin: **obslužné stanice** (*servers*), které poskytují své prostředky a služby ostatním počítačům v síti, a **pracovní stanice** (*workstations*), dále jen stanice, které žádné služby ostatním počítačům nenabízejí, ale využívají služeb serverů.

Servery lze rozdělit do skupin podle typu poskytovaných služeb:

Souborový server (*file server*)
: Pojem server se většinou používá právě ve významu *file server*, což je dnes nejpoužívanější druh serveru. Souborový server je počítač vybavený velkokapacitními disky, na kterých jsou uloženy soubory. Jeho základní funkcí je umožnit uživatelům řízený sdílený přístup ke společným datům a programům. Navíc poskytuje ochranu před neoprávněným čtením i zápisem dat na základě evidence databáze uživatelů a hesel.

Diskový server (*disk server*)
: Jde o předchůdce souborového serveru, který se dnes již prakticky nepoužívá. Jeho základní funkcí bylo zefektivnit přístup rozdělením velkého disku na části, ke kterým mají jednotlivý uživatelé přístup. Sdílení společných souborů může poskytnout jen ve velmi omezené míře.

Databázový server (*database server*)
: Databázový server je speciální typ souborového serveru. Databázová aplikace provozovaná na stanici nevyžaduje na serveru zasílání všech dat, ale pouze formuluje požadavek (např. ve formě SQL dotazu) na operaci s daty. Server odešle zpět na stanici pouze data vyhovující zadaným podmínkám. Veškeré akce pro práci s databází jsou prováděny přímo na databázovém serveru, což výrazně zvyšuje výkon systému a snižuje zátěž komunikačních linek.

Tiskový server (*print server*)
: Tento server umožňuje uživatelům sítě tisk na sdílené tiskárně (popř. tiskárnách) připojené k tomuto serveru. Není nutné každé pracoviště vybavovat tiskárnou, stačí ji připojit k tomuto serveru a tak zajistit zpracovaní požadavků na tisk v celé lokální síti. Dnes se již vyrábějí tiskárny, které samy o sobě mohou fungovat jako samostatná tisková fronta, jsou připojeny vlastním síťovým adaptérem do sítě a fungují tak vlastně podobně jako jednoduchý tiskový server s jednou tiskárnou.

Komunikační server (*communication server*)
: Tento typ serveru poskytuje ostatním uživatelům sítě komunikační služby. Většinou jde o prostředek umožňující vstup do sítě přes telefonní linky, kdy provádí emulaci zvoleného terminálu. Nebo naopak umožňuje napojení lokální sítě na veřejnou datovou síť a tím možnost spojení s rozlehlými počítačovými sítěmi.

Existují samozřejmě ještě další druhy serverů jako např.: servery pro obsluhu faxu, elektronické pošty, účtovací služby atd. Některé servery plní více funkcí, takže např. *file server* je zároveň i *print serverem* a nebo naopak v jedné síti se může vyskytovat více serverů stejného typu.

Serverové počítače mohou pracovat jako **dedikované** nebo **nededikované**. Dedikované serverové počítače slouží pouze k účelu serveru a nedají se využít jako pracovní stanice - takto funguje například souborový nebo tiskový server systému Novell Netware. Naproti tomu nededikované servery mohou být použity pro běžnou práci, neboť se vzhledem k jejich výkonu počítá, že neovlivní znatelně výkon serveru. Nededikovaně pracují zejména servery UNIX a Windows NT. U nededikovaných serverů musíme vždy počítat s nevyzpytatelností uživatele, který přímým přístupem k počítači může způsobit nečekané problémy (vypnutí počítače, ...).

V sítích osobních počítačů se používají dva typy stanic:

Standardní osobní počítače
Tyto stanice pracují často ve dvou režimech, buď pouze jako normální lokální počítače nebo navíc využívají služeb poskytovaných počítačovou sítí. Lokální disky jsou používány k uchování privátních dat nebo dat lokálních programů, často také ke swapování systémů typu UNIX nebo Windows.

Bezdiskové stanice (*diskless stations*)
Tyto stroje jsou, na rozdíl od standardních osobních počítačů, speciálně určeny pro práci v síti. Nemají žádný vlastní pevný disk ani disketovou mechaniku (nemohou tedy pracovat jako lokální počítač) a server využívají už k zavedení operačního systému (*remote boot*). Přináší to nejen menší náklady na pořízení stanice, ale i vyšší bezpečnost informací uložených na serveru a znemožnění zavlečení virů do síťového prostředí.

Bezdiskové stanice mohou výrazně negativně ovlivnit zátěž komunikační podsítě. Často právě takto ušetřené investice způsobují zahlcení lokálních sítí Ethernet.

5.2 Přenosová média

Možnosti využití jednotlivých druhů kabelů při propojování počítačů v síti jsou určeny především jejich kvalitativními parametry. K nejdůležitějším parametrům patří:

Přenosová rychlost, které lze na daném kabelu dosáhnout. U drátových přenosových cest se přenosová rychlost pohybuje řádově mezi jednotkami až desítkami Mbit/s, u optických kabelů jde o stovky Mbit/s.

Útlum představuje zeslabení přenášeného signálu. Je způsoben odporem, který kabel klade přenášenému signálu. Útlum roste se zvyšujícími se frekvencemi a s klesajícím průměrem kabelu. Celková hodnota útlumu je přímo úměrná délce kabelu a je jedním z rozhodujících faktorů, které určují maximální použitelnou délku segmentu kabelu.

Odolnost vůči rušení. V okolí kabelu může docházet k různým jevům (např. provoz elektrospotřebičů), které mají nepříznivý vliv na přenášený signál. Každý druh kabelu vykazuje obecně jinou odolnost vůči rušení. U souběžně vedoucích vodičů se může vyskytnout jiný typ rušení, tzv. **cross-talk**, tedy přeslech. V tomto případě signál přenášený jedním kabelem interferuje se signálem v druhém kabelu.

5.2.1 Kroucená dvojlinka (*twisted pair*)

Kabel ze dvou kroucených měděných vodičů je klasickým spojovacím prostředkem, který dnes představuje nejlevnější propojení počítačů. Vodiče v kroucené provedení jsou odolnější proti vnějším rušivým vlivům, zejména účinkům rušivých elektromagnetických polí. K přenosu se totiž využívá rozdíl signálů na jednotlivých vodičích a dojde-li vlivem rušení ke změně signálu, projeví se změna na obou vodičích a v rozdílu nebude patrná. Přesto je kroucená dvojlinka náchylná vůči některým rušení a používá se **stíněná** kroucená dvojlinka.

Použití kroucené dvojlinky je běžné nejen pro rozvody telefonů, ale formou tzv. strukturované kabeláže velmi často také pro přípojky počítačů do počítačové sítě. Je použitelná například v sítích Ethernet, kde dosahuje přenosy s rychlostí 10Mbit/s, tedy srovnatelné s propojením koaxiálním kabelem. V praxi je používán také standard **Fast Ethernet**, který na kroucených dvojlinkách dosahuje přenosových rychlostí 100Mbit/s.

5.2.2 Koaxiální kabel (*coaxial cable*)

Dnes zřejmě nejpopulárnější drátová přenosová cesta je tzv. koaxiální kabel. Jedná se o kabel kruhového průřezu, kde je ve středu datový vodič oddělený izolací od vnějšího vodivého opletení (tzv. stínění). Vnější ochranná vrstva kabelu pak bývá vyráběna v různých provedeních, a to od obyčejného PVC, přes teflonovou izolaci až ke speciálním mrazuvzdorným umělým hmotám. Podle vnější izolace je pak možno používat kabel buď pro vnitřní nebo venkovní rozvody. Koaxiální kabely se používají např. v lokálních sítích Ethernet (viz dále). Nejznámější jsou dva typy kabelu označované jako **tlustý Ethernet** (*Thick Ethernet*) a **tenký Ethernet** (*Thin Ethernet*). Výhodou tlustého Ethernetu je především značný dosah a velká odolnost proti rušení, nevýhodou pak vysoká cena a malá ohebnost. U tenkého Ethernetu je tomu naopak - menší dosah a odolnost vůči rušení, ale nízká cena a velká ohebnost. Všechny druhy Ethernet kabelů se používají pro standardní přenosovou rychlost 10Mbit/s.

5.2.3 Optické kabely (*optical cable*)

Oba výše uvedené typy slouží k přenosu binárních dat ve formě elektrického signálu. Optické kabely přenášejí binární informace ve formě světelného toku. Ve snaze zvyšovat přenosovou rychlost byly hledány takové způsoby přenosu, které mají co možná největší šířku pásma. Světlo se k tomuto účelu jeví svojí vysokou frekvencí (v řádu 10^8 MHz) jako velmi vhodné. Výhodou optického přenosu informací je tedy vysoká přenosová rychlost a odolnost proti rušivým elektromagnetickým polím. Jedná se o rychlosti dosahující až 1 Gbit/s.

Princip přenosu optickým vláknem je založen na reprezentaci číslicových dat světelnými impulsy. K samotnému přenosu se používá zdroje světla (LED **dioda** nebo laserová dioda), které na základě přiváděného proudu (binárních dat) vyzařují světelné pulsy. Na opačné straně optického vlákna pracuje **fotodioda**, která má opačný úkol.

Přenos světla zajišťuje **optické vlákno**, které je tvořeno velmi tenkým jádrem (průměr v desítkách mikrometrů) a pláštěm. Přechod mezi jádrem a pláštěm je realizován tak, aby se světelný paprsek vycházející z jádra pod jistými úhly lámal bez ztráty zpět do jádra (světelný paprsek se při dopadu na rozhraní dvou prostředí láme podle vlastností tohoto rozhraní a úhlu, pod kterým tento paprsek na rozhraní dopadá). Důsledkem opakovaných odrazů sleduje paprsek dráhu jádra optického vlákna.

Z výhod optických vláken jmenujme vysokou přenosovou rychlost (Gbit/s), necitlivost vůči elektromagnetickému rušení, bezpečnost vůči odposlechu, nízká hmotnost optických kabelů (svazek optických vláken) a velká vzdálenost bez potřeby zesilovacích prvků.
Nevýhodou je složitý způsob spojování vláken v porovnáním s jinými kabely.

V přenosech používaných k propojení počítačových sítí existuje standard **FDDI** pro lokální sítě Token Ring s přenosovou rychlostí 100Mbit/s a **DQDB** pro metropolitní sítě s přenosovou rychlostí 155Mbit/s.

Důležitým klasifikačním faktorem je také šířka přenosového pásma. Kroucené dvojlinky mají šířku pásma maximálně 100MHz, koaxiální kabel okolo 500MHz a optický kabel používaný v LAN až 2000MHz.

5.3 Technika přenosu

Při přenosu paketů po lokální počítačové síti může docházet v zásadě ke dvěma chybovým situacím. V prvním případě cílová stanice sice paket přijme, ale kontrolou jeho zabezpečovací části zjistí, že přenos proběhl nesprávně. V druhém případě cílová stanice paket nepřijme vůbec v důsledku toho, že bylo např. poškozeno adresové pole, nebo proto, že během vysílání nebyla cílová stanice připravena k příjmu (např. neměla volnou vyrovnávací paměť). Pro bezchybný přenos informací musí být vysílací strana schopna oba uvedené typy chyb detekovat a příslušný paket pak opakovat. K tomuto účelu se nejčastěji používají metody potvrzování zpráv. Existuje několik postupů, které mohou být v praxi kombinovány a modifikovány.

5.3.1 Pozitivní potvrzování

Přijímací stanice v případě, že převzala datový paket bez chyby, vyšle speciální potvrzovací paket. Na chybně přijatý paket a samozřejmě i na nepřijatý paket přijímací stanice nereaguje. Vysílací strana po zaslaní paketu musí čekat určitou dobu na příchod potvrzovacího paketu. Teprve po jeho příjmu může pokračovat vysíláním dalšího paketu. Nepřijde-li potvrzení ve vymezené době, opakuje se vysílání paketu znovu. Uvedený postup se dokáže vyrovnat se ztrátou původního paketu, ale ztráta potvrzovacího paketu vede k duplicitnímu vyslání původního paketu. Pozitivní potvrzování je základem v praxi používaných metod, ve své čisté podobě má však nevýhodu ve velkém nárůstu zpoždění při růstu chybovosti a relativně velkém zatížení sítě při vysílání krátkých paketů.

5.3.2 Negativní potvrzování

Základní princip této metody vychází z toho, že je vysílací strana upozorněna pouze na chybně přijatý paket vysláním speciálního paketu. Tato metoda rychle reaguje na chybně přijatý paket, ale nedokáže se vyrovnat s jeho ztrátou. Proto se obvykle kombinuje s pozitivním potvrzováním. V této kombinaci pak rychle reaguje na chybně přijatý paket nebo potvrdí správně přijatý paket, přičemž ztracené pakety se detekují uplynutím časového limitu. Ztráta potvrzení vede k duplicitě na přijímací straně.

5.3.3 Číslování paketů

Číslování paketů je metoda, která umožňuje přijímací straně detekovat duplicitu. Aby přijímací strana poznala, že se jedná o již dříve přijatý paket, jsou všechny vysílané pakety číslovány vzestupně. Z praktických důvodů se používají např. cyklicky čísla v rozsahu 0 až 127. Lze tak detekovat nejen opakovanou zprávu, ale i ztrátu paketu, a to za předpokladu, že propojovací síť zaručuje dodržení posloupnosti vyslaných paketů. Kromě vysílaných paketů mohou být číslovány i pakety potvrzovací.

5.3.4 Skupinové potvrzování

Při potvrzování každého paketu samostatně mohou značně narůstat časové ztráty, zvláště pak při větším zpoždění v přenosové cestě. Výkonnější je pak potvrzovat celou skupinu paketů najednou. Vysílací stanice očekává potvrzení až po vyslání N paketů. Přijímací strana pak potvrzuje pořadovým číslem posledního správně přijatého paketu. Na vysílací straně je nutno zajistit vyrovnávací paměť o kapacitě N paketů.

5.3.5 Nesamostané potvrzování

Další zvýšení využitelnosti přenosové cesty lze dosáhnout tím, že v každém vysílaném paketu vyhradíme zvláštní pole na informaci o čísle potvrzovaného paketu. Pokud pak dojde k obousměrné výměně zpráv mezi stanicemi (což je běžný případ), může být potvrzení přijatého paketu součástí následně vyslané vlastní zprávy. Nemá-li přijímací stanice připravena vlastní data k vysílání, použije se klasický potvrzovací paket.

Většina protokolů komunikace po lokální síti používá některou z forem skupinového nesamostatného potvrzování.

5.4 Přístupové metody

Pokud je pro vytvoření počítačové sítě použito sdílené přenosové médium, je zapotřebí mít metodu, která dovolí předávat data mezi libovolnými stanicemi, aniž by jejich spojení bylo narušeno vysíláním jiné stanice. Jednotlivé přístupové metody používají principy přidělování média uvedené v následující kapitole.

5.4.1 Principy přístupových metod

Statické přidělování

Kapacita přenosového spoje je pevně rozdělena do částí přidělených jednotlivým stanicím. Rozdělení může být buď frekvenční nebo časové. Frekvenční multiplex rozdělí celkovou kmitočtovou šíři kanálu do několika částí, které jsou potom přiděleny příslušným stanicím. Tento způsob se používá pro využití dálkových kabelů nebo pro současný přenos různých typů informace (data, hlas, obraz). Časový multiplex rozděluje sdílenou kapacitu na posloupnost dílčích časových úseků, z nichž každý je přidělen jedné stanici. Jeden úsek musí být věnován synchronizační zprávě vysílané řídící stanicí, nutné k jednoznačné identifikaci jednotlivých úseků příslušnými stanicemi. Časový multiplex mohou využívat družicové sítě.

Centrální přidělování

V síti existuje jedna stanice, která může být pověřena úkolem přidělovat kapacitu kanálu podřízeným stanicím, které ji skutečně potřebují. Výhodou je celkově lepší využití přenosové cesty. Existují dvě varianty: přidělování na žádost a přidělování na výzvu.

U **přidělování na žádost** musí mít každá stanice pro sebe vyhrazenou malou část přenosové kapacity kanálu. V ní může kdykoli požádat řídící uzel o přidělení volného přenosového kanálu. Po potvrzení řídícím uzlem pak zahájí přenos dat a uvolnění kanálu oznámí opět centrální stanici.

Přidělování na výzvu (*pooling*) je plně řízeno centrální stanicí, která se periodicky dotazuje všech podřízených stanic, zda mají připravena data pro vysílání. Pokud ano, dotazovaná stanice data ihned odešle. Nemá-li data připravena, odpoví pouze potvrzovacím paketem nebo neodpoví vůbec. Centrální stanice potom pokračuje dotazem na další stanici.

Náhodný přístup

Metody, které využívají náhodný přístup, nepotřebují centrální stanici. Okamžik svého přístupu na sdílený kanál si všechny komunikující stanice určují samostatně. Nedokáže se však vyloučit situace, kdy se dvě (obecně několik) stanic současně pokusí o vysílání. Používané přístupové metody se snaží počet těchto kolizí minimalizovat a tím udržet propustnost dat co největší. Základní nevýhodou této metody je, že nelze zaručit časový interval, ve kterém dojde k přenosu zprávy.

Distribuované přidělování

Distribuované přidělování zaručuje bezkonfliktní přidělování sdíleného kanálu jednotlivým stanicím tak, že lze zaručit hodnotu maximálního zpoždění paketu, přičemž není potřebná existence centrální stanice. Algoritmus přidělování kanálu běží na všech zúčastněných stanicích. Uvedeme dvě skupiny distribuovaného přidělování.

Synchronní metody předpokládají časové rozdělení kanálu do krátkých úseků - minislotů. Každá stanice má svou logickou adresu. Když stanice ukončí vysílání, následuje minislot, ve kterém může zahájit vysílání stanice s následující adresou. Pokud tak neučiní, pak v dalším minislotu přechází právo na následující stanici. Nevýhodou této metody je větší zpoždění při malém zatížení sítě a hlavně nutnost synchronizace jednotlivých stanic. V **logickém kruhu** jsou opět všechny stanice označeny adresou, přičemž tyto adresy tvoří cyklickou posloupnost. Každá stanice zná kromě své adresy také adresu svého bezprostředního následovníka. Stanice, která odvysílala svá data předá vysílací právo následující stanici. Ta, pokud má data pro vysílání, může je vyslat. Pokud data nemá, ihned předá vysílací právo svému následníkovi.

Distribuované přidělování lze realizovat dvěma způsoby:

neprioritní – uzel má právo vyslat pouze jednu zprávu a po ní hned zprávu představující vysílací právo. Pokud chce vysílat více zpráv, musí počkat „jedno kolo", než se mu vysílací právo vrátí .

Prioritní – některý uzel může mít přidělenu vyšší prioritu spočívající v možnosti, že po obdržení vysílacího práva může vyslat několik zpráv najednou (popř. jednu delší zprávu).

5.4.2 Token - Ring

Přístupová metoda *Token-Ring* se používá na kruhové topologii (jednotlivé uzly jsou uspořádány ve fyzickém kruhu a jde o neprioritní distribuované přidělování). Tato metoda vychází z vlastnosti, že sítí je od jednoho uzlu ke druhému předávána speciální zpráva (*token* - „pešek"), která představuje vysílací právo. Uzel, který od svého souseda (předchůdce) dostal vysílací právo, může provést jednu z následujících akcí:

- pokud chce v síti poslat zprávu, vyšle ji. Poté vyšle zprávu představující vysílací právo na následující uzel;

- pokud nechce vysílat, pošle pouze zprávu, která představuje vysílací právo, následujícímu uzlu.

Výhody:

- jedná se o deterministickou přístupovou metodu, tzn. každý uzel v síti má zajištěn během určité maximální doby přístup k médiu
- metoda umožňuje potvrzení přijetí zprávy adresátem (vysílací uzel vyčká, až se vrátí jím vyslaná zpráva s případným potvrzením adresátem, a teprve pak předá vysílací právo)

Nevýhody:

- metoda (a její implementace) je složitější v tom, že uzel musí zkoumat obsah zprávy (zda je to zpráva či vysílací právo) a určit, zda chce vysílat zprávu či nikoliv
- přenosy vysílacího práva snižují přenosový výkon sítě
- při přenosech vysílacího práva může dojít k chybovým stavům (ztráta práva, jeho duplikace)
- uzly mohou být příčinou poruchy sítě

5.4.3 Token - Bus

Token-Bus je obdobou metody Token-Ring s tím, že se používá na sběrnicové topologii. Neexistuje tedy fyzický kruh, vytváří se pouze logický uzavřený kruh pomocí adres, které jsou přidělovány jednotlivým uzlům sítě dynamicky. Právo se nepředává fyzickému sousedovi aktivního uzlu, ale podle logických adres. Uspořádání logického kruhu je nutno modifikovat při přihlášení a odhlášení uzlu.

Poznámka: Odhlašující se uzel pošle svému předchůdci adresu svého následníka a zprávu, že končí. Ostatní vlastnosti jsou analogií výše popsané metody.

5.4.4 CSMA/CD

Metoda CSMA (*Carrier Sense Multiple Access*) je neprioritní metoda s náhodným přístupem, určená pro sběrnicovou topologii. Vychází ze situace, kdy jednotlivé uzly soupeří o získání přenosového kanálu. Uzel, který chce zahájit vysílání, „naslouchá" (*CS*) provozu na přenosovém kanále a zahájí vysílání pouze tehdy, když „neuslyší" na kanále přenos. Při tomto přístupu ovšem může zjistit klid na lince několik uzlů najednou a zahájit vysílání ve stejný okamžik (*MA*). Dochází ke kolizi, kterou je třeba řešit buď přímo na úrovni CSMA (tedy linková vrstva) nebo pomocí vyšších vrstev RM OSI.

Přístupová metoda CSMA/CD (*CSMA/Collision Detection*) je zdokonalením výše uvedené přístupové metody tak, že detekci kolize provádí přímo vysílací uzel. Ten po zjištění klidového stavu kanálu začne vysílat data a současně monitoruje stav přenosového kanálu, aby zjistil případnou kolizi. Pokud ji zjistí, ukončí vysílání. Opakované vysílání uvažované zprávy jednotlivé uzly zahájí po náhodně zvoleném časovém okamžiku, aby se zamezilo opětovnému vzniku kolize.

Výhody:

- algoritmus metody je poměrně jednoduchý, zcela distribuovaný a nemá žádný řídící prvek
- metoda je dobře zvládnuta jak prakticky tak teoreticky

Nevýhody:

- metoda není deterministická, tj. nezaručuje danému uzlu přístup na médium za určitý časový úsek (není vhodná pro řízení v reálném čase)
- neumožňuje vysílacímu uzlu zjistit, zda adresát zprávu obdržel (zajištění dodání zprávy je záležitostí vyšších vrstev)
- při vysoké zátěži (a s tím souvisejícím zvýšením četností kolizí) prudce klesá přenosový výkon sítě

CSMA/CD je v současnosti přes své nevýhody zřejmě nejrozšířenější přístupová metoda.

5.5 Typy sítí

V současnosti mají vedoucí postavení na trhu tři sítě: Ethernet, Arcnet a Token Ring, které se opírají o výše vysvětlené přístupové metody. Uvedené sítě mají vztah (byť různého stupně) ke třem základním normám IEEE 802.3, 802.4, 802.5, což mělo samozřejmě velký vliv na jejich prosazení v praxi.

5.5.1 Ethernet

Ethernet je jméno sítě, která vznikla na počátku 70. let v laboratořích firmy *Xerox*. Verze, která se používá v současné době, byla standardizována díky společnému úsilí amerických firem *Digital Equipment Corporation*, *Intel* a *Xerox*. Tato verze byla uvedena na trh pod zkratkou **DIX Ethernet** v roce 1980. Během roku 1981 došlo k úpravám, které zohlednily požadavky normy IEEE 802.3.

Síť Ethernet používá jako základní sběrnicovou topologii, kterou lze přetvářet na hvězdicovou, případně stromovou. Základní přístupová metoda je *CSMA/CD* a jako přenosové médium se používá koaxiální popř. optický kabel. Nejnovější snahy jsou o zavedení bezdrátového spoje v oblasti GHz vln.

V závislosti na použitém přenosovém médiu se mění i charakteristiky a struktura celé sítě.

Thick Ethernet:
Při zapojení pomocí tlustého Ethernetu je maximální délka segmentu 500m. Jednotlivé stanice jsou připojovány pomocí speciální přípojky (*transceiveru*). *Transceiver* je připojován ke stanici přes *transceiver cable*, který může mít maximální délku 50m. Existují také multitransceivery, které umožňují připojení více uzlů najednou.

Thin Ethernet:
Maximální délka segmentu je pouze 185m (při použití speciálních karet 3Com, SMC až 300m). K připojení stanice se používají tzv. BNC konektory.

Oba typy kabelů lze kombinovat pomocí přechodových konektorů. Jednotlivé segmenty lze propojovat pomocí *repeaterů* (maximálně 2 na trase). Obvyklé je použití páteřní sítě z tlustého Ethernetu a jednotlivých větví z tenkého Ethernetu.

Kromě názvu Ethernet se pro síť tohoto typu používá označení, které přesněji vystihuje jednotlivé charakteristiky sítě. Toto označení se skládá ze tří částí: přenosová rychlost, typ přenosu, přenosové médium.

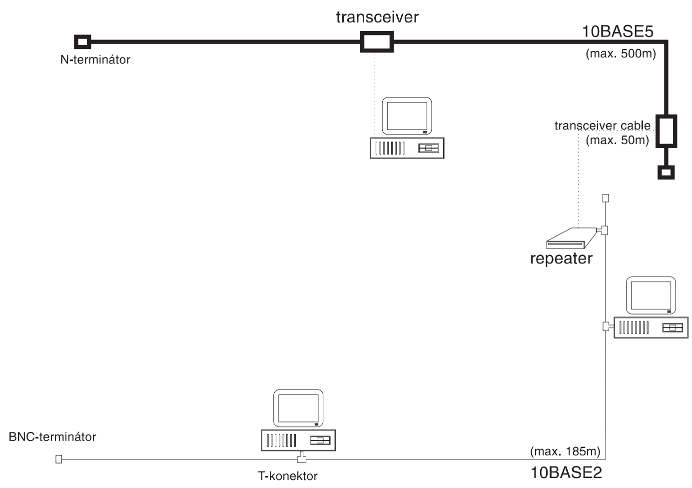
Označení	Přenosová rychlost	Typ přenosu	Přenosové médium
10BASE5	10 MBit/s	v základním pásmu	tlustý Ethernet
10BASE2	10 MBit/s	v základním pásmu	tenký Ethernet
10BASET	10 MBit/s	v základním pásmu	kroucená dvojlinka
xxBROAD		širokopásmový přenos	

Adaptéry pro sítě Ethernet jsou vyráběny řadou výrobců v různých modifikacích. Většinou existuje možnost připojení dvou různých médií např. tlustého a tenkého koaxiálního kabelu nebo tlustého koaxiálního kabelu a optického vlákna, nejnověji pak všech tří typů současně. Adresa adaptéru je vytvořena již výrobcem a uživatel ji nenastavuje (ani nemůže).

5.5.2 Arcnet

Lokální počítačová síť typu Arcnet byla vyvinuta americkou firmou *Datapoint Corporation* a byla vůbec první komerčně dostupnou LAN uvedenou na trh již v roce 1977 (tj. v době, kdy teprve začal vznikat RM-OSI). Původně byla vyvinuta pro práci na tenkém koaxiálním kabelu, ale později umožnila zapojení i pomocí kroucené dvojlinky a optického kabelu. Instalace tohoto typu sítě je poměrně levná a vzhledem ke své dlouholeté existenci i velice rozšířena, přestože nebyla nikdy standardizována.

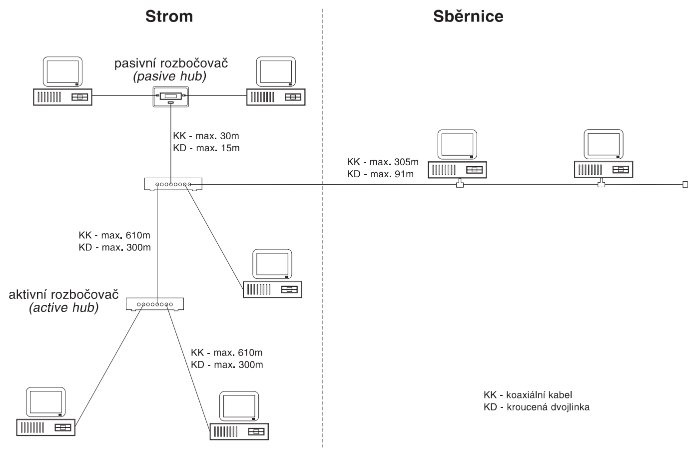
Síť typu Arcnet používá hvězdicovou, stromovou i sběrnicovou topologii a je založena na principu přístupové metody token passing. Typická přenosová rychlost je 2.5 Mbit/s (v současnoti až 20 Mbit/s) a celkové rozměry sítě se pohybují až do 6 km.



Obrázek 15: Rozvody sítě Ethernet

Základním stavebním prvkem jsou tzv. **rozbočovače** (*hubs*), které se dělí na aktivní a pasivní. Za aktivním rozbočovačem, který slouží zároveň k regeneraci signálu, může být umístěno až 8 resp. 16 stanic, a to jak počítačů, tak dalších rozbočovačů. Naopak za pasivním rozbočovačem se již další pasivní ani aktivní rozbočovač vyskytovat nesmí.

Síťové adaptéry pro síť Arcnet jsou nabízeny v osmi i šesnáctibitovém provedení. Jednotlivé adaptéry jsou rozlišeny osmibitovou adresou, která se nastavuje pomocí přepínačů umístěných přímo na adaptéru. V jedné síti se tak může vyskytovat nejvýše 255 stanic, neboť adresa s nulovou hodnotou je rezervována pro všesměrové vysílání všem připojeným stanicím.



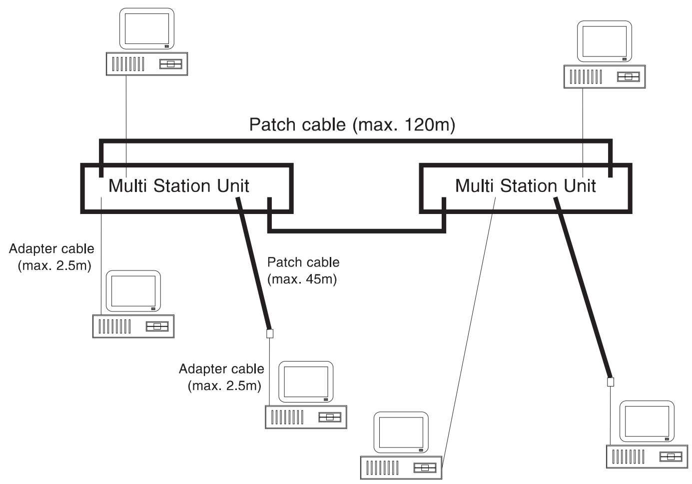
Obrázek 16: Rozvody sítě Arcnet

5.5.3 Token Ring

Počítačová síť Token Ring firmy IBM je založena na stejnojmenné přístupové metodě. Kabelový systém firmy IBM pro sítě IBM Token Ring je tvořen několika typy kabelů od kroucené dvojlinky až po optická vlákna.

Síť IBM Token Ring se vyskytuje ve variantě s přenosovou rychlostí 4Mbit/s nebo 16Mbit/s. Samozřejmě rychlejší varianta se více prosazuje na trhu, zvláště vzhledem k dobrým výkonům při velkém zatížení sítě.

I když přístupová metoda Token Ring předpokládá kruhovou topologii, není nutné v případě IMB Tokeng Ring toto omezení dodržet. Pomocí tzv. **kruhových rozbočovačů** (*Multistation Access Unit*) je možné dosáhnout hvězdicového propojení. MAU obsahuje přípojky pro maximálně 8 stanic a lze je propojit také mezi sebou navzájem.



Obrázek 17: Rozvody sítě IBM - Token Ring

5.5.4 LocalTalk

LocalTalk je rozhraní vyvinuté firmou Apple, které používá sběrnicovou topologii a přístupovou metodu CSMA/CA (*Carrier Sense Multiply Access with Collision Avoidance*). Základním přenosovým médiem je kroucená dvojlinka a přenosová rychlost se pohybuje okolo 230kbit/s, což je výrazně méně než v případě ostatních významných standardů.
Instalace LocalTalk je velice levná a jednoduchá. Každý systém firmy Apple je automaticky vybaven hardwarem a softwarem potřebným ke komunikaci a zapojení do sítě.

Později následovala varianta pro Ethernet - EtherTalk (přenosová rychlost 10 Mbit/s) a v současnosti i implementace pro Token Ring - TokenTalk (4 Mbit/s).

5.5.5 Srovnání

	Ethernet	Arcnet	Token Ring	LocalTalk
Přenosová rychlost	10 Mbit/s	2,5 Mbit/s	4 Mbit/s	23 kbit/s
Topologie	sběrnice	strom	kruh	sběrnice
Přístupová metoda	CSMA/CD	Token	Token Ring	CSMA/CA
Max. rozměr sítě	2 800 m	6 500 m	závisí na konfiguraci	
Max. počet stanic	1024	255	260	
Typy kabelů	thick a thin Ethernet, optický kabel	koaxial, kroucená dvojlinka, optický kabel	kabelový systém IBM	

5.6 Protokoly v sítích LAN dle modelu OSI

Úkolem komunikačního systému je doručení správné informace ve správný čas, správném pořadí a srozumitelně pro adresáta. To musí být provedeno způsobem daným určitými pravidly - **protokolem**. Protokol ustanovuje pravidla tak, aby byla srozumitelná oběma stranám. Ve světě existuje samozřejmě celá řada protokolů a další postupně vznikají. Ne všechny byly vyvinuty tak, aby splňovaly nároky uživatelů nebo mezinárodně uznávaný standard OSI.

Vzájemná komunikace v počítačové síti samozřejmě nemůže být zajištěna jediným protokolem, vznikají tedy tzv. **sady protokolů** (*Protocol Stack*). Každý z protokolů pak zajišťuje určitou část komunikačních funkcí a vzájemně mezi sebou spolupracují.

5.6.1 Novell Netware

Mezi nejvýznamnější sadu protokolů pro lokální sítě patří **XNS** (*Xerox Networking Protokol*), který byl původně vyvinut firmou Xerox. Firma Novell použila tento protokol jako vývojový základ pro svůj síťový operační systém založený na protokolu IPX.

IPX (*Internetworking Packet Exchange*)
Tento protokol je základem pro všechny ostatní protokoly, které pro komunikaci používají protokol IPX jako transportní prostředek. IPX patří k nezajištěným protokolům a uskutečňuje tedy pouze pokus o doručení paketu. Není použito ani číslování datových paketů, ani potvrzení o příjmu, takže při chybových situacích se neprovádí žádné opakované vyslání paketu. Funkce zajištění doručení přebírají protokoly vyšších vrstev. Pomocí IPX může každý uzel v síti přijmout nebo vyslat paket. Směrování paketů, pokud se síť skládá z více podsítí, zajišťují směrovací programy IPX, ale pro komunikující uzly jsou transparentní. Každé zařízení zapojené do IPX sítě je určeno jednoznačně svou vlastní logickou adresou.

Tato adresa se skládá ze dvou částí: **adresy podsítě**, která je volena správcem sítě, a **adresou uzlu**, která je hardwarově závislá a je určena adresou síťového adapteru.

SPX (*Sequenced Packet Exchange*)
Protokol, který přímo navazuje na IPX. Příjem každého paketu musí být v určité době potvrzen, jinak dojde k jeho opětovnému vyslání. Toto potvrzení zajišťuje protokol SPX. Používá se všude tam, kde je nutno zajistit spolehlivou komunikaci při službách, které nejsou poskytovány přímo pomocí níže popsaného protokolu NCP. Protokol SPX se používá převážně pro komunikaci mezi stanicemi. Typickým příkladem je dálkové ovládání serveru v Novell Netware v3.11 nebo „*talk*" mezi stanicemi.

SAP (*Service Advetizing Protocol*)
Zjednodušuje integraci nových serverů do sítě Novell Netware. Každé zařízení nabízející službu (např. souborový server, tisková obslužná stanice) oznamuje tuto skutečnost pravidelně prostřednictvím SAP. Z tohoto důvodu dokáže každé zařízení zapojené do sítě rozpoznat všechny služby, které jsou v síti nabízeny.

RIP (*Routing Information Protokol*)
Pracuje přímo nad protokolem IPX. Úkolem tohoto protokolu je vyhledat nejrychlejší cestu ze stanice, resp. směrovače, do určité podsítě. Jako měrná jednotka pro zjištění skutečné rychlosti přenosu jsou časové intervaly (*Ticks*) cca 1/18 sekundy (přesněji 1 sekunda je 18,21 *Ticks*). Zařízení vysílá požadavek na směrování ve formě univerzální zprávy (všesměrové vysílání) a směrovače odpovídají na tyto dotazy na základě směrovacích informací. Tyto informace si směrovače vyměňují v pravidelných časových intervalech.

NCP (*Netware Core Protokol*)
Je po IPX nejdůležitějším protokolem sítí Novell Netware, neboť implementuje vlastní realizaci služeb mezi serverem a stanicemi. NCP pakety jsou zasílány uvnitř IPX paketů. Obslužná stanice potvrzuje přijaté NCP pakety tak, že v paketu s odpovědí s vyžádanými daty opakuje sekvenční čísla paketů, které obdržela. Pracovní stanice je odpovědná za to, že odešle znovu ztracené pakety.

Příklady pokytovaných služeb mohou být přihlášení k *file serveru*, přenos a zamykání souborů nebo tiskové služby.

RM-OSI	Protokoly Novell Netware
Aplikační vrstva	vlastní aplikační programy
Prezentační vrstva	
Relační vrstva	NCP, MHS
Transportní vrstva	SPX, SAP, RIP
Síťová vrstva	IPX
Linková vrstva	CSMA/CD, Token Ring, Token Bus, FDDI
Fyzická vrstva	fyzické médium

6 Směrování

Pasáž o směrování je vlastně podrobným popisem činnosti síťové vrstvy referenčního modelu OSI. Popisuje činnosti, které se vykonávají v případě, že mezi odesílatelem a adresátem zprávy neexistuje přímé spojení. Tehdy síťová vrstva musí umět najít cestu, která vede přes jiné uzly sítě. Taková cesta se hledá buď pouze jednou za celou zprávu (uzavírá-li se virtuální okruh) nebo pro každý paket zvlášť (datagramová služba). Samotný algoritmus hledání cesty v síti se nazývá **směrování** (*routing*) a tento text seznámí čtenáře s několika známými způsoby hledání takových cest.

Algoritmů hledání cesty pro paket v počítačové sítí existuje celá řada. Jsou založeny na různé úrovni potřeby znalosti topologie sítě a dalších jejích parametrů - statických i dynamických. Základní vlastností směrovacího algoritmu je vždy to, že nesmí generovat chybné výstupy, nalezená cesta tedy musí být platná. Přitom je nutné požadovat, aby byl algoritmus jednoduchý a robustní, nebo alespoň s časem se stabilizující. To znamená, že je schopen se vypořádat s chybami na linkách, se změnami topologie, přidáním uzlu a podobně. Na nastalé změny topologie musí algoritmus reagovat co nejdříve, dále musí rovnoměrně využívat síť a směrovat pakety spravedlivě všem komunikujícím uzlům. Informace o tom, kudy budou zasílány pakety určené konkrétnímu adresátu, se uchovávají ve **směrovacích tabulkách** jednotlivých uzlů.

6.1 Statické a dynamické směrování

Základní rozdělení směrovacích algoritmů můžeme provést na dvě velké skupiny podle toho, jestli průběžně reagují na aktuální stav sítě a přizpůsobují mu své výstupy.

Algoritmy, které se dokáží přizpůsobit okamžitému stavu sítě, se označují jako **adaptivní** nebo **dynamické algoritmy** (*adaptive algorithms*) a směrování, které tyto algoritmy realizují, se nazývá **dynamické směrování**. Tyto algoritmy reagují například na změny okamžitého zatížení jednotlivých přenosových cest, výpadky linek, délky čekání ve frontách jednotlivých uzlů a podobně.

Druhá skupina algoritmů - **neadaptivní algoritmy** (*nonadaptive algorithms*) nevyužívá žádné dynamicky se měnící informace. Vycházejí výhradně ze statických tabulek, které vznikly v okamžiku inicializace směrování a pak byly zaneseny do jednotlivých uzlů. Statické algoritmy a **statická směrování** jsou použitelné zejména v konfiguracích, kde nedochází k výpadkům, nemění se výrazně zatížení linek ani topologie sítě.

6.2 Centralizované směrování

Existuje několik metod zajištění adaptivního algoritmu. Jednou z nich je shromažďování všech informací o aktuálním stavu sítě v jednom uzlu. Tento **centrální směrovací uzel** (*Routing Control Center*) provádí také všechna rozhodnutí o směrování a tato oznamuje ostatním uzlům. Zřejmou výhodou tohoto přístupu, která vychází z existence pohledu na síť jako celek, je naprosto přesné rozhodnutí o nalezení optimální cesty. Nevýhodou, která tento postup staví do úplně jiného světla, je potřeba extrémně silného výpočetního systému pro výpočet optimálních cest. Má-li být algoritmus skutečně adaptabilní, musí totiž kromě častého sledování skutečného stavu sítě (roste nezanedbatelně komunikační zátěž) probíhat také často výpočet optimálních cest, který je velmi náročný na výpočetní kapacitu. Vypočítané výsledky je nutno distribuovat zpět k uzlům, což opět zvyšuje komunikaci. Zcela zásadní problém nastává v okamžiku výpadku tohoto centrálního uzlu.

6.3 Izolované směrování

Nevýhody centralizovaných metod směrování nemají tzv. **izolované metody**, které jsou založeny na principu vyloučit potřebu komunikace s okolními uzly při rozhodování o lince, na kterou má být paket vyslán. K rozhodování jsou využívány pouze informace, které je uzel schopen zjistit sám. Jednou z těchto informací jsou čekací doby na jednotlivých výstupních frontách zpráv z daného uzlu.

Jednou z možností realizace je algoritmus, nazvaný algoritmem **horké brambory** (*hot potato*). Jde o snahu vyslat každý paket nebo datagram co možná nejdříve. Algoritmus tedy sleduje počty paketů čekajících ve výstupních frontách a nový paket zařadí do té fronty, která je momentálně nejkratší. Uzel se tedy nerozhoduje podle adresy ani nehledá nejkratší cestu pro přenos paketu, ale pouze se jej snaží co nejrychleji zbavit ve víře, že po jisté době přece jen dojde ke svému cíli. Tento algoritmus se v praxi většinou kombinuje s jinými algoritmy. Většinou mají algoritmy podobnou metodu jako pojistku v případě, že počet zpráv v některé výstupní frontě překročí únosnou mez.

Dalším algoritmem izolovaného směrování je algoritmus **zpětného učení** (*backward learning*). Tento algoritmus vychází z předpokladu, že jestliže z daného směru přicházejí pakety od jistého odesílatele A, je pravděpodobné, že bude výhodné danému uzlu A také pakety posílat. Každý uzel tedy eviduje ve svých tabulkách ze kterého směru dostává pakety pocházející od jiných uzlů. Takto se postupně učí, ve kterém směru se tyto uzly nalézají, a když pak potřebuje odeslat nějaký paket jinému uzlu, vyšle jej směrem, ze kterého dříve přijal od daného uzlu paket.
Optimistický předpoklad však v případě výpadků či změn neplatí a jestliže se uzly naučí cestu, která přestane platit, danou změnu vůbec nezaznamenají. Z toho důvodu uzly postupně své ,,vědomosti" zapomínají.

Poslední metodou izolovaného směrování je tzv. **záplavové směrování** (*flooding*). Tato metoda předpokládá, že přijatý paket je znovu odeslán všemi směry kromě toho, odkud sám přišel. Takový algoritmus je evidentně velmi robustní - dokáže se vyrovnat s libovolnou změnou topologie. Pakety jsou rovněž zaručeně dopraveny nejkratší možnou cestou. Současně se ale generuje velké množství duplicitních paketů zvyšujících zátěž sítě. tyto pakety je potom nutné důsledně rušit.

Umírněnější záplavová metoda je realizována v algoritmu **selektivního záplavového směrování**, při kterém není každý paket znovu vysílán všemi směry, ale pouze těmi, které jsou alespoň přibližně orientovány ke konečnému příjemci paketu. Neboli ze záplavy jsou vyřazeny ty linky, o kterých se ví, že nevedou k cíli.

6.4 Distribuované směrování

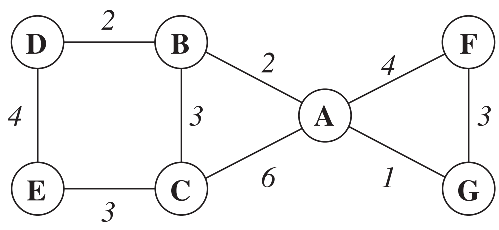
Přestože izolované směrovací algoritmy nepožadují centrální výpočetní kapacitu ani (s výjimkou záplav) negenerují přehnanou komunikační zátěž, chybí jim vlastnost deterministického rozhodnutí o jedné optimální cestě paketu. Optimistická metoda ani horký brambor nerozhodují o cestě, ale pouze předvídají tuto cestu na základě některých lokálních vlastností. Samotná deviza založená na nezatěžování sítě informacemi o jejím stavu nestačí.

Pokud připustíme, aby uzly přece jen vyměňovaly informace o stavu sítě (tentokrát ne s ohledem na centrální uzel), ale mezi sebou vzájemně, dostaneme tzv. **distribuované směrování** (**distributed routing**). To předpokládá, že si jednotlivé uzly pravidelně vyměňují informace o stavu sítě, a podle nich si pak samy volí příslušné cesty.

Umožníme-li výměnu stavových informací mezi jednotlivými uzly sítě, můžeme implementovat distribuovanou verzi algoritmu hledání nejkratších cest v síti. Hledání je založeno na prohledávání grafu, který ale na rozdíl od centrálního grafu v případě centralizovaného směrování je uložen po částech v jednotlivých uzlech.

Fungování uvedeného algoritmu si objasníme podrobněji. Jak budeme měřit délku cesty (a podle čeho stanovíme optimum)? Délku cesty budeme měřit pomocí ohodnocení jednotlivých přímých spojů mezi uzly (v nejjednodušším případě ohodnotíme všechny přímé spoje číslem 1 - pak určujeme počet meziuzlů, kterými zpráva prochází, jako její délku). Mějme například síť nakreslenou na obrázku 18. Na něm jsou vyznačena ohodnocení přímých spojů (uzly musí předem znát svoji vzdálenost od všech svých sousedů). Cílem algoritmu je nalézt vzdálenost z aktuálního uzlu do stanoveného. V první fázi algoritmus nastaví vzdálenosti do nesousedních uzlů na ∞.

Vlastní algoritmus pak probíhá v opakujících se krocích. V každém z nich se každý uzel dotáže svých bezprostředních sousedů, jaké jsou jejich vzdálenosti od ostatních uzlů. Podle toho si pak odvozuje i své vlastní vzdálenosti od těchto uzlů. Ukažme na uzlu A, jak se odvození vlastní vzdálenosti realizuje. Uzel A se v prvním kroku od svého souseda B dozví, že jeho vzdálenost od uzlu D je 2. K ní si uzel A připočte



Obrázek 18: Graf sítě

svou vzdálenost od uzlu B, tj. 2, a do své směrovací tabulky si poznačí, že cesta do uzlu D vede přes uzel B a má délku 4. Zatím však nezná všechny možné cesty do uzlu D, a tak neví, zda je to cesta nejkratší. Proto se bude v dalších krocích znovu ptát všech svých sousedů, jestli přes ně nevede cesta ještě kratší.

Každý uzel volí vždy minimální ohodnocení cesty k jinému uzlu. Na situaci z obrázku 18 zjistí uzel A při inicializaci, že jeho vzdálenost od uzlu C je 6 (přímý spoj). V druhém kroku se uzel A dotazuje sousedů, jak daleko mají oni k testovanému uzlu. Takto zjistí, že uzel B je od C vzdálen 3, takže celkově je kratší spoj A-B-C než A-C. Takových kroků provádí algoritmy tolik, až nedochází ve směrovacích tabulkách ke změnám.

Právě naznačený algoritmus distribuovaného směrování byl používán v síti ARPA (základu dnešní sítě Internet) a jednotlivé kroky algoritmu zde probíhaly s intervalem 640 milisekund. Ukázalo se však, že režie je příliš vysoká (předávání informací mezi sousedními uzly je založeno na zasílání směrovacích tabulek).

7 Propojování sítí

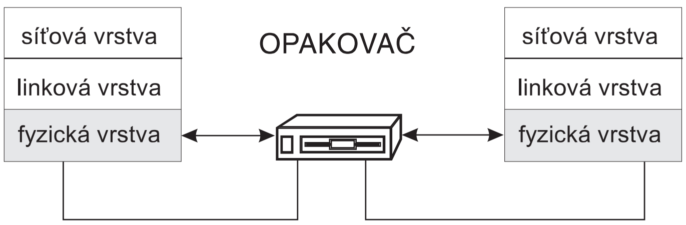
Až do této chvíle jsme uvažovali pouze v dimenzích jediné počítačové sítě, a nebrali v úvahu její možné vazby na sítě jiné. Otázka propojování sítí je dnes velmi aktuální, neboť je užitečné, aby existující sítě byly vzájemně propojeny, a to pokud možno co nejuniverzálněji a současně co nejméně ,,viditelně" (tak, abychom o propojení museli v okamžiku běhu aplikace vědět co nejméně). Ideální případ by z hlediska uživatele byl, kdyby všechny zdroje mohly být součástí jedné sítě. Tím by se nemusely specifikovat adresy sítí a podobně. V této kapitole rozebereme principiální možnosti vzájemného propojování počítačových sítí.

Propojením dvou a více sítí (stejného či různého typu) vzniká opět počítačová síť, která v anglickém textu bývá označována jako **internetwork**. Občas bývá tento pojem zaměňován s pojmem Internet. Internet (s velkým I) je sice také síť vzniklá propojením jiných sítí, ale označuje celosvětovou síť, která se vyznačuje specifickými vlastnostmi. Ačkoliv základní principy propojování sítí zůstávají stále stejné, objevují se na trhu stále nové komponenty, které lze k uvedenému účelu použít. V současné době existuje řada různých koncepcí, jak počítačové sítě navzájem propojovat. Pro jejich pochopení je ale klíčovým momentem poznání toho, že vzájemné propojení je možné realizovat na různých úrovních vrstvového síťového modelu - od fyzické až po aplikační vrstvu. Podle zvoleného řešení se také propojovací prvek chová.

Základní myšlenka vzájemného propojení je založena na tvorbě speciálních propojovacích prvků (Internetworking Unit resp. IWU) nebo **relay**. Tyto prvky se připojí do obou propojovaných sítí (mají rozhraní do obou sítí) a na určité logické úrovni jsou schopny přenášet informace z jedné sítě do sítě druhé. Propojovací zařízení může pracovat na úrovni elektrického signálu jako zesilovač nebo na vyšších úrovních, kde je schopno rozlišit adresy v záhlavích paketů.

7.1 Opakovače

Opakovač (*repeater*) je propojovací zařízení, které je pouhým zesilovačem elektrických signálů přenášených sítí. Tak je tomu například v lokálních sítích typu Ethernet, kde umožňují dosáhnout celkové délky kabeláže až 2, 5 kilometru, ale jednotlivé připojovací obvody (tzv. **transceivery**) jsou schopné generovat elektrické signály s dosahem jen asi 500 metrů. Při potřebě delšího vedení je nutné sestavovat celé kabelové vedení ze segmentů (souvislých úseků kabelů) délky maximálně 500 metrů (tlustý Ethernet) nebo 200 metrů (tenký Ethernet). Jednotlivé segmenty se pak musí spojovat pomocí zařízení, která se v tomto případě nazývají opakovače a fungují právě jako zesilovače elektrických signálů. Opakovače pracují na úrovni fyzické vrstvy referenčního modelu OSI (obr. 19), protože vzhledem k zesilování elektrických signálů ,,vnímají" jen jednotlivé přenášené bity, ale nikoli celé bloky (rámce) dat.



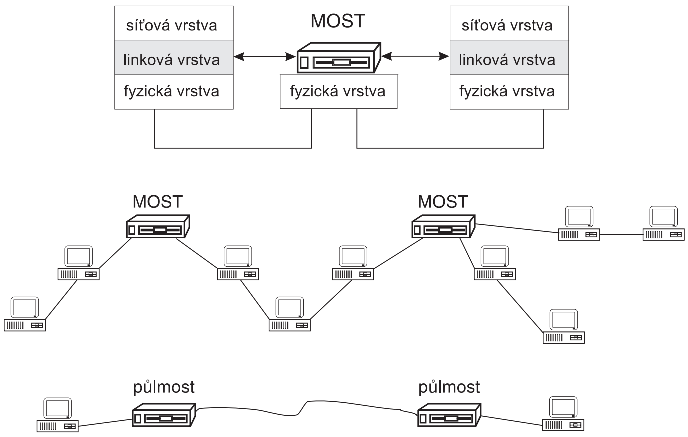
Obrázek 19: Opakovač

Z uvedené specifikace vyplývá, že opakovače propouštějí veškerý signál z jednoho segmentu do druhého, pouze ho zesilují. Nejsou tedy schopny rozlišit, jestli pakety, které přenášejí, směřují do daného segmentu sítě nebo ne.

7.2 Mosty

Posledně uvedená vlastnost opakovačů může znamenat zbytečné zatěžování segmentu signálem. Jako výhodnější se jeví možnost, aby propojovací prvek dokázal správně rozpoznat, která data mohou zůstat

,,uvnitř" jednoho segmentu sítě a nezatěžovat jimi provoz v ostatních segmentech. K tomu je ale nutné, aby propojovací zařízení mělo jiný pohled ,,vnímání" přenášených dat. Musí rozpoznat celé přenášené rámce a znát jejich formát natolik, aby dokázalo zjistit fyzickou adresu jejich odesílatele a příjemce (a podle toho je pak propustit do sousedního segmentu či nikoli). To ovšem znamená, že takovéto propojovací zařízení (**most** - *bridge*) musí pracovat na úrovni linkové vrstvy (viz obr. 20).



Obrázek 20: Most

Práce v jiných logických vrstvách není jediným rozdílem opakovače a mostu. Zatímco opakovač nemá paměť a přenášená data (signály) zpracovává průběžně, most pracuje na principu ,,store and forward" (přijmi a předej dál). Most tedy z každé strany sítě průběžně přijímá jednotlivé datové rámce a podle adres v nich se rozhoduje, zda je předá na opačnou stranu nebo ne. Existuje přitom více algoritmů, které mohou mosty v této souvislosti používat. Nejjednodušší varianta algoritmu je ta, při které most průběžně vyhodnocuje odesílatele jednotlivých rámců a podle toho, ze kterého směru příslušný rámec přijal, si pak sám odvozuje umístění jednotlivých uzlů. Tímto způsobem se časem ,,naučí" topologii sítě. Zpočátku, kdy topologii ještě nezná, předává všechny rámce do všech ostatních segmentů.

Uvedená metoda (kromě své jednoduchosti) je výhodná v tom, že nevyžaduje žádné konfigurování mostu (ten se vše potřebné o topologii a umístění jednotlivých uzlů naučí sám. Jednotlivé uzly v síti přitom nemusí o jeho existenci vůbec vědět - odtud název **transparentní most** (*transparent bridge*). Transparentní mosty lze použít ovšem jen na topologiích hiearchických (strom), kde mezi každými dvěma uzly existuje jen jediná cesta. Pro obecné topologie se používají složitější algoritmy.

Opakovače i mosty mohou být konstruovány na propojení dvou i více segmentů. Ve druhém případě (obr. 20) jde o tzv. **vícevstupové opakovače, mosty** (*multiport repeater, bridge.*

Pomocí mostů lze propojit také sítě, které jsou od sebe vzdáleny a jsou propojeny například optickým kabelem nebo pevným telefonním okruhem. Tento druh propojení umožňují tzv. **vzdálené mosty** (*remote bridge*). U vzdáleného mostu se jedná o dvě relativně samostatné poloviny mostu (**půlmosty** (*halfbridge*) (viz obrázek 20), které jsou mezi sebou propojeny zmíněným pevným okruhem. Takto lze například propojit dva segmenty lokální sítě umístěné ve dvou objektech na opačných stranách města. Výsledkem propojení je jediná logická síť (z pohledu síťové vrstvy a všech vyšších je totiž existence mostů transparentní).

Další užitečnou vlastností některých mostů je schopnost filtrovat rámce podle jejich odesílatele nebo příjemce, času, zatížení sítě apod. Uvedené parametry lze nastavit přímo manuálně v tabulkách mostu. Jedná se o *routing bridges*, které administrátoru sítě umožňují regulovat přenosy mezi jednotlivými segmenty. Mosty tak poskytují možnost **zvýšené bezpečnosti** mezi spojovanými sítěmi.

Důležitým úkolem mostů je rovněž rozšiřovat dosah lokálních sítí. Podstata je, jako u většiny propojovacích prvků, v rozčlenění sítě na relativně samostatné a nezávislé LAN segmenty a jejich propojení mostem. Správné rozsegmentování sítě je velmi důležité, neboť v případě silného provozu přes most zjistíme citelné zpomalení přenosu.

Srovnání použití a vlastností mostů a opakovačů.

	opakovač	**most**
cena	levnější	dražší
rychlost	vyšší	nižší
použití při	slabé zátěži	silné zátěži

Použití mostů je užitečné například na topologii využívající páteřní síť typu Ethernet (viz. strana 12). Jedná se o sběrnicovou topologii, ve které páteř je segment, určený zejména pro vzájemné propojení ostatních segmentů. Výhodou této topologie je skutečnost, že spojení mezi kterýmikoliv dvěma body sítě prochází vždy nejvýše přes dva mosty. Klasické použití je v patrové budově, kde páteřní síť vede vertikálně a jednotlivé větve horizontálně v patrech.

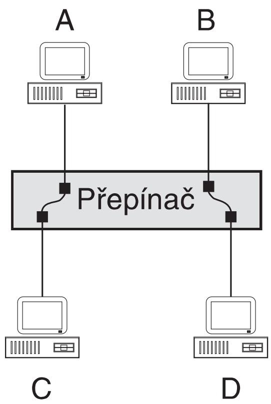
Jak již bylo řečeno, mosty jsou pro vyšší vrstvy než síťová naprosto transparentní, to znamená, že jsou schopny spolupracovat s libovolným protokolem vyšších vrstev. Jednotlivé mostem propojené segmenty tvoří z pohledu síťové vrstvy (a všech vyšších) jediný logický celek, který má jedinou společnou síťovou adresu.

7.3 Přepínač

Technologie přepínání výrazně zlepšuje propustnost sítí. Základním principem **přepínače** (*switch*) je umožnit současnou komunikaci mezi segmenty sítě podobně, jako když přetížené cestě přibude jeden jízdní pruh, takže po síti (např. Ethernet) může být současně přepravováno více rámců - přepínače tohoto typu pracují na úrovni linkové vrstvy. Z toho vyplývá, že pro přepínač není podstatný typ paketů síťové vrstvy (IPX, IP, ...).

Přepínání v sítích je modelově podobné přepínání v telefonní ústředně. Spoj mezi dvěma účastníky se může každé sezení (hovor) měnit, což ale vyšší vrstvy neregistrují a ani je to nezajímá. Jediná jejich starost je totiž získat volnou linku pro spojení.

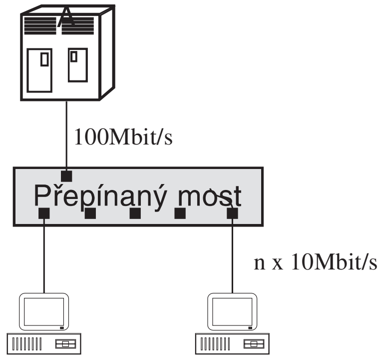
Vysvětlíme si fungování přepínače v síti Ethernet. Přepínané prostředí je podobné dvoubodovému spoji mezi segmenty, ale takto se chová pouze po dobu nezbytně nutnou pro průchod rámce mezi segmenty sítě. Který konkrétní obvod v přepínači toto spojení zajistil, je pro aplikaci nepodstatné. Na obrázku 21 posílá



Obrázek 21: Realizace spojů v přepínači

uzel A rámec uzlu C. Přepínač propojí pouze porty, na které jsou připojeny segmenty uzlů A a C. Současně může posílat uzel B rámec uzlu D a opět dojde k propojení pouze těchto segmentů. Obě propojení existují paralelně. Kdybychom tedy přepínačem propojovali síť Ethernet s celkovou efektivní průchodností 10Mbit/s, zdvojnásobili bychom tuto průchodnost použitím přepínače na obrázku 21. Prakticky je možné zvyšovat průchodnost volbou počtu portů na přepínači - máme–li přepínač s n porty, dosáhneme $n/2$krát širšího pásma, tedy $n/2$krát vyšší průchodnosti sítě.

Efektivita přepínaného Ethernetu je závislá na struktuře sítě a také použitém protokolu. Například v sítích peer-to-peer může přepínač výrazně zrychlit komunikaci, jestliže je síť navržena jako několik relativně nezávislých dvojic uzlů. Naopak v prostředí klient-server je použití sporné.



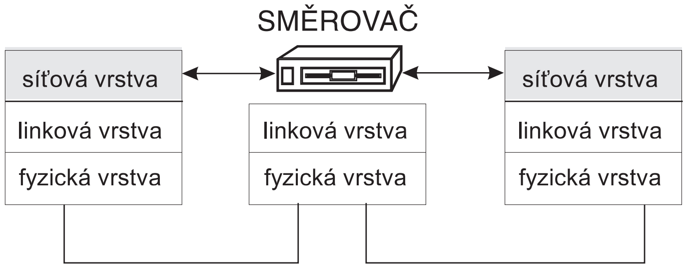
Obrázek 22: Přepínaný most

Přepínače jsou často kombinovány s mosty. Pak je možno uspokojit požadavek na přepínání sítě v prostředí klient-server, a to například přepínaným mostem na obrázku 22. Přepínaný most na obrázku má jeden port s výstupem 100Mbit/s pro připojení serverového počítače a n portů pro připojení segmentů s klienty. Na místo několika segmentů může být připojen opět přepínaný most, nebo jen most, čímž se vytvoří hierarchie (obdoba stromové topologie).

7.4 Směrovač

Společná síťová adresa více segmentů může být nežádoucí z důvodu potřeby identifikace uzlů, samostatnosti správy podsítě a podobně. Propojení pomocí mostů nedovolí implementovat samostatnost propojených segmentů (vlastní síťová adresa), propojení lokálních sítí různých typů, propojení dvou lokálních sítí přes rozlehlou síť a také některé obecné topologie. K tomu musíme použít obecnější řešení, které představuje propojovací zařízení na úrovni vyšší (síťové) vrstvy. Toto zařízení se nazývá **směrovač** (*router*) a jeho princip ukazuje obrázek 23. Směrovač dokáže rozpoznat obsah jednotlivých rámců na úrovni linkové vrstvy, formát paketů přenášených v rámcích a tyto informace zužitkovat.

Úkolem směrovače je v podstatě stejný jako základní úkol síťové vrstvy - postarat se o doručení paketů od odesílatele ke konečnému příjemci. Směrovač musí rozhodovat o tom, kudy mají dále odeslat každý paket tak, aby se dostal ke svému cíli. Samotné činnosti **směrování paketů** se také říká *routing*. Směrování paketů bylo popsáno v kapitole 6, jen v kostce připomeňme, že existují buď statické (předem nakonfigurované) nebo dynamické (měnící se s časem) algoritmy směrování. Základní představu o směrovači lze získat na obrázku 23

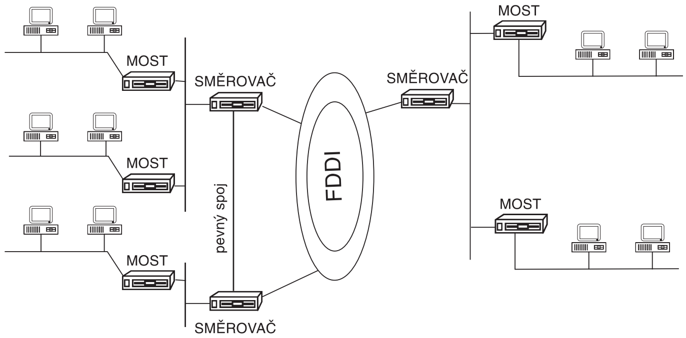


Obrázek 23: Směrovač

Z pohledu vnímání propojovacího prvku vrstvami síťového modelu existuje zásadní rozdíl mezi směrovačem a mostem, a to v tom, že pro ostatní entity na úrovni síťové a linkové je směrovač viditelný (není transparentní zařízení). Směrovač má svoji adresu – pakety, které má směrovat, musejí být adresovány přímo jemu (mosty naopak zachycují provoz v každém připojeném segmentu). Proto směrovače zpracovávají méně rámců než mosty, ale jejich zpracování je výrazně náročnější.

Podstatným faktem při používání směrovačů je skutečnost, že propojované sítě musí používat stejný protokol na úrovni síťové vrstvy (ve které směrovač pracuje). Přitom na úrovni linkové a fyzické vrstvy toto platit nemusí. Směrovače tedy obvykle obsahují více různých rozhraní (portů) a dovolují tak propojit různé lokální sítě podle standardů IEEE 802 a být vzájemně propojeny optickým kabelem.

Příkladem možného propojení lokálních sítí ve více budovách je obrázek 24. Na něm jsou páteřní sítě v budovách propojeny směrovači, které jsou propojeny optickými přenosovými cestami nebo pevným okruhem. Jednotlivé segmenty jsou ke své páteřní síti připojeny mostem.



Obrázek 24: Propojení lokálních sítí mosty a směrovači

Multiprotokolové směrovače

Multiprotokolový směrovač řeší problém potřeby stejného síťového protokolu v propojovaných sítích. To je totiž mnohdy omezující podmínka zejména v konfiguracích, kde koexistuje více rodin protokolů (TCP/IP, IPX/SPX, DECnet a další). V heterogenních sítích není tato koexistence výjimkou.

Problém propojení heterogenních sítí lze řešit dvěma způsoby - konverzí protokolů a směrováním více protokolů současně. Konverze protokolů se v praxi neprosadila z důvodu složitosti a nízké spolehlivosti. Majoritní výrobci dnes nabízejí tzv. **multiprotokolové směrovače** (*multiprotocol routers*), které jsou schopny pracovat současně s více různými protokoly. Multiprotokolový směrovač musí umět rozpoznat typ paketu, který obdrží od linkové vrstvy, a podle něho aplikovat směrovací algoritmus příslušného protokolu.

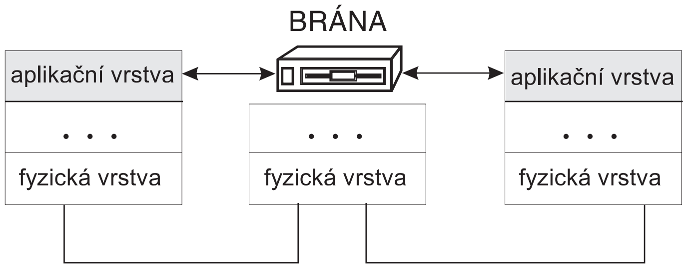
7.5 Brouter

V některých konfiguracích, které se navíc mohou dynamicky měnit, není někdy předem zřejmé, jestli na daném místě má být použit směrovač nebo most. Z toho důvodu se na trhu vyskytují zařízení nazývaná **brouter** (*bridge/router*), která spojují vlastnosti obou. Jde o zařízení, které se snaží fungovat jako směrovač, a teprve v okamžiku, kdy pro nějaký paket neumí aplikovat směrovací algoritmus, předá původní rámec dál stejně, jako by ho předával most. Taková zařízení se tedy umí při vysoké inteligenci směrování známých protokolů vyrovnat i s protokoly, které směrovat nelze, protože nepočítají se síťovou vrstvou (např. DECLAT, NetBIOS).

7.6 Brána

U směrovačů jsme uvedli, že konverze protokolů by byla náročná a nespolehlivá, někdy je však potřeba propojit sítě, které používají naprosto odlišné rodiny protokolů, a zde se konverzi nevyhneme. Zařízení, které konverzi realizuje, označujeme jako **brána** (*gateway*). Brány pracují vždy na takové úrovni, na které jsou schopny zajistit bezpečnou konverzi, často až v aplikační vrstvě (obrázek 25).

S pojmem brána se setkáváme někdy v jiném významu, než je právě zde popsán. Například v sítích s protokoly TCP/IP se vžil název brána (gateway) pro směrovače.



Obrázek 25: Brána

8 Topologie vysokorychlostních sítí LAN

Využívání počítačových sítí (lokálních i rozlehlých) se od dob jejich zavádění značně změnilo. Zatímco dříve se jednalo zejména o jednoduchý přenos souborů, komunikaci se servery (komunikační, souborový, tiskový), dnes se setkáváme s multimediálními aplikacemi, aplikacemi pracujícími v reálném čase (real--time applications). V souvislosti se stále rostoucím používání osobních počítačů roste také počet uzlů v lokálních sítích. Velmi rychlé a výkonné osobní počítače, které zpracovávají složité úlohy, vyžadující mnoho dat, způsobují na sítích, konstruovaných za jiným účelem, stav zahlcení. Sítě Ethernet případně Token ring, které jsou velmi rozšířené a podporované většinou výrobců komponent, nestačí splňovat požadavky na přenosovou kapacitu. Vzhledem k jejich masovému rozšíření se hledají řešení, která by vyžadovala co nejméně investičních zásahů do stávajícího vybavení. Ukážeme zde několik principiálních řešení, které mohou pomoci - vždy se však jedná o zakoupení několika propojovacích prvků, síťových adaptérů do stanic, případně nových propojovacích kabelů. Ethernet v tom stavu, jaký jsme byli zvyklí používat (max. rychlost 10Mbit/s sdílených mezi všechny uzly segmentu), se stává prakticky minulostí. Dnešním trendem v lokálních sítích jsou přenosové rychlosti 100Mbit/s (Fast Ethernet, 100VG–AnyLAN).

8.1 Zahlcení sítě Ethernet

Jaké příznaky lze sledovat v případě zahlcení sítě? Zde je nutné vycházet z jistých zkušeností, protože tento stav není signalizován žádnou kontrolkou nebo ukazatelem stavu. Z pohledu uživatelů, odkud se odezva dostaví nejdříve, lze sledovat

- problémy s přihlášením do sítě, zvláště např. z prostředí MSWindows;
- výpadky sítě a neočekávaná přerušení komunikace;
- výrazné zpomalení práce síťových aplikací, přičemž lokální aplikace svoji rychlost nezměnily;
- v podstatě nedeterministické chování sítě, která reaguje podle toho, kolik stanice je právě aktivně připojeno.

Uvedené příznaky mohou sice signalizovat například chyby na kabeláži nebo vadný konektor na segmentu, používání nových protokolů (aplikací), ale jestliže začnou vznikat v okamžicích zjevné aktivity všech uzlů a neobjevují se například v odpoledních hodinách (kdy aktivita není), jedná se o zahlcení sítě. Správce sítě Ethernet by měl zaregistrovat ukazatele jako okamžitou utilizaci sítě nad 50%, střední utilizaci nad 10% nebo počet kolizí nad 5%.

Příčiny zahlcení sítě Ethernet vycházejí z povahy této sítě (přístupová metoda CSMA/CD viz strana 37) a jsou většinou následující:

- příliš mnoho aktivních uživatelů na jednom segmentu
- provozování aplikací, které významně zatěžují síť
- existence velmi rychlých zařízení, které si vymáhají přístup k síti. Tato extrémně rychlá zařízení (servery, stanice) si tak přisvojují celou šířku pásma na většinu času.

Existuje několik řešení, která úspěšně odstraňují zahlcení sítě Ethernet. Zmíníme se o třech:

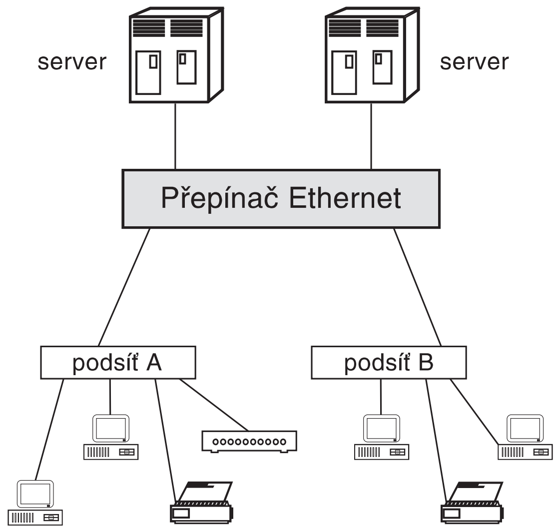
- zapojení přepínačů, které umožní vytvořit paralelní cesty
- rozdělení (segmentování) sítě pomocí propojovacích prvků most a směrovač
- koncepční změna - přechod k rychlejší síti (např. 100Mbit/s)

8.1.1 Přepínače v síti Ethernet

O metodě přepínání v sítích pojednává kapitola 7.3. Přepínač v síti Ethernet je složen z procesorů, které fungují jako komunikační mosty se dvěma porty - filtrují a dále předávají rámce podle vnitřní tabulky a obsahu rámce. Přitom znalost jednoho procesoru (je to most - učí se topologii) je přenášena i do tabulek ostatních procesorů. V okamžiku, kdy přepínač zná port cílového uzlu, přepne všechny následující rámce příslušný procesor na příslušný port.

V porovnání s mostem má typický přepínač v síti Ethernet **menší zpoždění** mezi přijetím rámce a jeho odesláním. Most (metoda store-and-forward) nejprve přijme celý rámec (store), pak se rozhodne o cílovém segmentu a odešle rámec na příslušný port. Přepínač identifikuje cílový port podle prvních několika (šesti) oktetů rámce a jestliže na segmentu příslušnho portu není provoz, okamžitě začne rámec vysílat. Nečeká tedy na jeho načtení do vyrovnávací paměti jako most, čímž dosahuje konstantního zpoždění kolem $40\mu s$ na rámec.

Obrázek 26 ukazuje řešení topologie sítě s několika servery a logicky seskupenými pracovními stanicemi a jimi využívaných zařízení (tiskárna, modem, ...). Pracovní stanice a jejich zařízení jsou seskupena do segmentů a každý segment pak připojen na samostatný port přepínače (skupina navzájem je propojena buď sběrnicí nebo jiným mechanismem). Také servery připojíme každý na samostatný port přepínače.



Obrázek 26: Topologie sítě s přepínačem

Existují také situace, kdy přepínače nejsou vhodné. Jmenujme zejména tyto:

- **Přepínač nenahradí směrovač**. Přepínač neumožňuje propojení víceprotokolárních sítí, neumožní ochranu bezpečnosti ve stejné míře jako směrovač.
- **Přepínač neřeší zahlcení spoje vedoucího k jednomu uzlu**. V tomto případě není možné paralelní vytvoření více cest, neboť existuje jediný možný spoj vedoucí k danému síťovému zdroji.

Jaké výhody přináší uvedené řešení přepínaného Ethernetu?

- **Zvýšení propustnosti sítě.** Podle výše popsaného vytvoření paralelních cest mezi segmenty lze dosáhnout násobného zvýšení propustnosti sítě Ethernet mezi logicky samostatnými segmenty. Míra zvýšení propustnosti je dána logickým rozčleněním uzlů do nezávislých segmentů, množstvím těchto segmentů a počtem portů na přepínači.
- **Lze použít stávající kabeláž Ethernetu**. Důležitým atributem této metody je využití stávající kabeláže včetně síťových adaptérů u stanic a serverů. Seskupení uživatelů lze provést i na stávající kabeláži. Jediná investice se týká propojovacího prvku přepínače.

8.1.2 Použití propojovacích prvků most a směrovač

Tato metoda je podrobně popsána v kapitolách 7.2 a 7.4.

8.1.3 Přechod k rychlejší síti - 100VG-AnyLAN

Jestliže situace vyžaduje propojení stanic sítí s vyšší garantovanou propustností než 10Mbit/s - například 100Mbit/s pro běh multimediálních aplikací, nebo se jedná o zavádění nové lokání sítě s perspektivami dalšího růstu, uvažují dnes správci spíše o komplexním řešení.

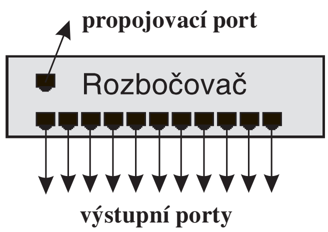
Tím může být například síť 100VG-AnyLAN, která vychází ze standardu IEEE 802.12 a která pracuje s rámci Ethernet i Token-Ring. Tato velmi podstatná vlastnost totiž zaručuje, že pro aplikační úlohy nebude přechod na novou komunikační podsíť překážkou. Navíc, síť 100VG-AnyLAN může být připojena k libovolné síti IEEE 802.3 (Ethernet) nebo IEEE 802.5 (Token Ring) pomocí rychlého mostu, nebo k FDDI či ATM páteři pomocí směrovače.

8.2 Síť 100VG-AnyLAN

V minulé kapitole jsme o síti 100VG-AnyLAN uvedli, že je kompatibilní se sítěmi Ethernet a Token Ring a je snadno propojitelná s páteřními sítěmi. Standard 100VG-AnyLAN zaručuje vysokou úroveň bezpečnosti až do úrovně filtrování jednotlivých paketů. Z pohledu správy sítě lze jen uvítat možnost monitorování a snadného konfigurování sítě.

8.2.1 Technické komponenty

Síť 100VG-AnyLAN je navržena jako hvězda, která se skládá z následujících komponent:



Obrázek 27: Rozbočovač v síti 100VG-AnyLAN

rozbočovače (*huby*)

Rozbočovač (hub) je propojovací zařízení, které je v podstatě ekvivalentem víceportového opakovače (repeateru). Rozbočovač je středem každé sítě 100VG-AnyLAN a každé síťové zařízení musí být připojeno k nějakému rozbočovači.

Rozbočovač má dva druhy portů (obrázek 27) - výstupní porty k připojení jednotlivých síťových zařízení a propojovací porty pro připojení k rozbočovači vyšší úrovně v hierarchii. Tím naznačujeme také možnost víceúrovňové hierarchické topologie (strom). Přitom **každý port** je možné nakonfigurovat na **privátní mód**, ve kterém přijímá pouze pakety jemu adresované, a **promiskuitní mód**, ve kterém přijímá všechny pakety. Promiskuitní mód se používá pro připojení síťových analyzátorů nebo mosty a směrovače.

koncové uzly
Koncovým uzlem je libovolná pracovní stanice, server nebo síťové zařízení vybavené síťovým adaptérem. Jako síťový adaptér lze využít i adaptéry 10Base-T (Ethernet) nebo Token Ring.

síťové spoje
(kroucená dvojlinka nebo optický kabel)

další volitelná zařízení (most, směrovač, přepínač)

9 Rozlehlé počítačově sítě

9.1 Vlastnosti a využití

Jak již bylo řečeno, rozlehlé sítě se rozprostírají na velké vzdálenosti a jejich hlavní funkcí je zabezpečit přístup k obrovským množstvím informací. Nejstarší WAN jsou vybudovány pomocí výkonných sálových počítačů firmy IBM nebo jiných počítačů, které mají analogickou výpočetní kapacitu. S rozvojem sítí postavených na protokolové sadě TCP/IP (viz dále) a díky trendu spočívajícím v distribuci databází, informací a služeb je tato mohutnost často nahrazována souhrnem několika méně výkonných počítačů středního typu. Tyto jsou však umístěny v rámci jedné sítě, jejíž výkon je pak srovnatelný s případem, kdy je použit pouze jeden výkonný stroj a síť terminálů. Osobní počítače se jako uzly WAN používají velmi zřídka.

Sítě WAN se vyznačují většinou nižšími přenosovými rychlostmi než sítě LAN (obvykle do 1 Mbit/s). Uvedená hranice však je pouze otázkou času, protože již v současné době (srpen 1996) existují reálné plány na propojení uzlových počítačů superpočítačových pracovišť na vysokých školách v České republice sítí s přenosovou kapacitou $30 Mbit/s$. Sítě WAN mají různé oblasti použití a to, čím se od sebe jednotlivé sítě liší, je dáno především vlastními službami dané sítě. Při zřizování WAN je třeba brát v úvahu především tyto faktory:

- objem přenášených dat,
- přístupnost služeb v čase (interaktivní, neinteraktivní),
- rychlost a kvalita přenosu,
- možnost řízení a správy sítě,
- vlastní výčet poskytovaných služeb.

Služby poskytovné rozlehlými počítačovými sítěmi jsou velmi rozmanité a zahrnutí velké množství činností. Mezi nejzákladnější, poskytované téměř všemi rozlehlými sítěmi, patří tyto služby:

- elektronická pošta
- diskusní kluby, elektronické časopisy
- vzdálené přihlášení
- přenosy souborů
- adresářové a vyhledávací služby

V následujích kapitolách se pokusíme shrnout obecné charakteristiky některých standardních služeb, které sítě WAN poskytují a vysvětlíme si případně standardy a protokoly, které jsou při tom používány.

9.1.1 Elektronická pošta, X.400

Potřeba vzájemné komunikace, a to nejen v počítačovém smyslu, je bezpochyby jednou z nejdůležitějších potřeb člověka, zvláště v dnešní době. Proto se i v počítačových sítích stala jednou z nejzákladnějších (a nejpoužívanějších) služeb elektronická pošta, která ve svých principech vychází z klasické „papírové" pošty.

Stejně jako pro jiné služby v počítačových sítích i pro elektronickou poštu vzniklo doporučení, kterému by měly vyhovovat existující poštovní systémy tak, aby byla možná vzájemná komunikace mezi nimi. V roce 1984 CCITT vydalo doporučení X.400, které se stalo základem standardu OSI - MOTIS. V roce 1988 bylo upraveno doporučení X.400 tak, aby vyhovovalo tomuto standardu.

Každý systém elektronické pošty by měl poskytovat těchto 5 tříd služeb:

- **Kompozice**, což je proces tvorby zpráv a odpovědí na ně. Ačkoli může být použit ke tvorbě těla zprávy libovolný textový editor, systém by měl poskytovat pomoc při adresování zprávy, vyplňování některých povinných částí záhlaví, které je připojeno ke každé zprávě. Například při odpovědi na zprávu automaticky vyplní adresní pole na základě odesílatele původní zprávy.
- **Přenos**, který zajišťuje doručení zprávy od odesílatele k příjemci zprávy. Přesněji, za použití prezentační vrstvy je vytvořeno potřebné spojení, přenesena zpráva a ukončeno spojení. Celý tento proces musí zajistit poštovní systém bez zásahu uživatele.
- **Hlášení** pro odesílatele zprávy, která informují o tom, co se se zprávou stalo: zda byla doručena, odmítnuta, ztratila se a podobně. Tato hlášení jsou opět generována poštovním systémem bez vnější obsluhy.
- **Konverze**, která může být vyžadována, aby doručená zpráva mohla být zobrazena na příslušném zařízení (display, tiskárna, fax) příjemce, pokud je jiné než zařízení odesílatele.
- **Zpracování** zprávy příjemcem. To znamená, co se se zprávou stane po přečtení příjemcem: zda se zruší nebo uloží pro pozdější použití.

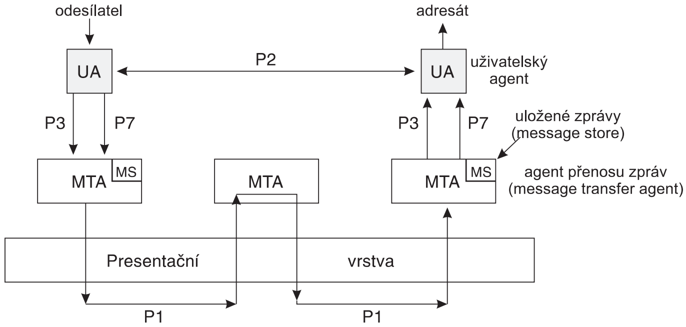
Kromě těchto základních služeb může poštovní systém poskytovat velké množství různých rozšíření. Například odeslání zprávy až v určitý stanovený den a hodinu.

Jiný příklad může být situace, kdy je příjemce dlouhodobě nepřítomen. Poštovní systém může každému uživateli automaticky odpovědět, že přijímající bude přítomen až ve stanovený čas a podobně. Někteří uživatelé opakovaně zasílají zprávy celé skupině příjemců. Poštovní systém může udržovat seznam těchto elektronických adres (*distribution list*). Pokud je potom zpráva zaslána tomuto ditribučnímu seznamu, je rozeslána poštovním systémem každému z tohoto seznamu. Dalšími rozšiřujícími službami jsou zasílání kopií, určování priority zprávy, ochrana (kódování) zprávy, alternativní příjemce apod.

Stejně jako v klasické i v elektronické poště se zpráva skládá ze dvou hlavních částí, a to **obálky** (*envelope*) a **těla zprávy** (*message body*). Obálka obsahuje informace nutné pro přenos zprávy a její interpretaci (cílová adresa, priorita, stupeň ochrany, typ požadovaného terminálu). Tělo zprávy potom obsahuje blíže nespecifikovaná data (text, grafika, zvuk, popř. jejich kombinaci). Pro určení, o jaká konkrétní data se ve zprávě jedná, slouží standard MIME.

Schéma poštovního systému

Standard MOTIS rozděluje celý poštovní systém na dvě části. **Uživatelský agent** (*User Agent*, UA) je program, který poskytuje rozhraní mezi uživatelem a poštovním systémem. UA umožňuje uživateli sestavit, poslat a obdržet zprávu. **Agent přenosu zpráv** (*Message Tranfer Agent*, MTA) přijímá zprávu od uživatelského agenta. Jde v podstatě o elektronický poštovní úřad, který zajistí ve spolupráci s dalšími poštovními úřady (MTA) doručení zprávy koncovému příjemci. V současné době je obvyklé, že UA i MTA jsou provozovány na jednom počítači. V budoucnosti by mělo být běžnější, že UA poběží na osobním počítači uživatele doma nebo v zaměstnání a MTA poběží naślediskovém počítači v běžném poštovním úřadě, telefonní společnosti apod. Schéma struktury poštovního systému ukazuje obrázek 28. Personální



Obrázek 28: Poštovní systém

počítače mají omezenou diskovou kapacitu a proto není možné všechny zprávy uložit zde. MTA proto udržuje elektronické poštovní schránky všech uživatelů v oblasti nazývané *Message Store*. Uživatel potom pomocí UA zpracovává zprávy z tohoto prostoru.

Jestliže UA a MTA pracují na oddělených počítačích, je mezi nimi samozřejmě nutná vzájemná komunikace (protokol P3), která by se měla stát předmětem standardu. Podobně je standardizovaná organizací OSI komunikace mezi UA a MS (protokol P7) a komunikace mezi jednotlivými MTA (protokol P1). Protokol P1 je nutný z toho důvodu, že celý **systém přenosu zpráv** (*Message Tranfer System* se skládá z několika (libovolně mnoha) MTA, které mohou být provozovány různými organizacemi a na různých základech. Poslední z kolekce těchto protokolů je (virtuální) protokol P2 mezi UA. Tento protokol je definován v podstatě obálkou a tělem zprávy. Protokol P2 je znám také pod názvem IPM (*Interpersonal messaging*).

9.1.2 Diskusní kluby, elektronické časopisy

Myšlenka diskusních klubů je vlastně přirozeným rozšířením metody doručování zpráv pomocí elektronické pošty. Diskusní kluby umožňují lidem se společnými zájmy vzájemně vyměňovat informace, řešit společně problémy apod. Zaměření diskusních klubů jsou velmi rozdílná - počínaje specialisty na počítačové sítě přes jaderné fyziky až po zahrádkáře.

Diskusní klub je pouze adresa poštovní schránky, do které uživatelé zasílají všechny příspěvky na dané téma (ať už se jedná o otázky nebo řešení problémů). Všechny došlé příspěvky (u nemoderovaných klubů) nebo ty, které zvolí moderátor klubu (v případě moderovaných klubů) jsou potom automaticky rozeslány všem uživatelů, kteří se do tohoto klubu přihlásili.

Všechny činnosti spojené s řízení diskusního klubu jsou vykonávány speciálním programem (*list server*). Uživatel samozřejmě komunikuje s tímto programem pomocí elektronické pošty. Jednotlivé příkazy jsou obsaženy v těle zprávy a program se podle nich řídí. Takto je možno se do klubu přihlásit (a poté automaticky dostávat všechny příspěvky), odhlásit, zvolit způsob zasílání zpráv apod.

Vzhledem k tomu, že systém diskusních klubů přece jenom zatěžuje počítačovou síť rozesíláním příspěvků mnohdy desetitisícům uživatelů, uplatnily se i jiné formy výměny informací. Jednou z těchto forem jsou elektronické časopisy (*News*). Aby se odstranil problém s přenosem jednotlivých zpráv, jsou příspěvky udržovány pouze na jednom počítači, uživatel je nedostává do své poštovní schránky. Pokud si chce přečíst došlé příspěvky, udělá to vzdáleně na tomto počítači a k sobě přenáší pouze ty zprávy, které zrovna čte. Samozřejmě existuje i možnost zaslat do News příspěvek nebo odpověď.

9.1.3 Vzdálené přihlašování

Dnešní počítačový svět není homogenní. Byl, je a zřejmě vždy bude obydlen výpočetními systémy, které vycházejí z odlišných filosofií, používají odlišné přístupy a metody, technologie a techniky, a v důsledku toho nejsou nikdy zcela kompatibilní. Praktická implementace vzdáleného přihlašování *remote login* a vzdálených terminálových relací *remote terminal session*, naráží tudíž na nemalé problémy. Podívejme se nejprve, v čem je podstata těchto problémů, a pak si ukažme, jaké jsou možnosti jejich řešení.

Největší rozdíly mezi jednotlivými terminály mohou být již v samotné koncepci terminálu a „míře inteligence", kterou je vybaven. Z tohoto pohledu se terminály obvykle rozdělují do tří skupin:

- **znakové (řádkové) terminály** (*scroll mode terminals*)
- **stránkové terminály** (*page mode terminals*)
- **formulářové terminály** (*form mode terminals*)

Znakový terminál je nejjednodušším typem terminálu a je charakteristický tím, že neumí pohybovat kurzorem nahoru a dolů po obrazovce. Jeho displej se tak chová obdobně jako tiskárna - každý znak je vytisknut (zobrazen) tam, kde se právě nachází kurzor, a ten je následně posunut doprava na následující pozici. Pokud by přitom kurzor přešel přes pravý okraj řádku, je přesunut na začátek řádku nového. Pokud by pro tuto novou řádek již nebylo na displeji místo, jsou všechny právě zobrazovaného řádku odrolovány (anglicky: scrolled) směrem nahoru. Znakový terminál tedy neumožňuje ,,vrátit" se zpět (posunout kurzor) na dříve zobrazený řádek (ale obvykle umožňuje pohybovat kurzorem zpět v rámci daného řádku). Pro ovládání znakového terminálu jsou pak zapotřebí příkazy typu: zobraz znak (na pozici kurzoru, s následným posunutím kurzoru na další pozici), přejdi na nový řádek, případně: vrať kurzor o jednu pozici zpět (v rámci daného řádku), vymaž znaky od kurzoru do konce řádku apod.

Při určitém zjednodušení si lze představit, že znakový terminál má jen jeden jediný řádek (zatímco na displeji je zobrazováno ještě několik ,,předchozích" řádků, které ale již nejsou jinak přístupné).

Naproti tomu u **stránkového terminálu** si můžeme představit, že jeho displej reprezentuje dvojrozměrné pole o n řádcích a m sloupcích a kurzor lze nastavit na libovolnou pozici v tomto dvojrozměrném poli. Díky této možnosti je pak možné pohybovat kurzorem všemi směry a na rozdíl od znakového terminálu je již možné implementovat na stránkovém terminálu například celoobrazovkové editory a další aplikace, které pohyb kurzoru všemi směry vyžadují. Z pohledu ovládání pak přibývají u stránkového terminálu nezbytné příkazy pro nastavování pozice kurzoru.

Formulářový terminál je ještě ,,chytřejším" typem terminálu. U něj si můžeme představit, že místo jednoho dvojrozměrného pole se jeho displej dokáže chovat jako několik (menších) jedno a dvojrozměrných polí. Ty pak můžeme interpretovat jako navzájem disjunktní ,,políčka" formuláře, se kterými se pracuje nezávisle na sobě (tj. která se vyplňují samostatně). Formulářový terminál pak obvykle umožňuje definovat strukturu ,,formuláře": stanovit počet, velikost, dimenzi a druh jednotlivých polí, způsob jejich zobrazení na displeji (včetně doplnění vysvětlujícím textem) atd.

I v rámci jedné a téže kategorie (např. znakových) terminálů se však jednotlivé druhy terminálů mohou výrazně lišit, a to ve dvou základních směrech:

- v konkrétních parametrech a dílčích schopnostech. Rozdílný může být například počet znakových pozic na řádku, počet zobrazovaných řádek, schopnost používat alternativní sady znaků a různé efekty při jejich zobrazování (podtržený text, tučné písmo, písmo dvojnásobné výšky, šířky apod.)
- v konkrétním způsobu ovládání. Některé terminály například očekávají, že konec řádku bude vyznačen dvojicí znaků CR a LF, zatímco jiné terminály znaky očekávají pouze znak CR, a ještě jiné pak pouze znak LF. Odlišné bývají i nejrůznější další řídící znaky, například znak nastavení kurzoru na zadanou pozici (u stránkových terminálů) či jeho posun na předchozí pozici, i samotný efekt jimi vyvolaných akcí (zda se např. při posunu kurzoru o jednu pozici zpět žádný znak nemaže, či naopak ano).

Virtuální terminál

Ve víceúlohovém prostředí na počítači vybaveném více terminály nemohou jednotlivé úlohy předem vědět, se kterým terminálem budou v rámci terminálových relací skutečně komunikovat. Konkrétní typ všech terminálů daného počítače sice může být předem znám, ale připustíme-li možnost vzdálených terminálových relací, uskutečňovaných prostřednictvím počítačové sítě, není již možné předpokládat prakticky nic.

Jednotlivé aplikace, které v rámci terminálových relací pracují s terminály, je však nutně musí umět ovládat. Musí vědět, co všechno terminál ,,umí", jaké znaky je schopen přijímat a zobrazovat, s jakými řídícími znaky pracuje, jak mají být zakončovány jednotlivé řádky, jaké používá řídící sekvence apod.

Každá aplikace si samozřejmě může předepsat, že potřebuje pracovat jen s terminálem určitého typu a tomu se plně přizpůsobit. To je ale příliš omezující vůči uživatelům, kteří disponují jinými terminály. Stejně tak není dost dobře možné, aby se jednotlivé úlohy přizpůsobovaly určitému konkrétnímu repertoáru terminálů - vždy se totiž najde někdo, kdo bude chtít používat ještě jiný terminál. Navíc neustále vznikají nové typy terminálů s novými a dokonalejšími vlastnostmi.

Vzájemné přizpůsobení všech aplikací všem možným terminálům tedy není dost dobře možné. V podstatě jedinou schůdnou alternativou je vytvořit jednotný mezistupeň a přizpůsobení typu ,,každý s každým" nahradit mnohem schůdnější variantou, při které se všechny aplikace přizpůsobují jednotnému mezistupni, kterému se na druhé straně přizpůsobují i všechny konkrétní terminály.

Z pohledu aplikací jde vlastně o terminál, který se vždy ,,tváří" stejně - tedy který má vždy stejné schopnosti, stejné vlastnosti a stejný způsob ovládání (rozhraní). Ve skutečnosti však nejde o fyzické zařízení, ale jen o určité konvence (formát dat a příkazů), či o objekt programové povahy (datovou strukturu), který skutečný terminál pouze zastupuje.

Parametrický model virtuálního terminálu

Hlavním problémem právě naznačeného mechanismu je nalezení takové konkrétní formy (modelu) virtuálního terminálu, která by dokázala „pokrýt" co nejširší spektrum různých terminálů, ale současně umožňovala co nejlépe využít jejich nejrůznějších schopností. Tedy takového modelu, který by neomezoval „individualitu" jednotlivých terminálů více, než je nezbytně nutné.

Nejjednodušším modelem virtuálního terminálu je tzv. parametrický model. Zde si můžeme představit, že virtuální terminál je reprezentován tabulkou, která má pevně daný počet položek a pevně daný jejich význam. Tyto položky přitom mohou obsahovat údaj o tom, kolik znakových řádků má displej terminálu, kolik znakových pozic je na jednom řádku, zda terminál očekává zakončení každé řádky dvojicí CR a LF apod. Dále zde mohou být příkazy (escape sekvence) pro ovládání nejrůznější efektů a možností (podtrhování, tučné písmo apod.), nastavování kurzoru (u stránkových terminálů) atd.

Vlastní aplikace se pak přizpůsobuje společnému mezistupni (virtuálnímu terminálu) tím, že skutečný terminál ovládá pouze prostřednictvím této tabulky. Jestliže například potřebuje nastavit kurzor na určitou pozici, najde si na příslušném místě v tabulce konkrétní řídící příkaz a ten pak zadá terminálu. Konkrétní terminál se zase přizpůsobuje virtuálnímu terminálu tak, že jeho předem definované položky naplní konkrétními údaji.

Právě naznačený mechanismus se ve své nejobecnější podobě prosadil v BSD Unixu. Zde je výše naznačených tabulek více - po jedné pro každý druh terminálu, který daný operační systém zná - a jsou všechny sdruženy do jedné databáze, označované jako `termcap` (zkratka od: terminal capabilities). Aplikace, která chce korektním způsobem pracovat s terminálem, si nejprve musí zjistit, o jaký konkrétní druh terminálu se jedná. Pak si z databáze termcap „přečte" odpovídající tabulku, a příslušný terminál ovládá prostřednictvím příkazů (escape sekvencí), které v této tabulce najde. Správce systému přitom může databázi termcap doplňovat a tím rozšiřovat repertoár dostupných terminálů.

Objektový model virtuálního terminálu

Parametrický model je použitelný pro znakové, a do značné míry i pro stránkové terminály. Pro formulářové terminály je ovšem zapotřebí poněkud obecnější model, kterým je model objektový. Ten si můžeme představit opět jako určitou tabulku, definující schopnosti konkrétního terminálu - v případě objektového modelu je ale tato tabulka označována jako profil. Kromě profilu ovšem tvoří virtuální terminál i obecnější datová struktura (objekt), která funguje jako vyrovnávací paměť mezi aplikací a terminálem. Aplikace, která chce něco zobrazit na displeji terminálu, jednoduše zapíše zprávu do zmíněné datové struktury a o další se již nestará. Je pak na skutečném terminálu (resp. jeho ovladači), aby obsah této datové struktury průběžně zobrazoval.

Zmíněná datová struktura přitom zdaleka nemusí být jen dvojrozměrné pole, reprezentující znakové pozice skutečného displeje. Například u formulářového terminálu to mohou být jednotlivá dílčí pole, představující části formuláře, a jiná část datové struktury zase může definovat, jak mají být jednotlivá pole „poskládána" na skutečném displeji.

Tímto mnohem obecnějším způsobem jsou pak řešeny virtuální terminály v rámci referenčního modelu ISO/OSI.

9.1.4 Adresářové a vyhledávací služby

Každý síťový systém spravuje celou řadu základních informací. K nim patří uživatelská jména, hesla, přístupová práva apod. Nejen rozdílné síťové operační systémy, ale i celá řada různých uživatelských programů jako např. elektronická pošta, terminálová emulace a další vyžadují samostatný systém správy uživatelů. Následkem toho jsou kromě jiného vysoké požadavky na správu systému jako celku, které kladou extrémně vysoké nároky na personál a čas.

Vyjděme z dnes běžné situace. Uživatel se přihlásí do lokální počítačové sítě pomocí svého uživatelského jména a hesla. Poté např. hodlá prohlédnou došlou počtu. Poštovní systém ho opět nutí zadat uživatelské jméno a heslo. Stejná situace se opakuje ve chvíli terminálové emulace pro přístup ke středis-kovému počítači.

Několikanásobná identifikace může být pro uživatele sice snesitelná, avšak na úrovni správce sítě se celá věc stává extrémně složitou a nepřehlednou.

Z výše popsaného téměř zákonitě vyplývá potřeba jednotné uživatelské databáze. Ke konci roku 1988 bylo předloženo doporučení CCITT X.500 – **Distribuovaná adresářová služba** (*Directory Service*). Do úvahy byla vzata celá řada požadavků, které bylo třeba splnit:

- jednotný adresář jmen a objektů
- optimální struktura jmen pro uživatele
- akltualizace záznamů
- distribuce kopií a shodnost databáze
- přehlednost datového souboru

Podrobnější informace o standardu X.500 najde čtenář v kapitole 11 na staně 71.

9.2 Přenosová média

9.2.1 Pevné spoje (*non-switched circuit, direct circuit*)

Jde o spoj, který existuje trvale (tj. i v době, kdy není případně využíván). Zřízení takového okruhu obvykle znamená, že v rámci telefonní ústředny se napevno propojí vhodné, již existující přenosové cesty (resp. na nich vytvořené okruhy).

Pevný okruh obchází přepojovací zařízení v ústřednách, která jsou obvykle zdrojem nejrůznějších poruch a rušení komutovaných okruhů (viz dále), a proto může dosahovat vyšších přenosových rychlostí. Pro rozlehlé počítačové sítě se dnes ve světě používají pronajaté pevné okruhy nejčastěji s přenosovou rychlostí 64 kbit/s.

9.2.2 Komutované spoje (*switched circuit, dial-up circuit*)

Klasický příkladem je telefonní síť. Telefonní přístroj je připojen na telefonní ústřednu pomocí dvojice kabelů. Chceme-li se spojit s jiným účastníkem, který je stejně připojen k telefonní ústředně, musí dojít uvnitř ústředny k dočasnému propojení obou vedení tak, aby mezi oběma účastníky vznikl potřebný telefonní okruh. Po ukončení hovoru pak tento dočasný okruh zaniká a v rámci ústředny se obě přenosové cesty mohou rozpojit.

Komutovaný okruh je tedy spojení mezi dvěma uzly, které neexistuje trvale v čase, ale je pouze dočasně zřizováno až v případě skutečné potřeby.

9.2.3 Mikrovlnné spoje

Přívlastkem mikrovlnné (microwave) se označují elektromagnetické vlny o extremně krátké vlnové délce, resp. vysoké frekvenci. V praxi se používají fekvence od 1 do 12GHz. Vlny o takto vysoké fekvenci již lze vhodnými parabolickými vysílacími anténami soustředit do úzkého paprsku a ten nasměrovat na přijí-mající anténu. Vzhledem k přímočarému šíření soustředěného paprsku elektromagnetických vln je dosah mikrovlnných spojů omezen na přímou videtelnost vysílače a přijímače. Dosah lze uměle prodlužovat umístěním vysílacích a přijímacích antén na co nejvyšší věže. Dosažitelná přenosová rychlost na mik-rovlnných spojích je závislá na použitém frekvenčním pásmu a možnostech vysílače a přijímače (může dosahovat až 10Mbit/s). Mikrovlnné spoje se používají obvykle v městských aglomeracích, v místech kde neexistují vhodné drátové přenosové cesty a instalace nepřipadá v úvahu (např. v historické části města).

9.2.4 Troposferické spoje

Přímá viditelnost mezi vysílačem a přijímačem je značně omezujícím faktorem. Většího dosahu lze dosáhnout tak, že necháme úzce směrovaný paprsek odrazit od horních vrstev troposféry ve výšce cca 16km. Nevýhodou tzv. troposferických spojů je nutnost používat velký výkon vysílače (v řádu desítek kW), přičemž přijímaný signál bývá velice slabý (v řádu nW).

9.2.5 Družicové spoje

Dnes se pro družicové spoje (*satelite links*) používají převážně tzv. geostacionární družice. Pohybují se ve výšce přibližně 36000km nad Zemí, kde je doba jejich obletu kolem Země rovna době otočení zeměkoule kolem osy. To znamená, že se tyto družice vždy nacházejí nad stejným bodem zemského povrchu (většinou se umísťují nad rovník). Družice mohou být dvou typů:

- **pasivní družice** – pouze odrážejí signál přicházející z pozemských vysílačů,
- **aktivní družice** – přijmou signál z pozemského vysílače, zesílí ho, popř. převedou do jiného frekvenčního pásma a vyšlou ho zpět směrem k Zemi.

Pomocí družicových spojů lze vytvořit přenosové kanály a okruhy širokého spektra přenosových rychlostí (desítek až stovek Mbit/s), které se svými vlastnostmi plně vyrovnají pozemním kabelovým spojům a v mnohém je i předčí.

Nevýhodou družicových spojů je zpoždění, které vzniká při přenosu signálu na družici a zpět na Zemi. Ze Země na geostacionární družici a zpět dorazí signál za přibližně 250 až 300ms. To znamená, že odpověď na odeslaný paket může zpět k vysílači dorazit až za cca 500 až 600ms. To je pro vetšinu protokolů velmi dlouhá doba, kterou interpretují jako ztrátu vyslaného paketu a opakují jeho vyslání. Z toho vyplývá nutnost úprav protokolů tak, aby toto zpoždění neinterpretovaly jako chybné.

Družicové spoje lze velice úspěšně využít k šíření signálu od jednoho zdroje k velkému množství příjímačů, ale i naopak.

9.2.6 Privátní síť

Privátní rozlehlé sítě jsou typem sítí, se kterými se můžeme setkat nejčastěji. Jsou to sítě vybudované jednotlivými firmami a institucemi. Jedním z důvodů budování privátní sítě je skutečnost, že objem přenášených dat je již tak značný, že poplatky za přenos po veřejné datové síti (viz 9.2.7) by dosáhly neúnosné výšky. Jiným důvodem může být situace, kdy je třeba zajistit služby na vyšších přenosových rychlostech, než poskytuje veřejná datová síť. Zřizovatel privátní sítě spolu s dalšími uživateli využívajícími tuto síť zajišťuje provozní část celé sítě. Může rozhodovat o tom, kdo síť používá a v jakém rozsahu.

9.2.7 Veřejná datová síť

Veřejné datové sítě jsou sítě, které jsou obvykle rozprostřeny po celém uzemí daného státu a jsou anologií veřejných telefonních sítí. Uživatelem veřejné sítě se může stát kdokoli, kdo požádá o zřízení datového napojení. Vlastníkem, provozovatelem a správcem bývá buď příslušná státní instituce nebo soukromá telefonní společnost. Uživatel platí pouze poplatky za zřízení linky a pevný paušál za přenesenou jednotku informace. Veřejná datová síť je tedy určena uživatelům, pro které by zřízení potřebné datové sítě spolu s přenosovými zařízeními a následnou správou a údržbou bylo nepřiměřeně nákladné (vzhledem k nízkému využití techto prostředků).

V současné době je na území České republiky provozována veřejná datová síť (VDS) firmy EuroTel, která získala pětiletou výhradní licenci na poskytování těchto služeb u nás. VDS EuroTel pracuje na principu přepojování paketů (viz dále).

K VDS EuroTel je možné se připojit v zásadě dvěma způsoby. Jednak přímým připojením (pevný datový okruh) k nejbližšímu přístupovému bodu sítě. Nebo prostřednictvím veřejné telefoní sítě na veřejný (přístupný všem účastníkům) nebo dedikovaný (pro jednoho účastníka) port VDS EuroTel.

Základní druhy spojení v VDS EuroTel jsou:

Komutované virtuální spojení (*SVC - Switched Virtual Curcuit*),
které se vytváří na základě žádosti koncového zařízení pouze na dobu výměny dat.

Pevné virtuální spojení (*PVC - Pernament Virtual Curcuit*),
které existuje trvale mezi dvěma koncovými zařízeními připojenými k VDS. Toto propojení typu bod-bod je analogií pronajatého okruhu.

Na jediném fyzickém vedení může v jednom okamžiku existovat více komutovaných i pevných virtuálních spojení současně.

Jednou ze služeb VDS je možnost vybudování virtuální privátní sítě. Pomocí prostředků VDS má uživatel možnost vytvořit si síť, kterou má možnost sám řídit a spravovat.

9.2.8 Digitální sítě

Počátky digitálního přenosu dat se nacházejí v roce 1837, kdy pan Samuel Morse představil světu svůj telegraf. Teprve později (v roce 1876) se objevil na světě analogový přístroj na přenos informací - telefon Alexandra Grahama Bella. Svoji časovou ztrátu však analogový přenos okamžitě srovnal a teprve poslední doba ukázala možnosti digitálního přenosu zejména v oblasti počítačových sítí.

Jaké výhody shledáváme u digitálního přenosu oproti analogovému? Digitální přenosové cesty dnes umožňují dosahovat mnohem vyšších přenosových rychlostí, digitální signál se snáze zpracovává než signál analogový a příznivější jsou i ekonomické ukazatele. Z tohoto důvodu se například všechny telefonní ústředny přebudovávají na digitální.

Přestože se stále více přenášejí čistě digitální data, nelze předpokládat, že by měl vymizet ze světa klasický analogový telefon, nehledě na rozvíjející se používání telekonferencí, videopřenosů prostřednictvím počítačových sítí a podobně. Stále tedy bude aktuální otázka, jak převést analogový signál na digitální a naopak. Jaká je podstata přenosu analogového signálu (např. telefonní hovor) po digitálních přenosových cestách?

Codec

Zařízení, které převádí analogový (např. telefonní) signál do číslicového tvaru, se označuje **codec** (zkratka pro **CO**der/**DEC**oder). Toto zařízení pracuje na opačném principu než modem: na základě analogového signálu vytváří číslicová data. Codec pravidelně měří (vzorkuje) stav analogového signálu a jeho okamžitou hodnotu aproximuje jednou ze 128 nebo 256 úrovní. To jsou hodnoty, které je schopno zařízení odlišit a které jsou vyjádřeny ve formě sedmi- nebo osmibitového čísla. Důležitou otázkou při zavádění digitálních přenosů je, jak velká je minimální přenosová kapacita kanálu, který má přenášet současně například N telefonních hovorů. Vzhledem k tomu, že běžný telefonní signál je chápán jako signál o šířce pásma 4kHz, stačí jej vzorkovat 8000 krát za sekundu. Codec tedy musí kódovat např. $8000 \cdot 8$ bitů, což je 64kb za sekundu. To je také minimální přenosová rychlost kanálu potřebná pro jeden telefonní hovor.

T spoje

Vzhledem k růstu používání digitálních přenosů bylo nutno i v této oblasti zavést standardy zajišťující kompatibilitu spolupracujících komponent. Jak ale bývá zvykem, vzniklo několik standardů, které jsou mezi sebou vzájemně neslučitelné.

Prvním standardem (zavedeným firmou AT&T jako majoritním provozovatelem telefonních sítí v USA) používaným k propojování telefonních ústředen byly tzv. **T spoje**. Jsou to dvoubodové propojovací spoje, které lze využít buď jako jeden kanál nebo rozdělený na podkanály s přenosovou rychlostí 64kbit/s. Standard zavádí pět skupin T spojů s počty hlasových kanálů od 24 do 4032 a z toho vyplývající přenosovou rychlostí od $1,5$Mbit/s do $0,2$Gbit/s. Vybereme nejfrekventovanější spoj **T1** a vysvětlíme podstatu přenosu. Spoje T1 mají 24 hlasových kanálů, které jsou přepínány na vstup jednoho codec zařízení. Tento codec kóduje hodnotu signálu a generuje 8bitové číslo pro každý kanál (7 datových bitů a 1 řídící bit). Přepínání vstupů je pravidelné a vzniká tak 24 osmibitových položek doplněných jedním řídícím bitem, což tvoří **rámec** o velikosti 193 bitů. Za sekundu se vygeneruje 8000 rámců, v součtu $1,554$Mbit/s.

Vzhledem k tomu, že přenosová kapacita jednoho T spoje je značně vysoká (naše ukázka popisovala nejslabší variantu T spoje), jsou tyto spoje pronajímány často po podkanálech. V Evropě je zaveden standard organizace CCITT zvaný **evropský T1** spoj nebo **E1** spoj, u kterého codec generuje osmibitové číslo, předpokládá 32 kanálů (30 hlasových a 2 řídící) se vzorkováním opět 8000 za sekundu. Přenosová rychlost tohoto spoje vychází na $8000 \cdot 32 \cdot 8 = 2,048$Mbit/s.

ISDN

Využití počítačové sítě pro přenos „nepočítačových" dat však nekončí u přenosu telefonního signálu. Klasické měření přenosových spojů na počet hlasových kanálů se ukázalo být nedostačující. Uživatelé dnes požadují nejen vyšší kvalitu hlasových přenosů, než pro jaké je stávající telefonní síť koncipována, ale také zavádění nových služeb od zpětné vazby na volající osobu přes zlepšování faxových služeb až po možnost normalizovaného připojení počítače přímo na veřejnou přenosovu síť.

Výsledkem tlaků zákazníků je snaha o přeměnu celé nynější analogové telefonní sítě na síť digitální (což je pozorovatelné i v našich podmínkách), která by integrovala veškeré přenosové služby. Nový jednotný systém, ke kterému by měl tento vývoj směřovat, se nazývá **ISDN** (**I**ntegrated **S**ervices **D**igital **N**etwork nebo-li Digitální telekomunikační síť integrovaných služeb).

Standard ISDN (první doporučení k ISDN byla vydána v roce 1984) předpokládá komplexní nabídku přenosových kanálů, pro které zavádí následující označení.

A	analogový telefonní kanál s šířkou pásma 4 kHz
B	digitální kanál s přenosovu rychlostí 64 kbit/s pro přenos hlasu nebo dat
C	digitální kanál s přenosovou rychlostí 8 nebo 16 kbit/s
D	digitální kanál pro služební účely s přenosovou rychlostí 16 nebo 64 kbit/s
E	digitální kanál pro interní potřeby ISDN s přenosovou rychlostí 64 kbit/s
H	digitální kanál s přenosovou rychlostí 384, 1536 nebo 1920 kbit/s

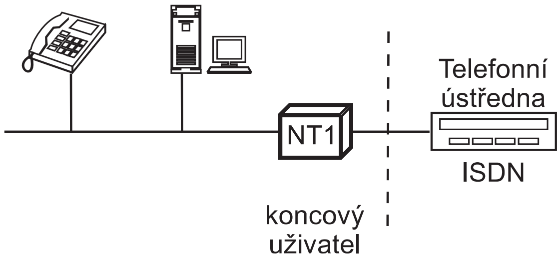
Přitom se nepředpokládá libovolná kombinace těchto kanálů, ale jsou standardizovány tři kombinace:

basic rate	2 kanály B a 1 kanál D
primary rate	23xB + 1xD v USA a 30xB + 1xD v Evropě
hybrid	1xA + 1xC

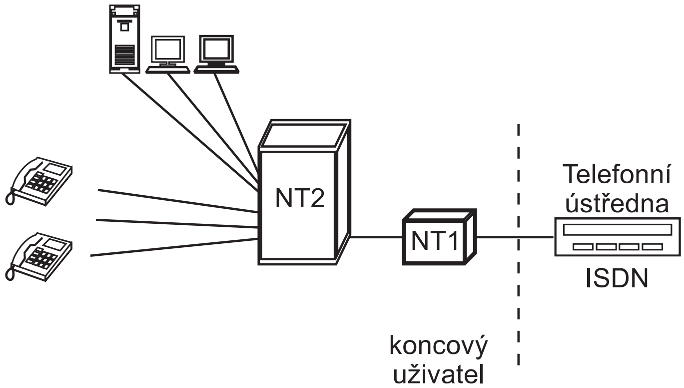
Velmi zajímavé je první řešení (basic rate), které je zamýšleno jako náhrada stávajících účastnických přípojek telefonu do domácností a kanceláří. Každý kanál B dokáže přenášet rychlostí 64kbit/s telefonní hovor nebo data, veškerý režijní přenos je realizován kanálem D. Tento přístup vede k výraznému zkvalitnění služeb (dva kanály dovolí provozovat progresivní služby paralelního přenosu dat i hlasu) - současně můžete provozovat připojení do počítačové sítě i telefonovat.

Zajímavá je také otázka řešení přenosové cesty pro 192kbit/s (celkový součet rychlostí basic rate) pro jednoho koncového uživatele k telefonní ústředně. Pro tyto účely stačí využít stávající účastnickou přípojku pro analogový přenos, která dokáže na vzdálenosti do 8km přenášet rychlostí až 2Mbit/s. Přípojka je dělena metodou časového multiplexu mezi všechny tři kanály B a D. Na konci přípojky je umístěno ukončující zařízení (NT1 - Network Terminantor), které vytváří rozhraní ISDN s možností připojit až 8 zařízení (obrázek 29).

Pro organizace s potřebou většího počtu kanálů je určena přípojka Primary rate, u které se předpokládá připojení vlastní digitální ústředny IDSN, ke které je možné připojovat větší počet zařízení s ISDN rozhraním. Takovou ústřednu označujeme NT2 podle toho, že se připojuje za NT1 směrem k uživateli (obrázek 30).



Obrázek 29: Přípojka ISDN - jednoduchá verze



Obrázek 30: Přípojka ISDN - pobočková ústředna

9.3 Technika přenosu

Pro vlastní transport dat může každá síť použít několika principů pro vysílání, multiplexování a přepojování signálu mezi zdrojem a příjemcem jednoho spojení (například relace). CCITT používá pro techniky přenosu termín přenosový režim (*transfer mode*).

9.3.1 Přepojování okruhů

Přepojování okruhů (*circuit switching*) lze nejlépe přirovnat k mechanismu činnosti klasické veřejné analogové telefonní sítě. Také zde se propojují přenosové cesty tak, aby mezi dvěma komunikujícími uzly vznikl jediný přenosový okruh postupným přepojováním úseků spojů mezi ústřednami. V analogovém prostředí je sdílení média realizováno frekvenčním multiplexováním (každé spojení má k dispozici část šířky pásma), v digitálním prostředí se používá časové multiplexování. Jeho základem je vysílání informací jedné relace jako posloupnost bitů rozložené na časové díly, které se s jistou frekvencí zasílají. Časové díly všech relací sdílejících médium se slučují do **rámců**, které se opakují s danou frekvencí. Přitom časové díly okruhu jedné relace mají v rámcích vždy stejnou pozici. O časových dílech, které se nacházejí na stejné pozici ve všech rámcích dále hovoříme jako o **kanálu**.

Přepojování v ústřednách probíhá na základě překladové tabulky vzniklé v okamžiku navazování spojení:

{{vstupní_spoj, časový_díl_vstup}, {výstupní_spoj, časový_díl_výstup}}

Okruh existuje až do doby, kdy je na konci spojení rozpojen. Oba účastníci spojení tak mají celou dobu své vzájemné komunikace k dispozici pouze pro sebe datový okruh (kanál), po kterém si mohou vyměňovat data v podstatě libovolným způsobem. Mohou si posílat libovolně velké bloky dat nebo spojitou posloupnost bytů.

Nevýhoda této metody je v pevně stanoveném schématu sdílení média. V případě komunikujících multimediálních aplikací je totiž nezbytné, aby některá aplikace měla k dispozici přenosovou rychlost například 140Mbit/s (televizní přenos). Kdyby v takovém prostředí komunikovala aplikace vyžadující pouze 1Mbit/s, většinu přidělených časových dílů by nevyužila a tím plýtvala kapacitami spoje.

9.3.2 Multirychlostní přepojování okruhů

Tato metoda odstraňuje poslední uvedený nedostatek přepojování okruhů. Každý okruh má k dispozici několik ($n < 30$) samostatně přepojovaných základních kanálů o kapacitě například 64kbit/s. Poněkud se tím ztíží přepojování, neboť je nutno všechny tyto kanály synchronizovat. Vyvstává zde problém volby kapacity základního signálu tak, aby pro přenos televizního signálu nebylo nutné mít velký počet základních kanálů (a naopak přenosy malých dat neplýtvaly kapacitou základního kanálu.

Existuje také řešení, které vyhrazuje časové díly různých délek pro různé služby. Vznikne tak vlastně několik druhů základních kanálů. Uvedená metoda ovšem neřeší nárazové požadavky na provoz.

9.3.3 Přepojování paketů

Přepojování paketů (*packet switching*) je založeno na rozdělení přenášených data na stejně velké bloky, které se doplní o některé údaje (např. adresu příjemce a odesílatele). Vznikají tak pakety, které se přenášejí sítí jako dále nedělitelné celky. Nevzniká zde žádný datový okruh mezi odesílatelem a příjemcem paketu. Místo toho si jednotlivé uzly sítě předávají pakety mezi sebou, dokud je nedoručí až k adresátovi paketu. Mechanismus přepojování paketů lze realizovat dvěma způsoby: datagramovou službou a virtuálním spojením. Datagramová služba předpokládá, že uzel sítě paket přijme a pro každý zvlášť se rozhodne, kudy ho má poslat dál. Jde o analogii listovní pošty. Výhodou datagramové služby je možnost reakce sítě na okamžité zatížení některých linek a volby různé cesty přenosu pro jednotlivé datagramy (pakety). V důsledku toho datagramová služba nezaručuje pořadí příchodu datagramů k adresátovi, může se totiž stát, že se v důsledku různě dlouhých cest některé datagramy předbíhají. Nevýhodou datagramové služby je režie spojená s rozhodováním o dalším směru přenosu a určováním správného pořadí datagramů.

Na přepojování paketů je založen standard veřejných sítí **X.25**. Rámce se na spoji vymezují bitovými vzorky, detekce vzniku chyby při přenosu rámce jsou detekovány generováním a kontrolou CRC pro rámec. Je rovněž zajištěno opakování požadavku chybně přenesených nebo ztracených rámců protokolem ARQ (Automatic Repeat reQuest). Záhlaví paketů obsahuje prostor pro informace pro multiplexování logických kanálů na úrovni aplikace. Tato vlastnost ovšem způsobuje, že proměnlivost délky paketů vyžaduje netriviální softwarové ošetření vyrovnávacích pamětí a tato režie vylučuje možnost podpory služeb provozovaných v reálném čase. X.25 sítě se ovšem osvědčují při přenosu dat do 2Mbit/s a jsou také k těmto účelům hojně využívány.

9.3.4 Virtuální spoje

Výše popsané nevýhody přepojování paketů se snaží odstranit mechanismus virtuálních spojů (*virtual links*), nazývaných též virtuální okruhy (*virtual circuits*). Virtuální spoj je realizován tak, aby se každý vnitřní uzel sítě nemusel rozhodovat pro každý nový paket znovu o směru přenosu. Zvolí se jedna pevná cesta ještě před vlastním přenosem dat od odesílatele k příjemci. Po celou dobu přenosu se pak používá tato jediná cesta. Celý mechanismus funguje tak, že se nejprve od odesílatele vyšle speciální paket, který „vytyčí" momentálně nejvýhodnější trasu mezi oběma účastníky a tato informace se zanese do každého vnitřního uzlu, který na této cestě leží. Vlastní pakety, které se odesílají takto vytyčeným virtuálním spojem, mají místo adresy svého příjemce označení virtuálního spoje, který vede k příjemci. Virtuální spoje se tedy zřizují až v okamžiku přenosu dat a po jeho ukončení se zase ruší.

9.3.5 ATM (asynchronní přenosový režim)

Moderní způsob přenosu, se kterým se setkáváme zejména u rychlých páteřních sítí nebo u sítí, které přenášejí multimediální data, je asynchronní přenosový režim (**ATM**). Jedná se v podstatě o přepojování paketů s požadavkem na minimalizaci režie s jejich přenosem, což právě způsobuje výrazné zvýšení přenosových rychlostí v porovnání s klasickým přepojováním paketů. Zároveň je vytvořen mezi komunikujícími stranami virtuální kanál. Asynchronní podstata se projevuje tak, že lze vyrovnávat rychlost vysílače a přijímače tzv. prázdnými pakety. Další posun je v zabezpečení. Vzhledem k použití vysoce spolehlivých spojů je ponechána tato starost na aplikaci - spoj neprovádí žádné režie spojené s detekcí chyb.

Pakety (**ATM buňky**) mají pevnou délku (5 slabik záhlaví a 48 slabik dat) s minimálním záhlavím, které pouze označuje příslušnost buňky k virtuálnímu kanálu. Tím odpadají režijní ztráty vznikající při různé délce paketů na vyrovnávacích pamětech ústředen. ATM umožňuje běžně přenášet rychlostí 150 – 600Mbit/s.

Některé vlastnosti ATM sítí prodiskutujeme podrobněji:

pevný formát buněk
Pevný formát ATM buněk je velmi výhodný při průchodu buněk mezilehlými sítěmi s různými rychlostmi. Tehdy není třeba měnit formát (velikost) buňky, jak jsme byli zvyklí u rámců. Navíc některé operace lze řešit na úrovni technických zařízení. Obdobně jako předem omezená velikost FAT v operačním systému MS-DOS může ovšem právě pevná velikost buňky způsobit perspektivně značné nesnáze při rozsahu komunikace v řádu Gbit/s.

statistické multiplexování
Sdílení média vychází z principu, že zprávy většinou nejsou generovány s rovnoměrnou četností. Naopak, existují chvíle rozsáhlé komunikace a nulové komunikace. ATM proto zavrhlo metody statického přidělení kapacity a odvozuje přidělenou kapacitu z průměrných rychlostí emitovaných zpráv. Tento přístup například dovolí uzavřít spojení více zdroji videosignálu.

řízení toku dat
ATM sítě neposkytují žádnou službu pro řízení toku dat a zvládnutí zahlcení, protože by byla velmi drahá. Zahlcení se kontroluje v místě vstupu ověřením, že jsou dodrženy podmínky stanovené v době navázání spojení. Pro uzly s pravidelným provozem lze vyhradit podsíť virtuálních cest.

jednoduchá architektura
Jak bylo řečeno výše, uzly nesledují vznik chyb při přenosu. Pravda je taková, že se sleduje pouze bezchybný přenos záhlaví buněk, obsah buněk samotných se kontroluje až v koncových bodech spojení. Spoléhá se přitom na velmi spolehlivé přenosy po optických spojích.

Vnitřní uzly ATM sítě jsou vybaveny minimální inteligencí, takže například neřeší směrování - to je řešeno předem připravenými tabulkami na úrovni pevných cest. Nedostatek inteligence může být i na škodu - v případě zahlcení dojde s velkou pravděpodobností ke ztrátě dat.

bezpečnost v ATM sítích
Bezpečností se většinou rozumí ochrana systému proti útokům na zranitelná místa. V základním provedení je ATM síť nedůvěryhodná stejně jako síť s přepojováním paketů. Uživatel však vyžaduje vždy integritu zpráv a mnohdy i záruku důvěrnosti komunikace. Musí být detekovatelné rozšíření zprávy, zrušení zprávy nebo její části útočníkem.
Ochrana je realizována zavedením kryptografických mechanismů buď na úrovni fyzické (šifrují se všechna data), nebo na úrovni ATM buněk (šifrují se pouze těla buněk). Navíc je potřeba řešit problém s klíči kryptografických algoritmů. Jednodušší variantu jednoho klíče pro celou ATM síť lze nahradit bezpečnější - každý virtuální kanál používá svůj vlastní klíč.

10 TCP/IP a Internet

V kapitole 1.4.3 jsme popsali model protokolů TCP/IP ve vztahu k referenčnímu modelu ISO OSI. Význam této rodiny protokolů je zejména v tom, že se používá jako *de facto* standard pro komunikaci v Internetu a je veřejně dostupná veškerá dokumentace tohoto standardu. TCP/IP byl od začátku navrhován tak, aby splňoval parametry použitelnosti při propojování otevřených systémů, takže dovoloval propojování počítačů s různými operačními systémy.

Počátky TCP/IP nacházíme na konci 60.let, kdy tyto protokoly byly vyvinuty pro agenturu ARPA ministerstva obrany USA. Na jejich základě vznikla síť ARPANET, která se posléze přerodila do Internetu. Protokoly TCP/IP se prosadily jako standard do BSD Unixu a nakonec do všech UNIXů. Dnes je však najdeme implementovány na všech známých platformách.

10.1 Charakteristika Internetu a TCP/IP

Jak vyplynulo z kapitoly 1.4.3, liší se základní koncepce ISO/OSI a TCP/IP v počtu vrstev. Klíčový rozdíl je však především v koncepci těchto standardů. Jestliže při zrodu referenčního modelu OSI stály spojové organizace, které prosadily představu spojovaných a spolehlivých služeb až do úrovně síťové vrstvy (tedy celé komunikační podsítě), tvůrci protokolů TCP/IP vycházeli naopak z předpokladu stále rostoucí spolehlivosti přenosu a navrhli řešit otázku spolehlivosti až na úrovni transportní vrstvy. Komunikační podsíť tedy nemusí ztrácet čas verifikací, potvrzováním, opětným zasíláním atd, může v ní docházet ke ztrátám. Předpokládá se tedy rychlá komunikační podsíť, ke které se připojují inteligentní uzlové počítače.

Způsob fungování komunikační podsítě je druhým klíčovým rozdílem obou modelů. Zatímco model OSI počítá se spojovaným přenosem virtuálními okruhy, TCP/IP se spokojí s nespojovaným přenosem datagramů. Aplikační vrstva TCP/IP poskytuje pestrou nabídku služeb, která rovněž přispěla k masovému rozšíření těchto protokolů. Mezi nejznámější patří elektronická pošta, přenos souborů, vzdálený terminál, systém jmenných domén, distribuce News a další.

Výsledkem propojování sítí protokoly TCP/IP vzniká vždy složitější soustava sítí, které říkáme internet. Název Internet (velké I) se pak vztahuje k administrativně řízené světové síti. Z hlediska uživatele se Internet jeví jako soustava vzájemně propojených sítí, mezi kterými existují brány (správně směrovače) jako kontaktní místa propojených sítí.

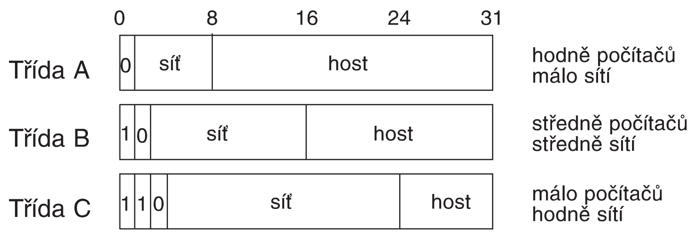
10.2 Adresování v TCP/IP síti

10.2.1 IP adresy

Síťový model TCP/IP vytváří abstrakci sítě na úrovni síťové vrstvy. Tato abstrakce se vyznačuje mimo jiné jednotným formátem adresy. Kromě fyzické adresy (různé délky) přichází TCP/IP s logickou 32bitů dlouhou adresou (**IP adresa**). Tato adresa se skládá ze dvou komponent, a to adresy sítě (*netid*) a adresy uzlu (*hostid*). Tato koncepce přesně vyhovuje pojetí Internetu jako propojených sítí.

Identifikace pomocí IP adresy je jednoznačná. Nemohou tedy existovat dva různé počítače se stejnou IP adresou. IP adresa se však vztahuje vždy k síťovému adaptéru počítače nebo zařízení v síti, což dovoluje, aby byl jeden počítač identifikován více IP adresami. Jestliže má mít každý počítač jedinečnou IP adresu v rámci celého Internetu, musí být adresy přidělovány centrálně jedinou autoritou, což je DDN Network Information Center v USA. Pojetím adresy jako dvousložkové se výrazně zjednodušuje doručení zpráv, neboť tyto se doručují do dané sítě.

IP adresa (32 bitů) se většinou zapisuje tečkovou notací ve formátu čtyř osmibitových čísel (například `158.194.80.20`). V tomto případě je adresou sítě `158.194.80` a adresou počítače v dané síti `20`. Podle toho, jak rozsáhlá je síť, ve které se počítač nachází, dělí se IP adresy do tří skupin (tříd). Jednotlivé třídy se liší počtem bitů vyhrazených pro adresu sítě a adresu počítače (obrázek 31). Z uvedeného obrázku lze odvodit, kolik sítí resp. počítačů připouštějí jednotlivé třídy adres. Některé adresy jsou vyhrazeny pro speciální účely, například pro adresování všech počítačů v dané síti:



Obrázek 31: Třídy IP adres

- nula na pozici *hostid* označuje adresu sítě určenou *netid* (např. 147.228.0.0)
- nula na pozici *netid* označuje adresu počítače *hostid* v aktuální síti (např. `0.0.0.88`)
- nula na pozici *netid* označuje adresu počítače *hostid* v aktuální síti (např. `0.0.0.88`)
- `127.0.0.0` označuje loopback adresu - používá se k testovacím účelům.

Poznámka: Nevýhodou takto pojatého adresování je vazba konkrétní adresy na síť, což se projeví například u přenosných počítačů. Přeneseme–li počítač do jiné počítačové sítě, musíme mu změnit jeho IP adresu a teprve potom bude schopen komunikace.

10.2.2 Symbolická jména

IP adresa jako jednoznačné pojmenování počítače v rámci sítě Internet je pro identifikaci velmi výhodná. Pro uživatele, který rovněž potřebuje adresovat některé zdroje, je číselná notace adresy nepřijatelná. Z toho důvodu byl vytvořen mechanismus dovolující používat k adresování tzv. **symbolických jmen** (například risc.upol.cz).

Přepis symbolického jména na IP adresu je prováděn automaticky obslužným programem. Data tohoto obslužného programu jsou obecně uložena v souboru `hosts` (v UNIXu uloženého v adresáři `/etc/`), kde jsou nová jména zavedena ve tvaru

```
127.0.0.0      loopback
157.194.80.20  alpha.inf.upol.cz alpha
158.194.80.56  claudius.inf.upol.cz claudius
```

První sloupec označuje IP adresu, v ostatních sloupcích jsou název a volitelný alias pro každý nový uzel.

Evidence všech počítačů, které potřebujeme pro komunikaci, v lokálních datových souborech by byla naprosto nemožná. Přes milión počítačů evidovaných na každém lokálním uzlu by zabíralo nezanedbatelný prostor na discích, aktualizace takových databází by pak byla akce prakticky nerealizovatelná.

Pro správu jmen a administrativní správu větších částí sítě jsou vytvářeny tzv. **domény**. Koncepce hierarchicky členěných a samostatně spravovaných domén řeší uvedený problém dokonale. V hierarchii nejvýše stojí tzv. **základní domény** (edu, com, gov, cz, uk, ...). V USA vznikly základní domény podle typu organizace (edu - školská organizace, gov - vládní organizace ...), v dalších statech pak vznikaly základní domény pojmenované zkratkou názvu státu (cz - Česká republika, de - Německo, ...). Další členění jmen subdomén je plně v pravomoci správce základní domény. Pod základní doménu `cz` tak spadá například doména `upol.cz` Univerzity Palackého v Olomouci a v ní je vytvořena doména `inf.upol.cz` spravovaná katedrou matematické informatiky.

V každé doméně existuje počítač `nameserver`, který zajišťuje službu **DNS** - domain name service. Tento nameserver (resp. služba DNS) zná IP adresy a jména všech počítačů ve své doméně. [3] Požadujeme-li připojení k uzlu, jehož jméno zadáme symbolicky, je automaticky zaslán dotaz nadřazenému nameserveru. Jestliže tento nameserver neumí dotaz vyřešit (není to nameserver dotazované domény), postupuje dotaz v hierarchii výše. Aby nedocházelo ke zbytečným často se opakujícím dotazům, každý nameserver má vyrovnávací paměť s často se opakujícími jmény a jejich adresami.

[3] Pro úplnost je nutno dodat, že kromě uvedeného primárního nameserveru existuje mimo doménu a potenciální zdroj stejných chyb ještě záložní (sekundární) nameserver, který je dotazován v případě nedostupnosti primárního.

10.2.3 Přepis IP adresy na fyzické adresy - ARP

IP adresa jako standardizovaný 32bitový identifikátor se liší od obecně různě dlouhé fyzické adresy. Aby spolu mohly počítače komunikovat, musí znát své fyzické (hardwarové) adresy. Pro přepis ze známé IP adresy na hardwarovou adresu slouží protokol ARP (*Address Resolution Protocol*). ARP pracuje na lokální síti následujícím způsobem:

1. Počítač A vyšle žádost *ARP request* s identifikací své hardwarové a IP adresy a cílové IP_B adresy. Žádost je adresována všem uzlům v segmentu.
2. Počítač B, který podle IP_B akceptuje žádost, vytvoří ARP odpověď, ve které sdělí svoji fyzickou adresu. Tuto odpověď adresuje přímo uzlu A.

V případě, že ARP žádost obsahuje IP adresu, která je mimo segment sítě s uzlem A, odpoví na ni směrovač v tomto segmentu. Následně dostává datagramy pro adresáta s IP adresou, které směruje skutečnému počítači.

10.2.4 Získání IP adresy při startu systému - RARP, BootP

Protokoly, které při startu systému přidělují dynamicky nebo staticky IP adresu, se nazývají RARP (Reverse Address Resolution Protocol) nebo BootP (BootPacket). Podstatou získání IP adresy je zaslání požadavku s uvedením své hardwarové adresy. Centrální autorita vlastní překladovou tabulku s položkami HW adresa → IP adresa.

10.3 Směrování v TCP/IP sítích

Pro protokoly TCP/IP je síť tvořena sítěmi vzájemně propojenými pomocí bran (správně směrovačů). Propojení je realizováno tak, že mezi libovolnými dvěma sítěmi existuje aspoň jedna cesta, která může vést přes větší počet jiných bran. Směrování je proces hledání cesty, kterou se má paket posílat, aby došel k cílovému počítači. Sleduje se přitom efektivita takové cesty.

Na směrování se podílejí jak hostitelské počítače, tak (a to hlavně) brány. Cíl je minimalizovat podíl hostitelských počítačů na rozhodování o směrování datagramů a přenechat všechnu práci branám. Existují v podstatě dva druhy směrování, a to přímé a nepřímé. Přímé směrování v terminologii TCP/IP je plně v režii hostitelského uzlu a použije se, jestliže cílový počítač existuje na stejném segmentu sítě jako odesílatel (dá se to poznat podle IP adresy). V tomto případě se zpráva posílá přímo cílovému hostu.

Nepřímé směrování se použije v případě, že cílový uzel zjevně neleží na stejném segmentu jako odesílatel. V tom případě odesílající uzel zašle datagram bráně ležící v jeho segmentu sítě a záležitost tím pro něj končí. V případě, že v jeho segmentu leží více bran, musí se vysílající uzel rozhodnout, která z nich bude schopna nejdříve doručit datagram k cíli. Rozhodnutí je opět přenecháno branám, a to následujícím způsobem. Jestliže brána zjistí, že počítač A používá neoptimální cestu, sice ještě tento datagram pošle, ale zpět uzlu A zasílá zprávu o výhodnější bráně pro tuto komunikaci. Hostitelský počítač tuto informaci zanese do svých směrovacích tabulek a dále používá tuto optimalizovanou cestu.

Směrovací tabulky v branách nebo jednotlivých počítačích mají tvar

IP adresa sítě A	přímé doručení
IP adresa sítě B	IP adresa brány G1
IP adresa sítě C	IP adresa brány G2
DEFAULT	IP adresa brány G

Tato tabulka může být udržována dynamicky (s případnou změnou topologie nebo detekcí chyb se mění automaticky) nebo staticky (manuálním zásahem správce). Předpokladem funkčnosti směrování paketů v síti je pochopitelně dynamická aktualizace směrovacích tabulek. Ta je zajišťována směrovacími protokoly.

V sítích TCP/IP se používají dvě základní třídy směrovacích algoritmů (protokolů): **DVA** (*Distance Vector Algorithms*) a **LSA** (Link State Algorithms).

DVA (Distance Vector Algorithms) je založen na vysílání obsahů směrovacích tabulek jednotlivými směrovači a jejich aktualizaci podle přijatých informací od jiných směrovačů.

Každý router má na začátku ve své tabulce pouze adresy sítí, ke kterým je fyzicky připojen. Tabulka je naplněna trojicemi `(Síť, G, N)`, kde `Síť` je adresa cílové sítě, `G` označuje adresu routeru, na který se mají data posílat, a `N` je počet routerů v této cestě. Každý router zasílá periodicky kopii své směrovací tabulky všem přímo dosažitelným routerům. Při přijetí cizí směrovací tabulky porovná příslušná `N` ve své tabulce s `N+1` v tabulce cizího routeru a jestliže je nějaká nová cesta z cizí tabulky kratší, než cesta podle lokální tabulky, provede aktualizaci své tabulky. Stejně se zachová, najde-li cestu, kterou vůbec nemá zavedenu.

Aby se zabránilo nadměrnému růstu tabulek a umožnila rychlejší aktualizace, je stanovena určitá vzdálenost jako maximální (D_∞). Volbou vhodného čísla D_∞ je dána rychlost aktualizace tabulek při změně topologie a také velikosti tabulek, které se v každém kroku přenášejí.

LSA (Link State Algorithms) je založen na vysílání informací o stavu linek jednotlivými routery. Tak získá každý router představu o topologii, a to nezávisle na jiných. Nad touto topologií pak každý lokálně spočítá optimální směrování.

Každý LSA router tedy sleduje stav linek k sousedním LSA routerům a tento stav vysílá svým sousedům. Při tomto mechanismu komunikace se přenáší konstantní množství zpráv z jednoho routeru, které není závislé na rozsahu sítě.

11 Doporučení řady X.500 - Directory Service

Při práci s rozličnými aplikacemi pod různymi systémy - služby sítě, pošta, terminálová emulace, databáze je vždy vyžadována a ověřována autorizace pracovníka. Setkáváme se tak s několika běžnými kontrolami přihlášení

- do LAN
- do poštovního systému
- na vzdálený počítač

Pro uživatele je sice procedura několikerého přihlášení snesitelná, ale pro správce neúnosná (přestože se totiž jedná o jediného uživatele, musí se informace o něm udržovat několikrát). Návrh vytvořit jednotnou databázi všech práv uživatele v systému je podstatou standardu X.500. V celé řadě doporučení X.500 se hovoří o adresáři v jednotném čísle, čímž je zdůrazněn záměr budovat prostřednictvím jediného univerzálního jmenného prostoru jediný logický adresář složený z mnoha systémů a sloužící mnoha aplikacím. Adresářová služba udržuje vsechny informace o uživatelích, počítačích, aplikacích atd. (jméno, adresa, tel. a fax číslo, práva). Logickým důsledkem komplexnosti je, že toto správu nelze provozovat na jednom počítači. Důvodem je například nebezpečí havárie, zátěž tohoto počítače a linek.

11.1 Systém X.500

Adresářová služba pracuje na principu Client-Server. Je navržena plně distribuovaně, což usnadňuje spolupráci různých systémů.

Operace nad objekty jsou: čtení, srovnání, vyhledání, přerušení vyhledávání, výčet, přidávání, rušení a modifikace objektu.

11.2 Novell Netware v4.x a adresářové služby (Directory Services)

Netware Directory Services (NDS) jsou hlavní změnou oproti předchozím verzím NN. Jsou založeny plně na doporučení X.500. Dřívější verze udržovaly databáze uživatelů (Bindery) na každém serveru odděleně a uživatel se hlásil ke každé obslužné stanici zvlášť. Uživatelé NDS se hlásí do sítě jako celku. Pro přihlášení je vyžadováno jediné heslo, uživatel má k dispozici stejné pracovní prostředí nezávisle na tom, ze které stanice v síti se hlásí.

NDS rozlišuje několik druhů objektů. **Fyzické objekty** jsou například server, svazek, uživatel nebo tiskárna, **logické objekty** například skupina, tisková fronta či profily. V NDS se atributy nazývají vlastnosti. U uživatele jsou evidovány například tyto vlastnosti: jméno, adresa, telefon, příslusnost ke skupině. Je dovoleno vlastnosti přiřazovat několik hodnot (telefon do práce, domů, mobilní telefon). Při vyhledávání uživatele s určitými vlastnostmi mohou být potom prohledávány pouze tyto vlastnosti.

11.2.1 Přístupová práva

Zatímco v předchozích verzích byla práva přidělována pouze na adresář nebo soubor, od verze 4.01 je možné přiřadit práva navíc na práci s objekty. Tato práva nejsou vztažena na objekt jako celek, ale na jednotlivé vlastnosti objektu.

11.2.2 Stromová struktura

Poznámka o nemožnosti jediné centrální evidence nutně vede k myšlence hierarchie objektů vycházející z reálného stavu řízení. Vytváří se stromová struktura objektů. Ve stromu mohou být dva typy uzlů - *kontejnery* a *koncové objekty*. Umístění objektu ve stromu je nazýváno *kontextem objektu*. Kontext zároveň vytváří jedinečné jméno objektu. Kontext neurčuje ani práva ani fyzické umístění objektu.

11.2.3 Správa NDS

Při instalaci je založen uživatel ADMIN s absolutními právy. Toho lze však z organizačních důvodů zrušit, pokud jsou definováni jiní uživatelé s dostatečnými právy. Každý z nových uživatelů s vysokými právy se pak stává administrátorem příslušné části hierarchie stromu objektů. Neexistuje zde uživatel SUPERVISOR s absolutními právy. Může byt založen uživatel, ktery má práva nad určitým podstromem NDS, ale může byt také velice omezen (správa jmen, přístup k tiskárnám, osobní údaje).

Pro integraci starších verzí umožňuje NN 4.01 emulace Bindery. NDS databázi je možno rozdělit na několik částí (*partitions*), které mohou být umístěny na různych serverech. Při instalaci nového serveru je vytvořena nová partition, která se stane podstromem základní NDS. NDS může byt distribuována ve třech typech: Master, Read-Write, Read-Only. Master kopie může byt pouze jedna, zbylé dva typy v libovolném množství.

11.2.4 Časová synchronizace

Existuje jedna základní referenční časová stanice, která - určuje čas v celé síti. Poskytuje jednotný čas pro celou síť, tento čas je nastavován správcem sítě. Referenční časovací stanice nastavuje svůj čas z vnější časové základny. Může to být například čas z rádia. Primární časovací stanice spolu s ostatními primárními a referenčními časovacími stanicemi určuje společný síťový čas. Primární stanice synchronizují svůj čas v souladu s referenčními stanicemi. Sekundární časovací stanice pouze upravuje svůj čas podle primární a tento poskytuje jako informaci do sítě - neúčastní se tvorby síťového času.

11.3 Auditing sítě (Network Auditing)

V předchozích verzích NN byly činnosti spojené s evidencí využívání sítě velice omezené. Nová verze byla navrhována s ohledem na větší možnosti monitorování činnosti v rozsálých sítích. Může být definován *auditor sítě*, ktery může sledovat:

- události týkající se souborů a adresářů (rušení, zakládání, přesuny)
- události týkající se tiskových front
- ukončení činnosti serveru
- události týkající se objektů v NDS
- modifikace přístupovych práv (ne jejich definování, ale sledování jejich přidělování)
- události týkající se uživatelů (přihlášení, odhlášení)

Auditor může takto kontrolovat činnost správce.

A Seznámení s operačním systémem UNIX

Příloha k učebnímu textu o počítačových sítí je určena pro ty čtenáře, kteří se setkávají s operačním systémem UNIX poprvé, obsahuje tedy první uživatelské přiblížení UNIXu člověku, který dosud pracoval pouze v operačním systému MS DOS nebo Windows. Tyto systémy budou také brány ke srovnání podobných vlastností.

Operační systém UNIX je stále jedním z majoritních systémů, které pracují na uzlových počítačích sítě Internet. Velmi často se na školách setkávám s počítači, na kterých pracuje sharewarový kvalitní operační systém - Linux. Po úvodním seznámení s problematikou doporučuji podle potřeby projít některou z doporučených knih ([21], [10], [20]), kde jsou uvedená fakta popsána podrobněji na větším prostoru. Speciálně [20] obsahuje velmi podrobný popis některých tradičních programů, který se uživateli může hodit.

A.1 Historie

Historie UNIXu nevycházela z faktu, že by právě toto měl být jeden z nejužívanějších a nejkvalitnějších operačních systémů, jaké kdy spatřily světlo světa. Prvotní myšlenka vznikla v Telefonních laboratořích firmy Bell a cílem bylo vytvořit interaktivní prostředí pro práci (začátek velmi podobný jako nenápadné narození systému World Wide Web).

Myšlenka pana Ritchieho a Thompsona se brzy ujala a počet instalací systému, který byl záhy přepsán do jazyka C, vzrůstal. Koncem 70. let byl operační systém UNIX přenesen na počítač s rozdílnou architekturou a tehdy se jasně ukázalo, že myšlenka tohoto systému je životaschopná.

Bohužel UNIX ve svém vývoji nesměřoval vždy jedním směrem a dnes se setkáváme s komerčním „standardem" UNIX System V a akademickou verzí UNIXu, nazývanou podle místa vzniku BSD (Berkeley Software Distribution).

Prakticky každá velká hardwarová firma přichází na trh s vlastním klonem UNIXu, takže se setkáváme se systémy *Digital UNIX* firmy DEC, *AIX* firmy IBM, *HP-UX* firmy Hewlett Packard, *SINIX* firmy Siemens, *Solaris* firmy Sun a řadou dalších. Snahy o standardizaci dnes však vedou klíčové firmy ke sjednocení na standardu UNIX System V Release 4. Kromě komerčních distribucí existuje také (vlastně nejnovější) freewarová verze UNIXu - *LINUX*, která pronikla (bez nadsázky) do většiny organizací, které zpočátku experimentovaly a posléze korektně využívají Internet. Výhoda Linuxu je v jeho bezproblémové kompatibilitě se sítí Internet, pro kterou a na které byl vlastně tvořen a upravován.

Pokud se jedná o lokalizaci (počeštění) UNIXů, je možné korektně pracovat česky například v Linuxu, Digital Unixu, AIX.

A.2 Charakteristika operačního systému UNIX

UNIX je víceuživatelský a víceúlohový univerzální operační systém. Dovoluje tedy souběžnou práci více uživatelům, z nichž každý může mít spuštěno více úloh (procesů) najednou. UNIX je systém pracující na mnoha odlišných architekturách od superpočítačů až k PC a vyznačuje se několika základními vlastnostmi:

- poměrně slušně standardizované základní rozhraní
  (kromě klasického řádkového režimu většina unixů podporuje práci v grafickém prostředí)
- jednotný přístup k periferiím (každá periferie se chápe jako soubor)
- součástí UNIXu je kompletní síťové vybavení
- široká paleta aplikačních programů (komerčních i shareware)
- robustnost systému (bezpečnost, stabilita)

A.3 Přístup do systému

V systému je každý uživatel identifikován uživatelským jménem a heslem, které mu přiřadí správce systému - uživatel `root`. Současně také obdrží *domovský adresář* a je zařazen do pracovních skupin. Skupiny seskupují uživatele, kteří mají mít stejná práva na určitou množinu systémových prostředků (soubory, tiskárny, procesory, ...).

Ke komunikaci s UNIXem používá uživatel *terminál*, což může být i PC, na kterém spustíte emulaci terminálu (např. program telnet). Práce uživatele u teminálu je realizována v tzv. *relacích*. Relace je zahájena přihlášením do systému a končí odhlášením. Přihlášení je první operace, kterou terminál oznamuje výzvou

```
login:
```

Po zadání uživatelského jména a správného hesla (nevypisuje se) se ohlásí systém promptem (například `$`), čekajícím na zadání příkazu.

```
login: kopka
kopka's password:
$
```

Ukončení relace je provedeno klávesami Ctrl-D (znak konce souboru), popřípadě příkazy logout nebo exit (podle implementace).

Popsaný terminálový přístup je sice často používaný v textovém režimu, ale je běžné realizovat relaci prostřednictvím grafického *Xterminálu*, tedy grafického okna. K tomuto účelu prakticky každý UNIX umožňuje používat rozhraní X Window s podobným uživatelským komfortem, jako jsme zvyklí v okenních systémech firmy Microsoft.

A.4 Příkazy

Po zobrazení promptu (většinou `$`) je možné zadat příkaz. Obecný formát příkazu je `příkaz [volby] [argumenty]`. Následující příkazy vypisují aktuální datum a čas (příkaz `date`) a uživatele pracující na daném systému (`who`):

```
$ date
Tue Oct 25 21:38:20 NFT 1994
$ who
lumir  pts/0  Oct 25 21:29   (risc.upol.cz)
kopka  pts/1  Oct 25 20:00   (claudius.inf.upol.cz)
```

Systémová nápověda k příkazům je dostupná prostřednictvím manuálových stránek. Tyto stránky (man pages) vypisuje příkaz `man heslo`, nebo v grafickém režimu bývá k dispozici příkaz `xman`. Popis výše uvedeného příkazu `who` získáme

```
$ man who
```

Vzhledem k tomu, že UNIX je systém, který obecně rozlišuje velikost písmen, je nutné i u názvů příkazů a jejich argumentů dodržovat správnou syntaxi.

Může se zdát, že začínáme vysvětlovat nějaký zvláštní systém, který se zřejmě ovládá pouze tajemnými příkazy. Opak je pravdou, pokud jde o slůvko *pouze*. Většina systémů s grafickým rozhraním vyžadují minimální interakci uživatele v textovém režimu. Vzhledem k masovému využití UNIXových uzlů při práci v síti Internet však nelze spoléhat vždy na existenci grafického rozhraní. Potom se znalost základních příkazů stane nezbytnou nutností.

A.5 Soubory v UNIXu

Systém uložení souborů a řazení adresářů do stromové struktury je na první pohled řešen velice podobně jako například v MS-DOS (který se UNIXem nechal inspirovat). V UNIXu existuje kořenový adresář (`root, /`), ze kterého se odvíjí celá adresářová struktura. Vlastnosti systému souborů v UNIXu ukážeme na srovnání se známým systémem souborů v DOSu:

- UNIX nepoužívá pro práci s více fyzickými disky názvů (`C:`, `D:`, `A:`, ...). Místo toho eviduje jednotnou strukturu adresářů vycházející z `/` (kořenový adresář). Na každém fyzickém disku (a na každé partition) je vytvořen samostaný souborový systém a ten je připojen operací `mount` jako jeden adresář do základní hierarchie. Uživatel tedy nerozlišuje mezi diskovými médii a pracuje se systémem souborů jako celkem.
- V UNIXu rozlišujeme tři typy souborů
  - **obyčejný soubor** - pojmenovaná posloupnost slabik. Obyčejné soubory obsahují libovolná data, jejichž využití závisí na programu, který s nimi pracuje.
  - **adresář** - soubor, který obsahuje seznam jmen souborů a adresářů nacházejících se v daném adresáři. Příkaz `ls` vypisuje obsah aktuálního adresáře tak, že vypíše obsah daného souboru adresáře. Každý adresář obsahuje dva soubory - adresáře `.` a `..` se stejným významem jako v MS DOSu.
  - **speciální soubor** - slouží k ovládání periférií (/dev/null, /dev/tty, /dev/lp0)
- Každý soubor má v souborovém systému aspoň jedno jméno. Soubor však může mít více jmen - odkazů (*links*) umístěných v různých adresářích. Pro zakládání odkazů se používá příkaz *ln*.
- Jméno souboru nebo adresáře v UNIXu není omezeno délkou 8 znaků, může být tvořeno většinou aspoň 64 znaků. Tečka nehraje v názvu speciální úlohu, pokud není na první pozici. V názvu souboru se může vyskytnout i více teček. Nevhodné je v názvu souboru používat znaky `+`, `;`, `,`, `*`, `&`, `?`. UNIX rozlišuje velká a malá písmena, na což je potřeba dávat pozor i při pojmenovávání souborů.
- Úplné jméno souboru se udává plnou cestou z kořenového adresáře (např. /home/staff/matej/.profile). Zde je rozdíl pouze v použití opačných lomítek v cestě, než používá DOS.

A.5.1 Příkazy pro práci se soubory a adresáři

- `cp zdroj cíl` - kopírování souborů `zdroj` do `cíl`.
  Cílovým jménem může být soubor nebo adresář, ve druhém případě může *zdroj* označovat více souborů (například hvězdičkovou konvencí). Kopírujeme-li adresář zdroj i s jeho podadresáři, použijeme přepínač `cp -r zdroj cíl`.
- `mv zdroj cíl` - přesun (přejmenování) souborů `zdroj` na `cíl`.
  Parametry obdobné jako u `cp`.
- `rm soubory` - smazání souborů `soubory`.
  Příkaz `rm -r adresář` provádí mazání i v podadresářové struktuře adresáře `adresář`. Na tomto místě je vhodné zmínit se o jedné vlastnosti UNIXu, a to nemožnosti obnovit zrušený soubor. Některé implementace v grafických rozhraních sice používají známý koš, ze kterého jdou soubory obnovit, není to však platné pro UNIX obecně.
- `ln [-s] zdroj cíl` - vytvoří [symbolický] odkaz na zdrojový soubor a tento odkaz vytvoří jméno cíl. Zatímco klasický odkaz lze použít pouze v rámci jednoho souborového systému (ukazuje přímo na i-uzel daného souboru a zvyšuje jeho čítač odkazů), symbolický odkaz lze použít mezi souborovými systémy, protože se uchovává pouze ve formě řetězce.

- `ls -la` - vypíše obsah aktuálního adresáře (více viz `man ls`).

- `more soubor` - zobrazení textového souboru po jednotlivých jeho stranách. Ovládání výpisu je velmi jednoduché, z nejpoužívanějších příkazů uveďme

mezera	vypsat další stránku
Enter	vypsat další řádek
/text	vyhledat nejbližší výskyt řetězce *text*
q	ukončit výpis

A.5.2 Vlastnosti souboru

Každý soubor má následující vlastnosti:

jméno souboru - byla o něm již zmínka;

číslo i-uzlu - soubor je na disku jednoznačně určen tzv, i-uzlem, který specifikuje místo jeho uložení na médiu a další informace;

vlastníka, skupinu a přístupová práva - tomuto atributu se budeme věnovat podrobněji;

časy aktualizace - čas posledního přístupu, poslední aktualizace, poslední změny i-uzlu;

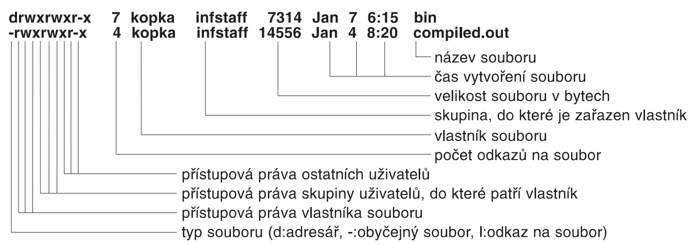
velikost souboru

počet odkazů na soubor

A.5.3 Ochrana souboru

Vlastník souboru je vždy jeden uživatel, kterému daný soubor patří a kterému (mimo jiné) zabírá soubor prostor v systému kvót. Vlastník má také některé speciální možnosti operací s daným souborem.

Pro každý soubor lze nastavit tři druhy přístupových práv (čtení (**r**), zápis (**w**), provedení (**x**)) pro vlastníka, vlastnickou skupinu a ostatní uživatele.



Obrázek 32: Práva k souboru

Výpis vlastnických práv souborů v aktuálním adresáři provádí například příkaz `ls -l` na obrázku 32. První znak ve výpisu označuje typ souboru (`d` adresář, `-` obyčejný soubor), následují tři trojice práv. První trojice pro vlastníka, druhá pro vlastnickou skupinu a třetí trojice pro ostatní uživatele. V každé trojici znamená

znak	**právo pro soubor**	**právo pro adresář**
r	právo číst	soubory v adresáři mohou být vypsány příkazem `ls`, jestliže je nastaveno také právo `x`
w	právo zrušit nebo modifikovat	soubory v adresáři mohou být modifikovány nebo rušeny, je–li nastaveno také právo `x`
x	právo vykonat soubor (binární nebo dávkový)	adresář může být prohledáván (`find`) a zpřístupněn (`cd`)

V případě odepření některého práva danému okruhu uživatelů je na místě tohoto práva `-` (pomlčka).

Práva mohou být kombinována libovolným způsobem, takže například soubor může mít právo `x` ikdyž daný uživatel nemá na tento soubor právo `r`.

Vlastník má většinou nastavena práva všechna (`rwx`), pokud je užitečné mít nastaveno právo `x`. Pouze pro čtení lze soubor zpřístupnit právy `r-x`. Zajímavé právo `--x` se použije v případě, že daná skupina uživatelů má mít právo pracovat se soubory v daném adresáři, ale jen s těmi, jejichž jméno zná.

A.5.4 Změna přístupových práv

Změna práv k souboru se provádí příkazem

```
chmod parametry soubor
```

`parametry` se skládají z

- kategorie uživatelů [**u**(user), **g**(group), **o**(others), **a**(all)],
- operátoru [=, +, −] a
- práva [r, w, x].

Přidání práva `x` všem uživatelům k souboru `ctimne.txt` se stávajícími právy `rw-r--r--` provedeme příkazem

```
$ chmod a+x ctimne.txt
```

Výsledná práva pak budou `rwxr-xr-x`. Podobně příkaz

```
$ chmod g+w o-w data.dta
```

přidá vlastnické skupině a ostatním odbere právo `w` na soubor `data.dta`.

Další způsob zadání `parametrů` je pomocí tří oktalových hodnot (hodnot v osmičkové soustavě). Každá číslice 0–7 znamená bitové nastavení práv v příslušné trojici práv (jedna pro práva vlastníka, druhá pro práva jeho skupiny a třetí pro ostatní).

```
$ chmod 777 soubor1
$ chmod 755 soubor2
```

V předchozí ukázce nastaví první příkaz všem plná práva k souboru `soubor1` a druhý příkaz vlastníkovi nastaví plná práva a ostatním práva `r-x` na soubor `soubor2`.

Změnu vlastníka souboru zajišťuje program `chown`, který se spouští s následujícími parametry:

```
chown vlastnik.skupina soubor
```

Ve většině implementací UNIXu je vykonání tohoto programu omezeno pouze na správce systému, neboť poněkud otevírá možnost machinací s kvótami prostřednictvím „darování" některých souborů jiným uživatelům.

A.5.5 Domovský adresář

Každý uživatel dostane přidělen domovský adresář (home directory), ten je při přihlášení nastaven jako pracovní (working directory).

Příkazy pro práci s adresáři:

- `pwd` - print working directory - zobrazí název pracovního adresáře (úplné jméno adresáře).
- `cd [adresář]` - změní pracovní adresář na `adresář` nebo (pokud `adresář` není uveden) na domovský adresář.
- `mkdir adresář` - založí `adresář`
- `rmdir adresář` - zruší prázdný `adresář`. Neprázdný adresář lze smazat výše popasným příkazem `rm -r adresář`.

A.5.6 Hledání souborů

Hledání realizujeme pomocí příkazu `find`, jehož formát je

```
find začátekHledání výraz
```

kde `začátekHledání` je seznam mezerou oddělených počátečních bodů prohledávání (adresářů), `výraz` je seznam činností, které se provádí při hledání. Často používané činnosti jsou

- `-name názevSouboru` - vyhledání souboru zadaného jména
- `-print` - výpis úplného jména nalezených souborů
- `-user` - výpis souborů, které vlastní zadaný uživatel

Příkaz

```
$ find /home -name chrd14.zip -print
```

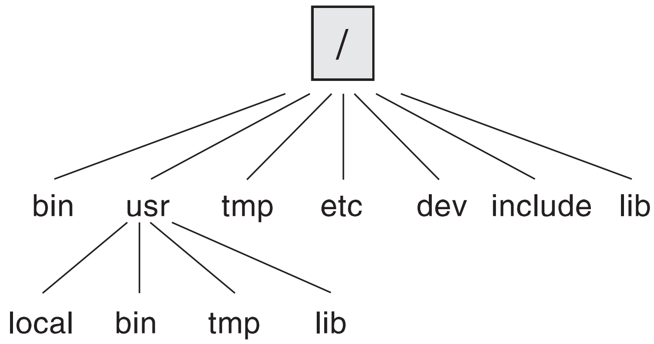
vyhledá v podadresářích adresáře `/home` soubor `chrd14.zip` a všechny jeho výskyty vypíše na obrazovku.

A.5.7 Standardní struktura adresářů

Významnou vlastností systémů typu UNIX je, že své soubory umisťuje do standardizované struktury adresářů. To je také důvod, proč je možné přecházet od jednoho UNIXu k druhému a prakticky všechny zásadní soubory najdeme vždy na stejném místě.

Vyjděme z obrázku 33 a popišme význam některých adresářů:

- / kořenový adresář obsahuje většinou několikamegabytový soubor, který obsahuje jádro systému a ze kterého se systém zavádí
- **/bin** adresář s programy nutnými pro zavedení a běh systému (mount, shell, ...)
- **/etc** obsahuje konfigurační soubory systému (soubor passwd se seznamem uživatelů, nastavení protokolů pro komunikaci, inicializační sekvence operačního systému, konfigurace pošty a řada dalších)
- **/dev** adresář se speciálními soubory zařízení
- **/home** kořen pro domovské adresáře uživatelů
- **/tmp** adresář pro ukládání dočasných souborů pro programy



Obrázek 33: Struktura adresářů

- **/lib** sdílené knihovny
- **/var** soubory měnící svoji velikost (záznamy o provozu systému)
- **/usr** adresář, kam se ukládá prakticky všechen software a jeho datové soubory.
- **/usr/lib** sdílené knihovny a datové soubory používané systémovými programy
- **/usr/bin** obslužné programy systému, které nejsou nutné pro korektní činnost systému (nemusí zabírat místo v souborovém systému /)
- **/usr/local** adresář, kam se instalují programy nesouvisející s provozem operačního systému (uživatelské aplikace). V prostředí sítě pracovních stanic a centrálního serveru je běžné práve tento adresář umístit fyzicky na serveru a prostřednictvím služby NFS ho připojit ke všem pracovním stanicím - jednotná správa softwarového vybavení.

A.5.8 Archivování souborů

Pro archivování souborů se používá program `tar`, pro kompresi dat programy `compress` a `uncompress`, případně `gzip` a `gunzip`. Utilita `tar` dovede komunikovat s páskovou jednotkou, je možné pomocí ní pouze vytvořit soubor (`archiv.tar`), který v sobě obsahuje archivovaný systém adresářů i se soubory. Nejjednodušší použití, ve kterém se do souboru `archiv.tar` zahrnou všechny soubory v aktuálním adresáři (i s podadresáři) ukazuje následující příklad:

```
$ tar cvf archiv.tar *
```

Obnovení struktury adresářů a souborů z archívu provede příkaz

```
$ tar xvf archiv.tar
```

Volitelný parametr `v` nařizuje programu `tar` vypisovat jméno právě archivovaného souboru. Místo parametrů `c,x` lze použít řídící parametry `t` pro provedení testu archívu nebo `l` pro pouhý výpis názvů souborů v archívu.

Program `compress` kompresuje soubor, výsledný soubor bývá zvykem pojmenovat stejným jménem a příponou .Z (pro obnovení původního souboru slouží program `uncompress`).

Program `gzip` pracuje obdobně jako compress, komprese souboru `archiv.tar` provedete příkazem

```
gzip archiv.tar
```

vznikne automaticky komprimovaný soubor `archiv.tar.gz`, který dekomprimujete příkazem

```
guzip archiv.tar.gz
```

A.6 Interpret příkazů, procesy

Po přihlášení uživatele je spuštěn interpret příkazů (shell), který vykoná příkazové soubory `/etc/profile` a `.profile` v domovském adresáři a zobrazí prompt, který dává najevo připravenost přijmout příkaz.

Existuje několik druhů shellu, které jsou ale v principu totožné, liší se některými charakteristickými vlastnostmi (Korn, C, Born, Bash). Jestliže se v parametru příkazu vyskytne některý ze znaků ?,* nebo [a], je tento znak nahrazen jmény všech souborů odpovídajících příslušné specifikaci. Znaky mají následující význam:

?	libovolný jeden znak
*	libovolný řetězec znaků (žádný nebo více)
[ ]	libovolný počet znaků v hranatých závorkách

```
$ ls te*
test
temp
```

Interpret příkazů umožňuje pracovat s proměnnými, které tvoří `prostředí (environment)`. Proměnné prostředí jsou použitelné i pro aplikace vytvářené uživatelem. Výpis prostředí (obsahu proměnných prostředí) získáme příkazem `set` (zkrácený výpis ukazuje následující ukázka).

```
$ set
HOME=/home/kopka               jméno domovského sdresáře
MAIL=usr/spool/mail/kopka      jméno souboru s poštou
PATH=.:/bin:/usr/bin:          cesty spustitelných souborů
PS1=$                          prompt
TERM=vt220                     typ terminálu
```

Proměnné nastavujeme přiřazením hodnoty názvu proměnné, za znakem = následuje buď přímo hodnota bez mezery, nebo hodnota s mezerami uzavřená do apostrofů. Pokud uvedeme jako parametr příkazu proměnnou, před kterou bude znak $, nahradí se proměnná svou hodnotou.

```
$ echo $HOME
/home/kopka
$ PATH=$PATH:/usr/local/bin
```

Do proměnné můžeme uložit také výstup z programu, jestliže použijeme obrácených apostrofů. Využití ukazuje následující ukázka.

```
$ HERE=`pwd`
$ echo $HERE
/usr/local/bin
$ cd
$ pwd
/home/kopka
$ cd $HERE
$ pwd
/usr/local/bin
```

A.6.1 Vstup a výstup

Každý proces v unixu používá standardně tři deskriptory souborů – standardní vstup, výstup a chybový výstup. Všechny tři bývají nasměrovány na terminál. Jednoduchým způsobem je možné některý z těchto standardních souborů přesměrovat na libovolné zařízení. K tomu se používá znaků `<` a `>` použitých při volání programu.

`ls -la > vypis`	do souboru `vypis` zapíše výsledek příkazu `ls -la`
`more vypis`	zobrazí obsah souboru `vypis`
`program < vstup 2> chyby`	spustí program, vstup bude brát ze souboru `vstup` a chybová hlášení posílat do souboru `chyby`, který založí
`program < vstup 2>> chyby`	spustí program, vstup bude brát ze souboru `vstup` a chybová hlášení přidávat do souboru `chyby` (který v případě potřeby založí)

Jestliže potřebujeme výstup z jednoho procesu přesměrovat na vstup do dalšího procesu bez použití souboru, můžeme použít tzv **rouru** (*pipe*). Program, který čte ze standardního vstupu, může číst i z roury, systém sám zařídí přesměrování jeho vstupu, některé programy mají speciální přepínač pro akceptování vstupu ze standardního vstupu místo ze souboru (tar, gzip).

```
$ ls -la | more
```

A.6.2 Procesy

Proces můžeme chápat jako realizaci programu, UNIX podporuje souběžnou činnost více procesů. Jeden z procesů, který běží od začátku relace až do jejího ukončení, je shell. Seznam procesů, které má uživatel spuštěny, vypisuje příkaz `ps`.

```
$ ps
  PID TTY      S           TIME COMMAND
 1745 ttyp2    I  +     0:00.22 -bash (bash)
 1752 ttyp2    I        0:05.69 emacs
```

Proces můžeme realizovat buď jako vykonání programu, na jehož výsledek shell čeká (prompt se objeví až po ukončení procesu), nebo tzv. na pozadí, kdy prompt je zobrazen ihned po spuštění procesu. Na pozadí pouštíme zejména dlouho trvající procesy. Spuštění na pozadí se provede, je-li na konci příkazového řádku znak `&`.

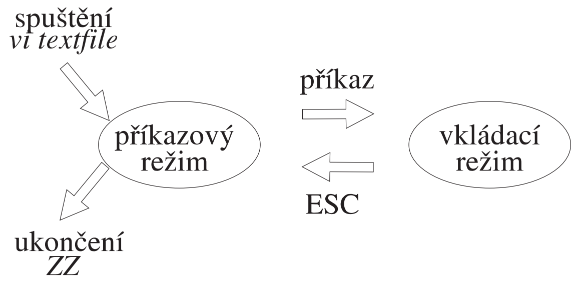
```
$ find / -name HledanySoubor -print > VystupniSoubor 2>Chyby &
$ netscape &
```

Zrušení běhu procesu běžícího na pozadí provedeme příkazem `kill`. Jestliže příkaz `ps` vypsal, že identifikační číslo procesu find je např. 22544 (první sloupec), pak zrušení procesu provedeme

```
$ kill 22544
```

A.7 Editor vi

Editor `vi` je standardní součástí každého systému UNIX. Existuje celá řada editorů, které jsou uživatelsky příjemnější (emacs, mc editor), nicméně základní seznámení s tímto editorem je užitečné, neboť uvedené editory nejsou standardní výbavou každého UNIXu. Editor vi pracuje ve dvou režimech, příkazovém a vkládacím. Příkazový režim umožňuje opravovat, ukládat text, vkládací režim slouží pro vkládání znaků. Základní stav po spuštění je příkazový režim. Příkazy jsou většinou složeny z písmene, popřípadě více písmen. Některé příkazy začínají dvojtečkou a ty se ukončují klávesou *Enter*. Obrázek 34 ukazuje přepínání mezi stavy.



Obrázek 34: Stavy editoru vi

A.7.1 Základní příkazy editoru

i	vkládání textu před kurzor
a	vkládání textu za kurzor
A	vkládání textu za konec řádku
I	vkládání textu na začátek řádku
x	zrušení znaku pod kurzorem
dd	rušení řádku
:w	uložení souboru
:w	:w newfile - uložení pod jménem newfile
:q	ukončení editace
:q!	ukončení editace bez uložení změn
ZZ	uložení souboru a konec editace
u	undo - zrušení poslední změny
U	undo line - obnoví všechny změny na řádku
/	hledání řetězce / řetězec
:r	načtení souboru do editovaného textu
:r	:r textfile

Tento seznam příkazů není úplný, editor poskytuje možnosti práce s více soubory, pojmenovanými oblastmi, ..., se kterými se čtenář může seznámit v literatuře.

A.8 Dávkové soubory (scripty)

Příkazy postupně zadávané z příkazového řádku mohou být uloženy v souboru a ten vykonán (obdoba dávkového souboru v DOSu). Textový soubor, dostane-li atribut `x` (právo execute), může být spuštěn jako script.

Ve scriptu je možné pracovat s proměnnými, akceptuje parametry zadané za názvem skriptu. Prostřednictvím `$1` až `$9` jsou přístupné parametry scriptu, proměnná `$#` obsahuje počet parametrů a `$*` obsahuje množinu všech argumentů. Příkaz `read variable` načte ze vstupu hodnotu proměnné variable. Následující ukázka vyhledá uživatele na aktuálním systému UNIX.

```
# znakem hash je uvozovana poznámka
# Program hledá uživatele na aktuálním systému unix
echo -n "Hledání uživatele: "
read uzivatel
who | grep $uzivatel
```

A.8.1 Podmíněný příkaz

Jazyk, ve kterém je psán script, obsahuje podmíněný příkaz ve formátu

```
if condition
```

```
then
  command_if_true
[else
  command_if_false]
fi
```

Condition (podmínka) může nabývat hodnot `true` - příkaz v podmínce proběhne úspěšně, nebo `false` - v opačném případě. Úspěšnost provedení příkazu pozná shell podle čísla, které každý proces při svém ukončení vrací (tzv. *stavový příznak* je uložen v proměnné `$?`). Když bychom přečetli, že `grep` vrací true, jestliže najde aspoň jeden výskyt řetězce, jinak 1, mohli bychom upravit předchozí ukázku.

```
# Program hledá uživatele na aktuálním systému unix
echo -n "Hledani uzivatele: "
read uzivatel
if who | grep $uzivatel
then echo -n " "
else echo "$uzivatel neni prihlasen"
fi
```

Další podmíněný příkaz (`test`) umožňuje testovat existenci proměnné, souboru, adresáře. Příklad ukazuje script, který kopíruje první parametr do druhého v případě, že druhý parametr není existující soubor.

```
# cpx
# copy if not exist
if [ -f $2 ]
then echo "cpx: soubor $x2 existuje"
else cp $1 $2
fi
```

Příkaz test je dost silným nástrojem pro práci se souboru, proto uveďme aspoň některé přepínače:

`test -f filename`	vrací `true`, jestliže soubor filename existuje a není to adresář ani speciální soubor
`test -r filename`	vrací `true`, jestliže soubor filename existuje a lze z něj číst (obdobně w, x)
`test -d filename`	vrací `true`, jestliže soubor filename existuje a je to adresář

A.8.2 Cyklus

Pro vytvoření programového cyklu slouží příkaz `for`

```
for variable in list_of_values
do commands
done
```

Proměnné variable jsou postupně přiřazovány hodnoty ze seznamu `list_of_values`:

```
for file in *
do
   echo "Delete file $file ? [A/N]"
   read confirm
     case $confirm in
      [aA]*) rm $file;;
     esac
done
```

Další konstrukce cyklu jsou `while` a `until`

```
while condition
do commands
done
```

A.9 Konfigurace uživatelského prostředí

A.9.1 Soubor .profile

Soubor .profile v domovském adresáři je script, který je vykonán jako první při přihlášení do systému. Nastavuje systémové proměnné, zejména TERM, PATH a může spustit test existence nové pošty ....

A.9.2 Změna hesla

Systémy UNIX jsou běžně nakonfigurovány tak, že každý uživatel může měnit své přístupové heslo. K tomu slouží příkaz `passwd` nebo `yppasswd`. Komunikace s programem `passwd` probíhá standardně tak, že si vyžádá vaše staré heslo (ověří vaši autorizaci) a pak dvakrát požádá o zadání hesla nového. Jestliže zadáte správně dvakrát stejné nové heslo, změna se provede.

Různé systémy kladou různé požadavky na formát hesla (délka, výskyt speciálních znaků) z důvodu zabezpečení systému proti vnějšímu útočníku.

A.9.3 Soubor .forward

Může se stát (a je to velmi častý jev), že počítač s operačním systémem UNIX je uzlovým počítačem sítě Internet a vy běžně pracujete na počítačích v síti Novell nebo MS Windows 95/NT. Používáte-li elektronickou poštu, pak tato vám dochází na váš účet na daném počítači s UNIXem. Žádáte-li, aby veškerá pošta byla doručována na jiný počítač ve vaší síti (obecně libovolný počítač v Internetu), založte ve Vašem domovském adresáři soubor `.forward` a do něj uveďte, na jakou adresu má být pošta posílána. Například uživatel `janacek` bude mít na uzlovém počítači `risc.upol.cz`, který je poštovním serverem, soubor `.forward` obsahující řádek

```
janacek@infnw.inf.upol.cz
```

A.10 Komunikace v UNIXu

Cílem této kapitolky je ukázat několik běžných způsobů komunikace v operačním systému UNIX a mezi UNIXy v počítačové síti Internet.

A.10.1 Vzdálené přihlašování

Vzdálené přihlášení (relace) je v síti Internet zajišťována protokolem a programem `telnet`. Program `telnet` je spouštěn s parametrem označující symbolické jméno vzdáleného počítače. Například příkaz

```
$ telnet alpha.inf.upol.cz
Trying 158.194.80.20
Connected to alpha.inf.upol.cz
....
```

se pokusí navázat kontakt s počítačem stanoveného jména a objeví se výstup informující o navázání kontaktu a následně jste žádáni o přihlašovací jméno a heslo.

Po zadání identifikace (jméno, heslo) může uživatel pracovat na vzdáleném počítači a využívat veškeré jeho zdroje. Pracuje ovšem v prostředí vzdáleného počítače, s jeho uživatelským rozhraním, na jeho hardware.

A.10.2 X-Window

Telnet poskytuje pro práci na vzdáleném počítači pouze základní textové rozhraní, které většině aplikací nedostačuje. Přitom počítače, které poskytují možnost vzdálené práce, jsou většinou velmi výkonné serverové počítače, u kterých je grafické rozhraní samozřejmostí.

Tuto nevýhodu překonává systém X-Window, který připojuje uživatele ke vzdálenému počítači v grafickém režimu. Oba dva komunikující počítače musí být nakonfigurovány tak, aby mohly prostřednictvím X-Window komunikovat.

Základní princip X protokolu používaného v X-Window je klient-server model. X klientem je zde vzdálený počítač, který vysílá povely (požadavky) na vykreslení grafických objektů, X serverem je lokální počítač, který se stará o vykreslení zaslaných pokynů.

Jestliže jsou oba počítače obsluhovány operačním systémem UNIX, pak z uživatelského pohledu je nutné nakonfigurovat následující:

- **X klient** musí dovolit uživateli, aby se vůbec připojil a nalogoval prostřednictvím X protokolu (spustil vzdáleně xterm). Povolení uděluje X klient (respektive uživatel na počítači X klienta) založením souboru `.rhosts` ve svém domovském adresáři.

  Obsahem uvedeného souboru je seznam počítačů, ze kterých se dovoluje provádět vzdálené příkazy na počítači X klienta a identifikace uživatele, kterému se toto dovoluje. Příkladem jsou následující dva řádky souboru `.rhosts`, které zapíše uživatel `jelinek` na serveru, který má dovolit vzdálené přihlašování z uvedených dvou stanic právě uživateli `jelinek`.

  ```
  gamma.inf.upol.cz jelinek
  delta.inf.upol.cz jelinek
  ```

  X klient říká svým aplikacím, aby se zobrazovaly na příslušném výstupním zařízení X serveru parametrem `-display xserver:0.0`.

- **X server** musí dovolit zobrazovat požadavky vysílané příslušným X serverem. Povolení X serveru `alpha.inf.upol.cz` lze udělit dvojím způsobem, a to příkazem
  `xhost alpha.inf.upol.cz`
  nebo uvedením jména X serveru v souboru `/etc/X0.hosts`.

A.10.3 Komunikace s jiným uživatelem

Jestliže potřebujeme kontaktovat uživatele pracujícího na konkrétním počítači, máme k dispozici interaktivní textový komunikační program - `talk`. Program talk se spouští s jedním argumentem, a to identifikací uživatele na vzdáleném nebo lokálním uzlu. Na uzlu `claudius.inf.upol.cz` může uživatel `kolar` psát

```
talk novak@alpha.inf.upol.cz
```

a na terminálu uživatele `novak` se objeví hlášení

```
Message from Talk_Daemon@alpha.inf.upol.cz at 13:25
talk: connection requested by kolar@claudius.inf.upol.cz
talk: respond with: talk kolar@claudius.inf.upol.cz
```

Uzivatel `novak` naváže s uživatelem spojení příkazem, který mu byl předložen v hlášce:

```
$ talk kolar@claudius.inf.upol.cz
```

Program `talk` se ovládá jednoduše - do jedné poloviny okna terminálu píše jeden uživatel svůj text a současně se mu ve druhé polovině ukazuje text psaný jeho kolegou.

A.10.4 Informace o jiném uživateli

Občas potřebujeme zjistit, jestli nějaký uživatel právě pracuje na počítači, jehož jméno známe. K tomuto účelu slouží program `finger`. Ten lze použít ve dvou formátech

```
finger @alpha.inf.upol.cz
finger kopka@alpha.inf.upol.cz
```

Zatímco první příkaz vypíše seznam aktuálně všech pracujících uživatelů na daném uzlu `alpha.inf.upol.cz`, druhý příkaz vypíše informace o uživateli `kopka` evidovaném na uzlu `alpha.inf.upol.cz`, které doplní obsahem souboru `.plan` z jeho domovského adresáře. Soubor `.plan` zakládá volitelně každý uživatel sám a slouží právě k těmto účelům.

A.10.5 FTP

Protokol a program FTP slouží k přenosu souborů mezi uzly počítačové sítě. Uživatel se, podobně jako u telnetu, hlásí na vzdálený uzel podporující FTP, zadá svoje uživatelské jméno a heslo a pak má dovoleno ze vzdáleného uzlu přenášet soubory nebo je na tento uzel kopírovat.

Při práci s FTP mějme na paměti, že pracujeme jakoby se dvěma panely správce souborů. Jeden panel je nasměrován do aktuálního adresáře na lokálním počítači a druhý do aktuálního adresáře na vzdáleném počítači. K ovládání tohoto „správce souborů" používáte jednoduchý příkazový jazyk popsaný v následující tabulce:

`open` *adr-poč*	otevření FTP relace ke vzdálenému počítači
`close`	uzavření relace se vzdáleným počítačem
`dir`	vypsání obsahu aktuálního adresáře vzdáleného počítače
`ldir`	vypsání obsahu aktuálního adresáře lokálního počítače
`pwd`	vypsání jména aktuálního adresáře vzdáleného počítače
`lpwd`	vypsání jména aktuálního adresáře lokálního počítače
`cd` *adresář*	změna aktuálního adresáře vzdáleného počítače
`lcd` *adresář*	změna aktuálního adresáře lokálního počítače
`get` *soubor*	přenos souboru z aktuálního adresáře vzdáleného počítače do aktuálního adresáře lokálního počítače
`put` *soubor*	přenos souboru z aktuálního adresáře lokálního počítače do aktuálního adresáře vzdáleného počítače
`ascii`	přepnutí do ASCII režimu přenosu
`binary`	přepnutí do binárního režimu přenosu (doporučeno pro většinu souborů)
`hash`	zobrazí znak "`#`" pro každý přenesený kilobyte
`!`	spuštění shellu na lokálním počítači, z něho se vrátíte zpět do FTP příkazem `exit`
`bye`	ukončení práce s programem FTP

Tabulka 1: Základní příkazy FTP.

A.11 Síť pracovních stanic

Práce na jednom počítači s operačním systémem UNIX je založena na evidenci uživatelů a jejich hesel a přístupových práv k prostředkům operačního systému. Jak je realizována správa více stanic s operačním systémem UNIX, které mají sloužit určitému okruhu uživatelů, popisuje tato kapitola.

Bylo by jistě velmi neefektivní udržovat na každé jednotlivé pracovní stanici informace o všech uživatelích, protože by to značně znesnadilo správu takové sítě. Otázkou tedy je, jakým způsobem zajišťuje UNIX transparentní pracovní prostředí v síti pracovních stanic. Transparentní v tom smyslu, že uživatel prakticky nepozná (až na výkon jednotlivých stanic), u které stanice právě pracuje.

K řešení jsou tedy nejméně dva problémy - jak zajistit, aby byli na všech stanicích známí všichni uživatelé a jak zajistit, aby měli transparentní přístup do svých domovských adresářů. Oba problémy jsou elegentně řešitelné.

A.11.1 Network Information System (NIS)

Operační systém UNIX má uloženy informace o uživatelích v souboru `/etc/passwd`, popřípadě na dalších místech podle úrovně utajení systému (shadow passwords). Řešení našeho problému spočívá v tom, že jednu ze stanic (většinou serverový počítač) prohlásíme za **NIS server** a uložíme na něj databázi všech uživatelů, kteří mají mít přístup k naší síti. Na jednotlivých stanicích sítě pracuje klientský program, který veškeré dotazy systému ohledně uživatelů zodpovídá prostřednictvím svého partnerského programu na NIS serveru. Řešení je zakomponováno až do úrovně jádra UNIXu, takže bez problémů fungují všechny služby operačního systému.

NIS je provozováno většinou pouze v rámci jedné domény.

A.11.2 Network File System (NFS)

NFS je původní produkt firmy SUN, ale dnes je součástí každého UNIXu (ikdyž u komerčních verzí se mnohdy musí dokupovat jako zvláštní modul).

NFS slouží ke sdílení podadresářových stromů v rámci sítě. Užitečnost tohoto řešení se ukazuje v síti pracovních stanic, kdy domovské adresáře jsou většinou fyzicky umístěny na souborovém serveru, který poskytne podadresářový strom `/home` umezené skupině počítačů ke sdílení (operace `export`).

Jednotlivé pracovní stanice připojí (`mount`) tento podstrom do svého souborového systému a vzhledem k tomu, že berou informace o uživatelích z centrální databáze NIS, jsou zajištěna i správná práva jednotlivých uživatelů do svých domovských adresářů.

NFS pronikl i do dalších operačních systémů, takže je tímto způsobem možné sdílet disky mezi libovolnými operačními systémy, které mají implementovány NFS klienty případně NFS servery.

Odkazy

[1] Peterka,J.,: *Co je čím v počítačových sítích*, Computerworld, seriál článků

[2] Frey, P., Zeltvay, P., Vlasák, J.: *Lokální počítačové sítě*, Adcomp 1991

[3] Tanenbaum, A., S., : *Computer Networks*, Prentice Hall, 1989

[4] Comer, Douglas, E.: *Internetworking with TCP/IP*, Prentice Hall, 1991

[5] *LAN COM* - časopis pro počítačové sítě a počítačovou komunikaci, CSc Computer Services, Praha 1991 - 1996

[6] Stallings, W.: *Network and Internetwork Security*, Prentice Hall 1995.

[7] Dočekal, D.: *Novell Netware - seznam a popis příkazů*, PLUS, Praha 1990

[8] *Novell Netware - soubory originálních manuálů k síťovému operačnímu systému Novell Netware*, Novell Inc., Provo Utah USA

[9] Volák, P., Bič, J.,: *Od LAN k WAN*, GRADA

[10] Brodský J., Skočovský L.: *Operační systém Unix a jazyk C*, SNTL Praha, 1989.

[11] Kauffels, F-J.: *Personální počítače a lokální sítě*, UNIS, Brno 1992

[12] Motyčková,.L, Staudek, J., : *Počítačové sítě*, Výběr informací z organizační a výpočetní techniky, Praha 1990

[13] Šárek, M., Ošmera, P., Švéda, M., Horský, J.: *Počítačové sítě*, skripta VUT Brno, 1993

[14] Rieger, P., Straka, L.: *Počítačové sítě Novell* PC-DIR, Brno 1991

[15] Staudek, J.,: *Přenos dat a počítačové sítě*

[16] *Sborníky z mezinárodního semináře ,,Počítačové sítě Novell"* PC-DIR, Brno 1990, 1991, 1992, 1993

[17] Motyčková, L., Staudek, J.: *Sítě ATM*, Computer Echo 3-4/1995.

[18] Jandoš, J. a kolektiv: *Sítě LAN*, Výběr informací z organizační a výpočetní techniky, Kancelářské stroje, Praha 1991

[19] Přichystal, O.: *Téměř vše o sítích Novell*, GRADA, Praha 1992

[20] Brandejs, M.: *Unix Linux*, Grada, Praha 1996.

[21] Sova, M.: *Unix V, úvod do operačního systému*, Grada, Praha 1991.

[22] Costa, J.F.: *Vysokorychlostní sítě 100VG/AnyLAN*, Prentice Hall, 1995, ISBN 80-7169-189-5