# 1. Úvod, historie

## 1.1 Historie vzniku TCP/IP, vztah k NCP a ARPANETu, vývoj "vlády nad Internetem"

Počátek vzniku TCP/IP byl položen již koncem 50. let, kdy americká armádní grantová agentura (ARPA) začala financovat vědu a výzkum. Díky tomu v 60. letech vzniká koncepce paketového přenosu. Aby bylo možné ověřit jeho použitelnost, byla akademiky vytvořena síť ARPANET (název je podle agentury ARPA, která projekt financovala). Později byl ARPANET americkým ministerstvem obrany plně předán akademické sféře k využívání a dalšímu vývoji. Pro účely testování této sítě vznikl experimentální protokol NCP (Network Control Program), který ale nebyl vhodný pro rutinní použití. Nebyl totiž v souladu s vrstevnatým modelem (staral se o činnost více vrstev najednou). Jako jeho náhrada vznikal protokol TCP, který měl také původně zajišťovat činnost transportní i síťové vrstvy. Časem se ale rozdělila práce mezi protokoly TCP (transportní vrstva) a IP (síťová vrstva). Dnes se jedná o celou síťovou architekturu čítající přes sto protokolů jejíž součástí je i představa o vrstvách a jejich úkolech, problematika systému IP adres, jmenného prostoru DNS, standardizačního procesu...

V roce 1981 byly publikovány RFC 791 a 793, které definovaly TCP a IP. V roce 1983 pak byl NPC plně nahrazen TCP/IP.

Původně byla tato technologie vyvinuta za peníze amerických daňových poplatníků a tím pádem jsou její specifikace veřejným vlastnictvím. Dnešní vývoj probíhá za peníze z komerční sféry, ale rozhodování o dalších postupech vlastní vývoj probíhá pod taktovkou širší vědecké veřejnosti.

## 1.2 Dokumenty RFC, STD, FYI a BCP

RFC je označení dokumentů, které jsou obecným „publikačním mechanismem" pro publikování standardů, informačních materiálů, výsledků experimentů... Jsou označeny pořadovým číslem a nikdy se nemění. Pokud je potřeba změnit RFC dokument, vydá se nový, ve kterém se označí, že nahrazuje ten původní (resp. mění některé informace). Tyto dokumenty jsou volně dostupné a šiřitelné a jsou jich již tisíce (přes 5 tisíc v roce 2008).

STD je takový soubor RFC dokumentů, který publikuje standard pro určitou problematiku. To znamená, že jeden standard jedné technologie může být složen z více RFC.

FYI (For Your Information) je také soubor RFC dokumentů zabývající se jednou problematikou, ale má informační charakter (tedy RFC v tomto souboru jsou informační).

BCP (Best Current Practices) je soubor RFC týkajících se jednoho společného problému a publikujících návody a doporučené postupy pro řešení tohoto problému.

## 1.3 Standardy TCP/IP a standardizační orgány

Standardy definující části síťové architektury TCP/IP se vydávají jako RFC dokumenty. Ty jsou shlukovány do svazků s označením STD.

Standardizací se původně zabývala společnost ICCB, která se později změnila na IAB. Ta měla pod sebou několik pracovních skupin zabývajících se konkrétními úkoly. Později se množina pracovních skupin rozdělila na dvě množiny. Jedna se značila IETF a sdružovala pracovní skupiny zabývající se aktuálními problémy (ty, které mají význam pro aktuální praxi). Druhá nesla označení IRTF a v její kompetenci byl výzkum (budoucí řešení problémů). Časem se také nad IETF i IRTF ustálily

skupiny označované jako „dozorčí rada” IETF a „dozorčí rada” IRTF.

Zastřešení nad veškerou aktivitou týkající se Internetu převzala od vlády USA společnost ISOC (Internet Society), která se tak postavila nad IAB. Dodnes tato organizace zastřešuje také standardizační proces (formálně standardy ale vydává IAB). Přestože vydávané standardy nejsou zákonné (nejsou „de jure”, jsou „de facto”), jsou velmi důsledně dodržovány. Standardy dnes nejsou přímo vyvíjeny skupinou IETF, ale spíše vznikají ve firmách, které je předkládají ke standardizaci a IETF z nich vybírá. Nic však není přijato jako standard, dokud to není alespoň dvakrát implementováno, neprošlo to testovacím provozem a neproběhla o tom veřejná diskuse.

O standardizaci WEBu se stará W3C.

## 1.4 Role ISOC, ICANN a WSIS ve vývoji Internetu a jeho standardů

ISOC je agentura zaštiťující veškerou standardizační činnost týkající se Internetu.

ICANN je organizace, která přebrala odpovědnost za distribuci IP adres, jmenný prostor DNS a správu čísel portů po organizaci IANA. Ta sice pracovala pod ISOC, ale byla formálně vedena vládou USA. ICANN ještě koordinuje vazbu na ostatní standardizační orgány a správu a provoz kořenových name serverů.

WSIS je summit (proběhl zatím dvakrát), na kterém se snaží země třetího světa dosáhnout toho, aby kompetence ICANNu byly převedeny na nějakou mezinárodní a nezávislou organizaci. Stále jsou v této organizaci totiž silně zastoupeny americké orgány (organizací sídlí v USA a většina jejích členů jsou Američané).

## 1.5 Jak je řešena správa TLD .cz? Včetně historického vývoje …

Zpočátku byla v ČR uplatňována restriktivní politika - byla zavedena velmi přísná pravidla pro registraci domén. Nejprve byla správcem VŠCHT v Praze a později se jím stala firma CoNET.

Výrazné změny nastaly v roce 1997, kdy se správcem domén stává sdružení CZ.NIC. Zavádí liberální politiku a pravidla pro zřizování domén se výrazně uvolňují. Zhruba dva roky však trvalo, než CZ.NIC převzalo plnou správu.

Do roku 2002 bylo možné spekulovat s doménami. Zájemce požádal o doménu a ta byla zřízena a byla „zamluvená” dokud neuběhla lhůta, po kterou musel zájemce za doménu zaplatit. Pokud zaplatil, měl ji pro sebe registrovanou. Pokud nezaplatil, byla zrušena a nabízena k další rezervaci. Protože tohoto postupu využívali hlavně spekulanti, změnil se postup registrací. Nyní je doména zřízena až po zaplacení.

Koncem roku 2003 došlo k „decentralizaci” správy domén. Pravomoc přidělovat domény mají registrátoři, jejich činnost koordinuje právě CZ.NIC.

V roce 2006 byla podepsána dohoda o spolupráci Ministerstva Informatiky ČR a CZ.NIC, ve které stát uznává sdružení CZ.NIC jako správce TLD .cz a vymáhá si jedno křeslo v řídících orgánech CZ.NIC.

# 2. Architektura TCP/IP

## 2.1 V čem se lišil přístup autorů TCP/IP a ISO/OSI při vzniku obou architektur?

Tvůrci ISO/OSI se spoléhají na vlastní síly a všechno řeší sami (nebo převezmou cizí technologii a vydají jí jako vlastní standard). Postupují od složitějšího k jednoduššímu, takže nejprve vznikne nabubřelé řešení, které je považováno za dokonalé, a pak se z něj ukrajuje co není potřeba. Navíc se všechno nejdříve standardizuje a pak se teprve zkoumá, jestli to vůbec funguje a jak moc je to praktické.

Tvůrci TCP/IP nespoléhají jen na sebe, ale spíše se více snaží provázat vlastní řešení s již dobře fungujícím cizím řešením. Nejprve jsou skromní a teprve postupně na jednoduché řešení nabalují vylepšení, která se osvědčila při testování praktickým používáním.

## 2.2 Srovnejte způsob řešení otázky spolehlivosti v TCP/IP a ISO/OSI.

ISO/OSI se snaží o vytvoření inteligentní sítě, takže spolehlivost zajišťuje síťová vrstva. Takže se inteligence umisťuje do směrovačů, což je ale drahé nepříliš pružné.

TCP/IP zastává názor, že inteligence má být mimo síť (v koncových uzlech) a samotná síť má prostě co nejrychleji přenést co je potřeba bez ohledu na spolehlivost. Tu si mohou zajistit koncové uzly, pokud ji chtějí.

## 2.3 Koncepce vrstvy síťového rozhraní v TCP/IP, srovnání s ISO/OSI.

ISO/OSI je omezeno tím, že definuje i linkovou a fyzickou vrstvu. Nemůže tedy fungovat nad čímkoliv.

TCP/IP definuje jen vrstvy od síťové „nahoru", takže mu nezáleží na tom, jaká přenosová technologie bude použita. Zajímá se jen o to, jak propojit síťovou vrstvu s vrstvou síťového rozhraní. Výjimkou jsou protokoly SLIP a PPP, které zasahují i do vrstvy síťového rozhraní a definují způsob přenosu po dvoubodových spojích.

## 2.4 Koncepce síťové vrstvy TCP/IP.

Koncepce síťové vrstvy vychází z katenetového modelu „světa". Protože internet je tvořen vzájemně propojenými sítěmi, které mohou různě fungovat, rozhodli se tvůrci TCP/IP vytvořit jednotnou „pokličku", které bude skrývat specifika jednotlivých IP sítí, a tím pádem se jimi TCP/IP nebude muset zabývat. Tato poklička je posazena právě na síťovou vrstvu a prochází skrz ní pouze údaj MTU (Maximmum Transfer Unit = maximální velikost linkového rámce), díky čemuž je zamezeno zbytečné fragmentaci přenášených dat.

V síťové vrstvě jsou zabudovány síťové adresy, mechanismy překládající mezi fyzickými adresami a virtuálními IP adresami, fragmentační mechanismy a protokoly podporující fungování síťové vrstvy. Nově do této vrstvy byly také přidány mechanismy překladu adres (NAT), koncept privátních IP adres, mechanismy dělení a sdružování adres, bezpečnostní mechanismy a podpora mobility.

Velmi úzce se síťovou vrstvou souvisí protokoly podporující směrování, mechanismy přidělování IP adres a mechanismy překladu mezi symbolickými doménovými jmény a IP adresami.

## 2.5 Koncepce transportní vrstvy TCP/IP.

Transportní vrstva realizuje „end-to-end" komunikaci a nabízí k tomu dva přenosové protokoly. Mimo toho zajišťuje také multiplex a demultiplex.

Protokol TCP (Transmission Control Protocol), který funguje spojovaně a spolehlivě. Vzhledem k aplikacím, které ho využívají, se tváří jako by přenášel proud bitů (předává jim bity postupně), ale sám ve skutečnosti přenáší bloky nazvané TCP segmenty. Pro zajištění spolehlivosti používá kontinuální potvrzování a selektivní opakování. Tento protokol se dokáže dobře přizpůsobovat různým podmínkám přenosu jako jsou přenosové zpoždění a rozptyl zpoždění. Jeho implementace je velmi komplikovaná (rozsáhlý a složitý kód)

Protokol UDP (User Datagram Protocol) je jednoduchou nadstavbou nad síťovým IP protokolem. Funguje nespojovaně a nespolehlivě. Aplikace mu data předávají po blocích a on je pak „balí" do datagramů.

## 2.6 Koncepce aplikační vrstvy TCP/IP, způsob řešení prezentačních a relačních služeb.

Aplikační vrstva TCP/IP zahrnuje vše co měly v ISO/OSI na starost vrstvy aplikační, prezentační a relační. Důvodem tohoto kroku bylo to, že ne všechny aplikace potřebují služby prezentační a relační vrstvy. Jsou tu ale dva protokoly: RPC (Remote Procedure Call), který zajišťuje relační služby; XDR (eXternal Data Representation), který nabízí prezentační služby. Služby protokolů RPC a XDR využívá pro své fungování protokol NFS. Navíc tyto protokoly může využít i libovolná aplikace (pouze pokud chce).

Pokud chce aplikace složitější relační či prezentační služby, musí si je zajistit sama.

## 2.7 Požadavky multimediálních aplikací, možné přístupy ke QoS v TCP/IP.

Multimediální aplikace potřebují svá data dostávat s malým a pravidelným zpožděním. Tedy by se hodila podpora QoS. Jenže TCP/IP funguje principiálně stylem „best effort". Řešení tohoto problému jsou dvě: kvantitativní (můžeme zvyšovat přenosovou a přepojovací kapacitu a tím zmenšit pravděpodobnost krácení požadavků) a kvalitativní (zavedeme podporu QoS).

K zavedení QoS do TCP/IP můžeme přistupovat třemi způsoby. Prioritizací, kde přiřadíme k různým druhům přenosů různou prioritu a budeme se k nim podle toho chovat, rezervací, což je řešení, při kterém rezervujeme pro potřebu konkrétního přenosu zdroje a ty pak využíváme, a nebo „hrubou silou" (prostě jen zvýšíme přenosovou kapacitu).

## 2.8 Řešení bezpečnosti a mobility v TCP/IP.

Architektura TCP/IP původně vůbec nepočítala se zajištěním bezpečnosti, protože se v přístupu k síti aplikovalo fyzické zabezpečení (ostraha objektu apod.). Přenášená data tedy nejsou chráněna proti odposlechu ani proti ztrátě či změně. Pokud chce aplikace nějaké zabezpečení přenášených dat, musí si jej zařídit. Vlastně to ale celé zapadá do filosofie TCP/IP, protože se vždy snaží nenutit něco těm, co to nepotřebují (aplikace nevyžadující zabezpečení ho nemusí používat). Časem však byl vypracován framework IPSEC pro zajištění bezpečnosti na úrovni síťové vrstvy. Umí šifrovat data (poskytuje důvěrnost) a zajistit, že data nejsou při přenosu změněna (integrita). Má dva režimy: transport mode (zabezpečovací údaje jsou vloženy přímo do IP datagramu) a tunnel mode (IP datagram je vložen do jiného zabezpečeného datagramu).

IP adresy nejsou mobilní (v době vývoje na to nebyl požadavek). To je problém, který bylo

postupem času třeba vyřešit. Povedlo se to několika způsoby. První možností je přidělení nové IP adresy v nové síti a druhou je využití agentů a tunelů (agent zůstane na původním místě a skrz tunel vše přeposílá tam, kde se uživatel právě nachází).

IP Mobility Support funguje pomocí metody agentů. Není to řešení určené pro častou mobilitu, ale na druhou stranu mobilní zařízení nemusí tuto metodu podporovat (nijak ji nevnímá).

## 2.9 Co přináší a řeší IPv6?

Protože IPv4 má 32-bitový adresní prostor, hrozilo jeho vyčerpání. Proto byl vyvinut IPv6, který má 128-bitový adresní prostor (pouhé rozšíření adres v IPv4 nebylo možné). Tento způsob adresování se považuje za dostatečný do vzdálené budoucnosti (dnes by mohl každý žijící člověk na planetě dostat 4 miliardy adres...). Navíc s sebou IPv6 přináší technologická vylepšení v podobě podpory hierarchického směrování, bezpečnosti a QoS.

# 3. IP adresy

## 3.1 Jaká je koncepce IP adres? (návaznost na linkové adresy, složky IP adresy, způsob přidělování IP adres hostitelským počítačům a směrovačům)

Základní představa o IP adresách byla, že budou abstraktní (tedy nebudou mít žádnou souvislost s linkovými adresami), budou mít 32 bitů a budou všude stejné, bez ohledu na typ linkové adresy. Tyto adresy vyhovují také představě katenetového modelu (světová síť je složena z dílčích sítí a ty z jednotlivých uzlů). Směrovací algoritmy se rozhodují na základě příslušnosti uzlu k síti, ne na základě celé adresy. Důsledkem toho je zmenšení objemu směrovacích tabulek a snadnější rozhodování.

Fyzicky jsou adresy celistvé, ale logicky je rozdělujeme na dvě části - síťovou adresu a adresu uzlu v rámci sítě. Hranice mezi těmito složkami tvoří bitová pozice. Ta byla v minulosti pevně dána ve třech možných pozicích, ale dnes může být plně „pohyblivá".

Uzly v jedné síti musí mít stejnou síťovou adresu a uzly z různých sítí tuto adresu musí mít odlišnou. Adresy se tedy přidělují po blocích se stejnou síťovou částí a pokud v přiděleném bloku nějaké adresy zbudou, nelze je vrátit (pro použití v jiné síti).

## 3.2 Dělení IP adres do tříd A až E a důsledky na rychlost čerpání adresového prostoru

Protože bylo potřeba zvolit poměr velikostí složek IP adres tak, aby nedošlo k jejich rychlému vyčerpání (malá síťová část, velká relativní) nebo k výraznějšímu omezení velikosti dílčích sítí (velká síťová část, malá relativní), přišli tvůrci TCP/IP s nápadem vytvořit několik tříd adres. Rozlišujeme je podle počáteční sekvence bitů. IP adresy se pro přehlednost symbolicky zapisují jako 4 čísla (každé je jeden byte) zapsaná v desítkové soustavě oddělená tečkou (např. 192.168.0.2).

Třída A je určena primárně pro velmi velké sítě, protože vyhrazuje 24 bitů pro relativní část. Takových sítí však nemůže být mnoho, protože síťová část má jen 8 bitů. Dá se poznat tak, že začíná bitem 0.

Třída B je určena pro středně velké sítě, protože má poměr rozdělení 16:16. Začíná bity 1, 0.

Třída C má poměr rozdělení 24:8, takže je určena pro malé sítě (ale může jich být celkem hodně). Tato třída začíná bity 1, 1, 0.

Třída D je určena pro multicasting (skupinový přenos) a patří do ní adresy od 224.0.0.0 až do 239.255.255.255.

Třída E je vyhrazena pro budoucí použití a obsahuje adresy od 240.0.0.0 do 255.255.255.255. Adresy z této třídy nebyly nikdy využity pro rozšíření.

Adresy tříd D a E nejsou logicky děleny na dvě složky a lze je přidělovat jednotlivě.

## 3.3 IP adresy se speciálním významem (broadcast, síť jako celek, loopback, multicast adresy)

Mezi IP adresami existují i adresy se speciálním významem. V následujícím textu značka '**|**' slouží jako oddělovač logických částí adresy a '**[**', resp. '**]**', značí začátek, resp. konec adresy.

**[**0**|**0**]** adresuje „sebe sama" (odkaz na tento počítač).

**[0|x]** je adresa počítače na této síti, který má relativní adresu x.

**[x|0]** označuje celou síť s adresou x.

**[x|1..1]** definuje řízený broadcast týkající se jen sítě s adresou x.

**[1..1|1..1]** je adresa pro omezený broadcast pro dannou síť.

**[127|.x.x.x]** je adresa pro loopback (rozhraní, která „nejdou ven").

Multicast adresy jsou adresy třídy D, tedy adresy od 224.0.0.0 až do 239.255.255.255, které nejsou logicky rozděleny.

## 3.4 Způsob distribuce IP adres (přidělovatelé …) a jeho vývoj v čase

Nejprve IP adresy přidělovalo středisko SRI NIC (později přejmenované na IANA) centrálně, protože žádná adresa nesmí být přidělena dvakrát. Kvůli rostoucímu zájmu se to ale stalo organizačně neúnosné a tak byli vytvořeni tři regionální „přidělovatelé" s označením RIR (Regional Internet Registry). Byly jimi organizace RIPE (měla v kompetenci Evropu), APNIC (měla na starosti Asii a Pacifik) a ARIN (obstarávala USA). Tito přidělovatelé byli řízeni organizací IANA, která jim dávala k dispozici celé bloky adres. Přidělovala se vždy nejbližší vyšší třída, což vedlo k enormnímu plýtvání adresami a začalo hrozit vyčerpání adresového prostoru.

## 3.5 Principiální možnosti řešení úbytku IP adres (včetně IPv6/IPNG)

Po zjištění, že IP adresy rapidně ubývají, se objevilo několik způsobů jak to řešit. V zásadě se tyto způsoby dělí také podle časového horizontu jejich účinku.

Okamžité řešení přišlo v podobě změny přidělování IP adres a subnettingu. Přidělování adres se změnilo v tom, že se začaly přidělovat násobky největších nižších kvant (místo jednoho B dostane síť 4-8 C). To se ale neblaze projevuje na velikosti směrovacích tabulek. Subnetting využívá oddělenost některých sítí od světa s pouze jedním přístupovým bodem. V adrese se z pohledu cílové sítě posune bitová hranice a pro dekódování adresy se použije maska. Umožňuje to použít jednu síťovou adresu pro více sítí.

Dočasná řešení jsou CIDR (Classless InterDomain Routing - umožňuje přidělovat IP adresy po libovolných kvantech a ne jen po mocninách 2) a zavedení privátních IP adres. Privátní IP adresy jsou vlastně adresy v privátních sítích, které se nedostanou do zbytku světa, a pokud ano, tak procházejí přes firewall nebo jsou překládány. Toto umožňuje používat jednu IP adresu pro více strojů zároveň.

Definitivním řešením je vytvoření IPv6 někdy označovaného jako IPNG (IP - the Next Generation), což je ale ve skutečnosti souhrnné označení pro všechny návrhy nové verze IP. Nový protokol IPv6 používá 128-bitové adresy. (Hodně adres...)

## 3.6 Princip subnettingu a jeho využití

Tento postup využívá toho, že Internet je vytvořen propojením dílčích sítí. Ty také někdy bývají tvořeny propojením menších sítí. Pokud tedy jedna větší síť je složena z menších, které mají jeden společný přístupový bod (směrovač), můžeme použít subnetting. Zbytek světa se bude chovat, jako by za směrovačem byla jen jedna síť a tak jí může být přidělena jedna síťová adresa. Ve skutečnosti je tam ale více sítí, které rozlišuje jen ten přístupový směrovač. Aby dokázal rozlišit adresy, posune si bitovou hranici v adrese (pomocí masky) a může tak k adresování použít i několik bitů, které byly původně určeny jako relativní část adresy. Je to ideální postup pro sítě se stromovitou strukturou.

## 3.7 Privátní IP adresy a způsob jejich využití

Pokud uzel komunikuje se zbytkem světa nepřímo (přes privátní síť, která je ke zbytku světa připojena bránou či firewallem), není potřeba aby měl vzhledem ke zbytku světa originální IP adresu. Takže by se adresy mohly v rámci různých privátních sítí opakovat. Musí však být jednoznačné v rámci jedné privátní sítě. Pro používání již obsazených adres se doporučuje využívat k tomu určené adresy, podle kterých směrovače poznají, že se jedná o privátní adresy (obecně se tyto adresy nebudou šířit do světa, ale při špatné konfiguraci přístupového směrovače by mohly). Předejde se tak vážnějším následkům.

## 3.8 Mechanismus NAT a jeho fungování (včetně NAT/PAT)

Mechanismus NAT (Network Adress Translation) slouží k překladu mezi privátními a veřejnými adresami. Překlad probíhá za chodu je využíván na rozhraní mezi privátní sítí a Internetem. Privátní adresy uzlů, které komunikují s Internetem, se překládají na veřejné. Uzly, které s Internetem komunikovat nepotřebují, nemusí mít také přidělenou veřejnou adresu a stačí jim tedy fungování s privátní adresou. To šetří veřejné IP adresy.

Můžeme mít statický NAT, kde jsou pevně dány vztahy mezi vnějšími a privátními adresami, dynamický NAT, kde jsou vztahy určovány dynamicky dle potřeby. Tyto varianty se používají, pokud je poměr vnitřních a vnějších adres 1:1.

Pokud je poměr vnějších a vnitřních 1:N, tak se využívá NAT/PAT (Port Adress Translation), který mapuje všechny vnitřní adresy na jednu vnější a rozlišuje je podle čísel portů. Výhodou tohoto postupu je, že dokáže využít jednu vnější adresu pro relativně hodně uzlů.

Problémem NAT je, že nelze navazovat spojení směrem dovnitř do privátní sítě. Navíc je tu nebezpečí, že pro některé služby a aplikace nebude NAT fungovat vůbec (např. pro IPSEC) protože mohou uchovávat adresy i jinde než v hlavičce. Tomu se snaží zabránit „inteligentní NAT", který podle protokolů mění i adresy v těle IP paketů.

## 3.9 Mechanismus CIDR a jeho přínosy

Jedním z řešení problému úbytku IP adres byl také mechanismus CIDR (Classless InterDomain Routing), o kterém se dá říci, že „nahradil" původní systém tříd IP adres. Zároveň tento mechanismus řeší i problém nárůstu směrovacích tabulek.

Princip této techniky vlastně spočívá v postupu zvaném „supernetting" (je to vlastně inverzní postup k subnettingu). Využívá posouvání bitové hranice v IP adrese směrem k vyšším bitům (odkrajuje síťovou část). Pracuje na principu agregace sousedních síťových IP adres (slučuje adresy se stejným prefixem). Tím se mu daří i zmenšit nárůst směrovacích tabulek, protože dříve musela každá síťová adresa mít záznam v tabulce, teď se záznamy mohou slučovat právě podle prefixu. O detailní směrovací informace (to co je za prefixem) se musí starat jen lokální směrovače.

Důsledky použití tohoto mechanismu byly šetření IP adresami (mohou se přidělovat po menších blocích), redukce objemu směrovacích tabulek a v neposlední řadě také změna způsobu přidělování IP adres. Nově musí adresy přidělovat provideři (ISP), kteří se registrují u regionálních přidělovatelů.

## 3.10 IP adresy v IPv6, srovnání s IPv4

Asi největší změna mezi IPv6 a IPv4 je ve velikosti adresy (128 bitů vs. 32 bitů). Zároveň IPv6 přidává podporu pro QoS, bezpečnost a mobilitu. Přináší také změny v adresování jako jsou

hierarchické členění adresového prostoru, lepší podpora (již povinného) multicastu, který nahrazuje broadcast z IPv4, a zavedení anycastu (přiřazení jedné adresy několika uzlům; ozve se ten co je nejblíž), možnost autokonfigurace a přečíslování.

Důležitá je také zpětná kompatibilita s IPv4. Zařízení IPv6 může komunikovat s IPv4 sítí, ale naopak to nefunguje. K eliminaci tohoto efektu se využívají různé techniky, jako možnost, že zařízení podporuje oba protokoly současně (nazýváme dual-stack), schopnost dual-stack směrovačů překládat mezi oběma verzemi nebo princip tunelování (IPv6 paket se vloží do IPv4 paketu).

## 3.11 Vyhrazené adresy v IPv6, symbolický zápis IPv4 a IPv6 adres.

Symbolický zápis IPv6 adresy se odlišuje od zápisu IPv4. Nově se IPv6 adresy zapisují v hexadecimální soustavě a oddělují se od sebe čtveřice čísel (slova) znakem ':'. Takže taková adresa může být třeba '805B:2D9D:DC28:0000:0000:FC57:D4C8:1FFF'. Jsou i různé varianty zápisu - nulová slova se zkrátí jen na ':0:'; nulová slova se zcela vynechají a nahradí '::'; poslední bity se zapíší ve tvaru IPv4 (trojice decimálních čísel oddělených znakem '.').

Stejně jako u IPv4 jsou i u IPv6 vyhrazené adresy pro speciální účely.

Jako multicast adresy se používají adresy začínající na 'FF'.

Adresy začínající na 'FE' jsou vyhrazeny jako adresy privátní. Ty můžeme dále rozlišit na adresy „site-local", které se přenášejí jen v rámci sítě zákazníka, a „link-local", které se používají jen v rámci daného spoje (segmentu).

Pro loopback je vyhrazena adresa '::1' (resp. '0:0:0:0:0:0:0:1').

Speciální adresou je také '::' (tedy '0:0:0:0:0:0:0:0'), kterou používá uzel, který se teprve dotazuje na to, jaká mu bude přidělena adresa.

Nakonec tu máme ještě vnořené IPv4 adresy, které se do formátu IPv6 pouze rozšíří pomocí vedoucích nul (např. '::196.92.0.1').

# 4. Systém DNS

## 4.1 Soubor hosts a použití symbolických doménových jmen před DNS, plochý prostor doménových jmen

Dříve (v ARPANETu) se k lepší orientaci v adresách používala symbolická doménová jména, která byla vedena v centrální evidenci. Tuto evidenci na požádání distribuovala centrální autorita jako soubor hosts. Evidence představovala plochý prostor doménových jmen, což je prostě seznam jmen přiřazených k adresám bez jakékoliv známky hierarchie. Jména jsou tedy jednorozměrná a přidělují se z předem určené množiny koncovým uzlům.

## 4.2 Vymezení pojmu "doména", Top Level domény, princip delegace pravomoci, pojem zóny

Hierarchický jmenný prostor odráží představu stromovitého zapojení sítí. Tedy jeden jmenný prostor má pod sebou další jmenné prostory a ty zase další, dokud se nedostaneme ke konkrétnímu uzlu. Dílčím jmenným prostorům se říká domény. Pokud vytváříme doménová jména v takovémto jmenném prostoru, vzniká vícerozměrné pojmenování domén a uzlů.

TLD (Top Level Domain) je doména nejvyšší úrovně hierarchického členění. Jsou to domény s předepsaným charakterem, např. ccTLD (country code TLD) je doména příslušící státním útvarům (.cz, .sk, .us) nebo gTLD (generic TLD) jsou domény vyjadřující charakter subjektu (.edu, .com, .gov, .aero).

Princip delegace pravomoci je založen na tom, že každá doména může automaticky vytvořit www doménu a, že v jedné doméně nesmí být jedno jméno použito dvakrát.

Zónou se označuje část hierarchie, kde má právo provádět změny jedna autorita. Typicky se jedná o celý podstrom. Např. autorita nad doménou cuni.cz je zároveň autoritou nad mff.cuni.cz, takže cuni.cz a mff.cuni.cz spadají do jedné zóny. Autorita se však může vzdát dceřinné domény.

## 4.3 Syntaxe doménových jmen, plně kvalifikovaná doménová jména, způsob začlenění prostoru IDN do jmenného prostoru DNS

Syntaxe doménových jmen je taková, že každý „label" (jméno domény či uzlu) smí mít nejvýše 63 znaků. V labelech se smí používat jen písmena bez diakritiky, číslice a pomlčky (ne na krajích). Také se zde nerozlišují velká a malá písmena. Celé doménové jméno pak musí mít nejvýše 255 znaků.

Plně kvalifikovaná doménová jména jsou taková jména, která jsou vyjádřením celé cesty od TLD až do požadované domény či koncového uzlu (např. peterka@ksi.ms.mff.cuni.cz). Ne plně kvalifikovaná jména (např. peterka@ksi) mohou být doplněna, ale jen v rámci aktuální domény (ms.mff.cuni.cz). Výsledek takovéto operace ale není zaručen.

Prostor IDN (podpora národních abeced) bude fungovat jako nadstavba DNS, tedy se nijak systém DNS nezmění. O překlad se starají klientské aplikace.

## 4.4 Vztah name serverů a domén, primární a sekundární name servery, kořenové name servery

Name server je server, který přísluší určité doméně (každá doména musí mít name server). Stará se o převod doménových jmen na IP adresy. Jeden stroj (počítač) může být name serverem pro více domén současně (typicky se pro jednu zónu vyhradí jeden name server).

Kořenový name server (root) zná IP adresy name serverů všech TLD (Top Level Domain).

Primární name server musí mít každá doména. Jedná se o hlavní (master) name server, který má data pro překlad (zone file) přímo na svém místním disku.

Sekundární name server je doporučeno mít alespoň jeden. Musí být umístěn v jiné síti (nejčastěji v nadřazené). Funguje jako záložní name server. Potřebná data (zone file) si nechává zasílat pomocí zone transferu, když usoudí, že je to potřeba.

## 4.5 Princip překladu symbolického doménového jména na IP adresu

Tazatel se nejprve zeptá kořenového name serveru na adresu TLD name serveru (např. cz). Toho se zeptá na plnou adresu (např. ksi.ms.mff.cuni.cz). Přijde odpověď „nevím, ale zkus to u mého dceřiného name serveru cuni, který má adresu ...". Tazatel se dotáže name serveru cuni.cz atd. Všechno končí ve chvíli, kdy přijde odpověď od name serveru ms.mff.cuni.cz „jo, toho znám, to je můj koncový uzel s adresou ...". Nyní se tazatel přímo připojí k uzlu pomocí získané adresy.

## 4.6 Architektura systému DNS (servery a resolvery, optimalizace fungování, autoritativní a neautoritativní odpovědi)

Systém DNS má architekturu typu klient/server. Name server je tu v roli serveru (odpovídá na dotazy) a resolver plní úlohu klienta (pokládá dotazy serveru).

Co se týká optimalizace fungování DNS, je největším přínosem cache-ování (servery si ukládají odpovědi name serverů do svých cache pamětí). Další možností optimalizace je replikace serverů. Name server musí mít primární server a měl by mít alespoň jeden sekundární. Často se replikuje primární server kvůli rozložení zátěže (kořenové name servery mají replikovaný jen primární server). Třetí možností optimalizace je forwarding name server (rekurzivní dotazy neřeší, jen je posílá dál).

Autoritativní odpověď je taková, která pochází od primárního či sekundárního serveru požadované domény.

Neautoritativní odpověď je taková, která pochází z cache paměti. Takovou odpověď může tazatel zahodit a vyžádat si autoritativní, pokud ji potřebuje. Existují dokonce „caching only" name servery, které mají za úkol pouze cache-ovat.

## 4.7 Resource Records v rámci DNS, jejich typy

Resource Record je název záznamu databáze DNS. Nazývá se též věta. Záznam obsahuje název položky, typ, třídu (Internet), TTL (doba, po kterou smí být záznam cache-ován), velikost dat, data.

Jejich typy jsou: SOA (popis zóny), A (IP adresa uzlu), NS (doménové jméno name serveru autoritativního pro danou doménu), CNAME (kanonické synonymum k NAME), HINFO (popis HW a SW), MX (kam má být doručována el. pošta), AAA (IPv6 adresa)...

## 4.8 Protokol DNS, formát zprávy

Protože DNS funguje na principu klient/server, byl vytvořen protokol DNS, který definuje právě komunikaci mezi klientem a serverem. Pro transport DNS paketů se nejčastěji používá UDP.

Formát zprávy (paketu DNS QUERY) je stejný jak pro dotaz, tak pro odpověď. Zpráva se skládá z částí HEADER, QUESTION, ANSWER, AUTHORITY, ADDITIONAL.

## 4.9 Domény a diakritika, systém IDN

Pro používání diakritiky ve jménech domén se využívá systému IDN, který byl vytvořen jako nadstavba pro DNS. Dovoluje použít podmnožinu znaků UNICODE 3.2 a o převod mezi tímto kódováním a kódováním ASCII se starají klientské aplikace. Adresa přeložená na ASCII má pak prefix 'xn--', což je tzv. punnycode (např. adresa kůň.cz by byla přeložena na xn--k-qla0j.cz). Problém je, že pokud uživatel nemá možnost zadávat speciální národní znaky, musí zadávat přímo přeloženou adresu (xn--k-qla0j.cz).

## 4.10 Princip alternativních DNS stromů, přednosti a nevýhody

Alternativní DNS stromy vznikají vytvořením alternativního root DNS serveru. Pod novým serverem si provozovatel může založit vlastní TLD a stavět celý strom domén. Jen je potřeba nějak zajistit, aby se DNS servery uživatelů ptaly tohoto alternativního serveru.

## 4.11 Princip dynamického DNS

Dynamické DNS je typicky placená služba, při které se DNS server aktualizuje podle potřeby, tedy tehdy, když dojde ke změně IP adresy nějakého uzlu. Na něm běží „DNS klient", který podle potřeby informuje právě dynamický DNS server o změně. Statické DNS totiž nejsou připraveny na častější změny IP adres, ke kterým dochází celkem často (např. u ADSL).

## 4.12 Princip překladu IP adresy na doménové jméno, reverzní domény

Držitel IP adresy má ve správě „reverzní doménu". Tou je subdoména domény 'in-addr.arpa'. K ní se přidá právě daná IP adresa v opačném pořadí a vznikne nám reverzní doména. Např. pro adresu 192.10.7.250 je reverzní doména 7.10.192.in-addr.arpa (250.7.10.192.in-addr.arpa je pak jméno uzlu).

## 4.13 Koncept ENUM, jeho princip a možnosti využití

Koncept ENUM je výsledkem snahy o provázání DNS a telefonních čísel. Pochází ze světa spojů a jeho cílem je umožnit, aby se k telefonním číslům daly přiřadit i nějaké další informace (o přesměrování hovoru, o typu koncového zařízení), aby se např. daly posílat e-maily na telefonní čísla nebo aby se telefonní čísla dala používat místo WWW adres (pro WAP).

# 5. Protokol IP et al.

## 5.1 Charakteristika a vlastnosti protokolu IP

Protokol IP je jediným přenosovým protokolem TCP/IP na síťové vrstvě a jako takový je univerzální. K přenosu používá virtuální pakety, kterým se říká IP datagramy. Funguje nespolehlivě a nespojovaně. Dnes používaná verze má číslo 4, ale postupně se přechází na verzi 6. Díky tomu, že je univerzální, tak nabízí jednotné přenosové služby bez ohledu na to, nad čím právě pracuje (zakrývá odlišnosti přenosových technik). Je zaměřen na jednoduchost, efektivitu a rychlost. Přenáší pakety s proměnnou velikostí, kterou určuje odesilatel (to může způsobit fragmentaci).

## 5.2 Formát IP datagramu, význam položek v hlavičce

IP datagram má povolenou velikost od 576 B do 64 KB. Hlavička mívá typicky 20 bytů a její povinné položky jsou:

- VERSION = verze IP protokolu
- LENGTH = velikost hlavičky v násobcích 32 bitů
- TYPE OF SERVICE = dnes ignorováno, původně vyjadřovalo požadavky na QoS
- TOTAL LENGTH = celková délka datagramu v bytech včetně hlavičky
- IDENTIFICATION = při fragmentaci mají dva fragmenty jednoho datagramu stejnou tuto hodnotu
- FLAGS = využívá se při fragmentaci a určuje jestli existují další fragmenty atd.
- FRAGMENTATION OFFSET = offset fragmentu od začátku celku
- TTL = time to live, tedy čítač průchodů přes směrovače (slouží k detekci cyklů)
- PROTOCOL = udává, ke kterému protokolu patří užitečný náklad
- HEADER CHECKSUM = kontrolní součet hlavičky, který se kvůli ttl musí přepočítávat při každém průchodu směrovačem
- INTERNET SOURCE ADRESS = IP adresa odesilatele
- INTERNET DESTINATION ADRESS = IP adresa příjemce

Nepovinné položky hlavičky jsou:

- OPTION FIELD = umožňuje rozšířit hlavičku o další funkce
- PADDING = funguje jako vycpávka - doplnění hlavičky na násobek 4 bytů

## 5.3 Způsob řešení fragmentace v IPv4

O fragmentaci se stará směrovač, který, když zjistí, že MTU sítě, do které má IP datagram předat, je menší než velikost tohoto datagramu, rozdělí jeho náklad (užiteční data) na potřebné části (podle MTU) a ke každé přidá hlavičku, která v mnohých položkách kopií hlavičky původního datagramu. Položku IDENTIFICATION vyplní u všech fragmentů identifikátorem původního datagramu, aby v cíli mohly být fragmenty k sobě přiřazeny. Každému fragmentu (vyjma posledního) nastaví příznak MORE FRAGMENTS v položce FLAGS. A nastaví FRAGMENTATION OFFSET podle toho jak je začátek dat toho kterého fragmentu vzdálen od začátku dat původního datagramu.

O zpětné sestavování fragmentů se stará až koncový příjemce. Pokud nějaký fragment nedorazí, jsou zahozeny všechny. Pokud je to v nějakém dalším směrovači potřeba, fragmentují se zase příliš velké fragmenty. Minimální velikost IP datagramu (576 B) by nikdy neměla být fragmentována. Pokud datagram má nastaven příznak DON'T FRAGMENT, místo fragmentace se zahodí.

## 5.4 Koncepce protokolu ICMP, jeho vztah k IP a zařazení do vrstev

Protokol ICMP (Internet Control Message Protocol) je určen pro potřeby informování o nestandardních situacích (kdy musel být doručovaný paket zahozen). Jedinou výjimkou je situace, kdy byl chybný kontrolní součet hlavičky, pak totiž nelze věřit údaji o odesilateli. Protokol ICMP je součástí protokolu IP, tedy musí být implementován všude tam, kde je i IP, a je s ním vzájemně provázán.

Problém je s přenosem ICMP zpráv. Jedna varianta je taková, že budeme tyto zprávy balit do IP paketů, aby prošly všude tam, kde právě IP pakety. Tady ale nastává problém v tom, že by byly tyto zprávy k dispozici až transportní vrstvě a ne síťové, která je potřebuje (nejčastěji ICMP zprávy generují směrovače a někdy mohou být i příjemci). Pak je možnost využití druhé varianty a sice takové, kdy se budou ICMP zprávy vkládat přímo do linkových rámců a tedy je bude přenášet síťová vrstva (paralelně k IP paketům). Jenže tady nastává ten problém, že ICMP není tak univerzální jako IP a tak by nemuselo fungovat všude tam, kde IP. K vyřešení tohoto dilematu bylo přistoupeno na kompromis. ICMP zprávy se balí do IP paketů, ale jsou brány jako součást síťové vrstvy (tedy dochází k malému porušení vrstevnatého modelu).

## 5.5 Způsob využití protokolu ICMP v utilitách traceroute a ping

Utilita traceroute funguje ke zjištění trasy cesty k nějakému koncovému uzlu. Funguje tak, že odesilatel (uzel, který testuje cestu) odešle cílovému uzlu datagram a TTL nastaví na 1. První příjemce na cestě dekrementuje TTL. Hodnota TTL se dostane na 0 a tak musí datagram zahodit a poslat o tom ICMP zprávu odesilateli. Ten si může poznamenat první bod trasy. Pak pošle datagram s TTL nastaveným na 2... V posledním kroku odesilatel odešle UDP datagram na neexistující port příjemce.

Utilita ping slouží ke jištění dostupnosti vzdáleného stroje. Funguje tak, že odešle několik IP paketů (velikost se určí podle MTU) přímo požadovanému stroji. Ten odpoví zprávou o přijetí, ve které je čas, který byl potřeba k doručení zprávy a TTL. Díky tomu zjistíme, kolik kroků muselo proběhnout, než se paket dostal k cíli a jak dlouho mu to trvalo. Využívá se nejčastěji při diagnostice sítě.

## 5.6 Překlad IP adres na linkové adresy - možné přístupy, protokol ARP a formát jeho zprávy

Protože IP adresy jsou z definice virtuální a nijak neodrážejí linkové adresy, musí existovat převodní mechanismus, který bude zajišťovat získávání linkové adresy z IP adresy a opačně.

Jednou z možností tohoto převodu je tabulkový převod. Prostě, kdo potřebuje přeložit adresu, podívá se do tabulky na její záznam a ví. Problém je však při updatech tabulky když nastanou změny (nastávají relativně často).

Druhou možností je přímý výpočet. To znamená, že zařízení bude mít funkci, která dokáže nějakými operacemi z IP adresy „vypočítat" adresu hardwarovou a opačně. Bohužel lze toto řešení použít jen tam, kde lze nastavit hardwarovou adresu.

Poslední možností je řešit problém pomocí dotazů a odpovědí. Uzel, který chce provést převod pošle dotaz tomu, kdo to umí, a odpovědí je výsledek převodu. Toto řešení má dvě možnosti. Buď se to bude dělat centralizovaně (přes převodní server) a nebo distribuovaně (na dotaz odpoví držitel adresy).

V praxi se nejvíce uchytilo distribuované řešení varianty dotazů a odpovědí. Pro účel použití tohoto postupu byl vytvořen protokol ARP (Adress Resolution Protocol), který definuje právě formát dotazu a odpovědi. Pro opačný převod je definován protkol RARP.

Převod funguje tak, že je dotaz vložen přímo do ethernetového rámce a broadcastem rozeslán všem uzlům. Který uzel se pozná v požadované adrese, odešle odpověď s linkovou adresou. Uzly si získané adresy cachují (jak odesilatel, tak příjemce) a každý uzel, přes který jde odpověď si také vyextrahuje adresy komunikujících uzlů a cachuje je.

ARP zpráva má tři části: hlavičku (HW ADDRESS TYPE = 1 pokud jde o Ethernet; PROTOCOL ADDRESS TYPE = 0x800 pokud jde o IP; HADDR LEN = délka HW adresy; PADDR LEN = délka protokolové adresy; OPERATION = 1 pro dotaz, 2 pro odpověď), část pro odesilatele (SENDER HW ADDRESS, SENDER PROTOCOL ADDRESS), část pro příjemce (TARGET HW ADDRESS, TARGET PROTOCOL ADDRESS). Při dotazu je odesilatelem tazatel a při odpovědi je odesilatelem odpovídající.

## 5.7 Princip IP multicastu, protokol IGMP

Pokud odesíláme paket pomocí IP multicastu, znamená to, že odesíláme paket celé skupině uzlů (příjemců) současně. Tito příjemci mohou být kdekoliv v dosahu IP protokolu. Multicast se odesílá skupině příjemců, která může být buď permanentní nebo dočasná. Funguje tak, že místní uzly (ve stejné síti jako je odesilatel) dostanou paket přímo (prostřednictvím linkového multicastu). O doručení uzlům mimo síť se postará IP multicasting agent, který paket přenese tunelem.

Protokol IGMP slouží ke správě multicastových skupin. Pakety tohoto protokolu se vkládají do IP paketů stejně jako ICMP zprávy. Podle typu IGMP paketu se pozná jestli se jedná o žádost o vytvoření skupiny či odpověď na ni, žádost o přidání uzlu do skupiny či odpověď na ni, žádost o vyjmutí uzlu ze skupiny či odpověď na ni...

## 5.8 Hlavní změny v IPv6 oproti IPv4

Nová verze IP protokolu označená IPv6 zavádí adresy o délce 128 bitů (původní byly 32-bitové). Výraznou změnou prošla také struktura přenášených IP paketů. V předchozí verzi se používala jedna hlavička a měla možnost přidat rozšiřující informace. V nové verzi je jedna základní (a povinná) hlavička a následovat jí mohou další rozšiřující hlavičky (nepovinné). Co se týká struktury samotných hlaviček, je zjednodušena, protože málo používané položky jsou odstraněny (nebo přesunuty do rozšiřujících hlaviček). Byla také odstraněna nutnost přepočítávat kontrolní součet hlavičky v každém směrovači. Dále IPv6 přináší také lepší podporu QoS, směrování, multicast a anycast. Staví se jinak k fragmentaci a přináší podporu zabezpečení.

## 5.9 Způsob řešení fragmentace v IPv6, odlišnosti od IPv4

V IPv6 je minimální přípustná MTU o velikosti 1280 B. Nově není možnost fragmentovat „po cestě". Pokud se přenášený paket nevejde do MTU následující sítě, je zahozen a odesilateli je odeslána ICMP zpráva „Packet too big" (v IPv4 by mohl kterýkoliv směrovač fragmentovat, což ale zvyšuje režii přenosu). Lepší (chytřejší) implementace IPv6 využívají Path MTU Discovery (zjištění minimální MTU po cestě) a pro „slabší" implementace se doporučuje generovat pakety do velikosti 1280 B. Pokud odesilatel fragmentuje, přidává do paketu rozšiřující hlavičku navíc („fragmentační hlavičku").

# 6. IP směrování

## 6.1 Přímé a nepřímé doručování v IP, způsob využití směrovacích tabulek, možnosti jejich aktualizace (dynamické a statické směrování)

Přímé doručování paketu nastane v případě, pokud se odesilatel i příjemce nachází ve stejné síti (mají stejnou síťovou část IP adresy). Bez nutnosti jakékoliv volby směru se odešle paket příjemci. Ještě před odesláním však musí zjistit odesilatel linkovou adresu příjemce. K tomu využije ARP.

Nepřímé doručování nastává v případě, že se příjemce a odesilatel nacházejí v různých sítích. Pak musí odesilatel určit nejvhodnější směr (směrovač, přes který bude nejlepší paket poslat). Ten zjistí ze své směrovací tabulky. Následně odešle paket danému směrovači. Směrovač pak doručuje buď přímo (cílový uzel je v dosahu) a nebo zase nepřímo.

Směrovací tabulka je důležitá při nepřímém směrování. Vybírá se podle ní směrovač, přes který vede nejkratší cesta k cílové síti. Je ovšem nutné nějak směrovací tabulky vytvořit a pak podle potřeb aktualizovat. Můžeme to provést buď pomocí statického směrování (obsah směrovacích tabulek se inicializuje a pak už se nemění) nebo pomocí dynamického směrování (obsah tabulek se mění podle potřeby, např. pokud se k síti připojí nějaký uzel, pokud se odpojí nebo pokud uzel „přejde” mezi sítěmi).

## 6.2 Možnosti optimalizace směrovacích tabulek (zmenšování počtu položek)

Agregace položek - v tabulce najdeme skupinu sítí, které mají síťovou adresu se stejným společným prefixem a jsou všechny ve směru jednoho směrovače. Takové síťové adresy můžeme sloučit do jednoho CIDR bloku (jedné položky tabulky) a směrovat pro celý blok.

Implicitní cesty - pokud máme stromovitou topologii sítě, tak to co nejde synům musí jít předkovi. V tabulce tedy vytvoříme záznam typu „vše ostatní” a dáme ho nakonec. Pak můžeme vymazat záznamy, které směrovaly stejným směrem jako „vše ostatní”.

## 6.3 Základní algoritmus směrování v IP

Pokud se příjemce nachází ve stejné síti jako já (odesilatel), zahaj přímé doručování.
Jinak:
    Prohledávej směrovací tabulku od nejkonkrétnějších záznamů k nejobecnějším.
    Pokud jsi nalezl odpovídající záznam v tabulce, zahaj nepřímé doručování v daném směru.
    Jinak:
        Pokud existuje implicitní cesta, zahaj nepřímé doručování v daném směru.
        Jinak:
            Skonči a vygeneruj ICMP chybu „Destination Unreachable”.

## 6.4 Role hostitelských počítačů při směrování, využití protokolu ICMP při směrování

Směrovače se účastní všech činností souvisejících se směrováním (aktualizace atd.). Oproti tomu hostitelské počítače si jen uchovávají směrovací tabulky a volí počáteční směr, ale neúčastní se aktualizace směrovacích informací. Pokud udělají chybu ve směrování, upozorní je na to směrovač.

Například nově připojený hostitelský počítač A „zná” ve své síti jen jeden směrovač (X). Přes ten

vede cesta do uzlu C. V této síti však existuje i druhý směrovač (Y), přes který je výhodné posílat pakety uzlu B. Protože počítač A chce poslat nějaká data pro B, využije k tomu jediný směrovač, který zná - X. Přes ten sice cesta k B nevede, ale o doručení se postará (pošle data směrovači Y). Pak pošle uzlu A zprávu „ICMP Redirect", která uzel A upozorní na to, že pakety pro uzel B by měl posílat přes směrovač Y. Uzel A si pak může opravit svou směrovací tabulku.

Další možností využití ICMP je zpráva „Router Solicitation", kterou hostitelský počítač vyšle pomocí IP broadcastu, aby zjistil jaké směrovače jsou v této síti dostupné.

Odpovědí na „Router Solicitation" je ICMP zpráva „Router Advertisement". Tu vysílá každý směrovač buď jako odpověď právě na „Router Solicitation" nebo jako pravidelnou zmínku o tom, že se v síti nachází.

## 6.5 Směrování v rozsahu celého Internetu a jeho vývoj v čase

V dávných dobách Internetu „v plenkách" existovaly centrální směrovače, které měly úplné směrovací informace. Internet byl ještě malý, takže to fungovalo.

Pak tento způsob přestal stačit a tak kvůli efektivitě vznikla dvouúrovňová struktura směrování. Základním prvkem byly „core gateway" - centrální směrovače nacházející se v páteřní části Internetu, které znali topologii celého Internetu. Sítě připojené k páteřní síti využívaly „non-core gateway" - směrovače, které znaly jen větev „pod sebou" (směrem od páteře) a vše ostatní posílali přes implicitní cestu do core gateway. O komunikaci core gateways se staral protokol GGP (Gateway-to-Gateway Protocol) a pro komunikaci mezi core a non-core gateways byl vytvořen protokol EGP (Exterior Gateway Protocol).

Pro neúnosnost nárůstu dat o topologii internetu bylo nutné opustit řešení s core gateways. Vznikly autonomní systémy. Internet se vlastně stal propojením těchto systémů. Jednotlivé autonomní systémy navenek nepublikovaly informace o své vnitřní struktuře a mohly si uvnitř definovat vlastní směrovací politiku. Ven z autonomního systému se dostávají jen informace o dostupnosti sítí.

## 6.6 Princip autonomních systémů, EGP a IGP

Autonomní systémy (AS) jsou výsledkem velkého nárůstu směrovacích informací Internetu. Bylo totiž potřeba rozdělit Internet na menší části, tak aby nebylo nutné někde uchovávat veškeré směrovací informace. Autonomní systém je vlastně skupina sítí propojených směrovači v libovolné struktuře (v minulosti byla podmínka na stromovou). Navenek má jen několik málo přístupových bodů a nešíří ven své interní směrovací informace. Pouze okolnímu světu dává informace o tom, které sítě obsahuje. Dnes je také možno využívat peering (každý AS si může sám určit kudy bude komunikovat s jiným AS).

Označení EGP se dříve používalo pro Exterior Gateway Protocol, který sloužil ke komunikaci dvou AS. Dnes se toto označení používá pro Exterior Gateway Protocols, což je soubor protokolů užívaných ke komunikaci mezi dvěma AS. Místo původního protokolu EGP se dnes používá BGP (Border Gateway Protocol).

Jako IGP (Interior Gateway Protocols) je označen soubor protokolů, které slouží k aktualizaci směrovacích informací uvnitř AS. Patří sem například RIP a OSPF.

## 6.7 Protokol RIP a jeho fungování

Protokol RIP (Routing Information Protocol) je ze skupiny IGP. Slouží tedy k aktualizaci směrovacích tabulek v rámci autonomního systému. Je typu „vector distance" (sousední uzly si

vyměňují informace o tom, kolik přeskoků potřebují, aby se dostaly k určitému cíli). Jedná se o dynamický a distribuovaný způsob směrování. Algoritmus na síti běží pořád kvůli aktualizacím a neuvažuje alternativní cesty.

Tento protokol je implementován na aplikační úrovni a využívá protokolu UDP. Jeho výhodou je jednoduchost.

## 6.8 Protokol OSPF a jeho fungování

Protokol OSPF (Opent Shortest Path First) je protokolem ze skupiny IGP a slouží k aktualizaci směrovacích informací v rámci autonomního systému. Algoritmus, který tento protokol využívá, je typu „link state" (každý uzel testuje dostupnost svých sousedů a získané údaje „zapisuje" do link state paketu, ten pak rozesílá broadcastem všem uzlům v síti buď při změně a nebo jinak každých 30 minut). Každý uzel tedy má úplné informace o topologii sítě, ve které se nachází, a může si sám spočítat nejkratší cestu k požadovanému cíli. Výhodou tedy je, že chyba jednoho směrovače se nepropaguje ostatním. Navíc tato metoda dovoluje zjišťovat alternativní cesty a rozdělovat sítě na menší oblasti, které jsou analogické autonomním systémům (=> můžeme zmenšit objem směrovacích informací).

# 7. Transportní protokoly

## 7.1 Dobře známé porty - podstata a význam, kdo udržuje konvenci?

Dobře známé porty (well-known ports) jsou porty na nichž jsou poskytovány služby (FTP, WWW, Telnet, POP3...). Jsou to porty s číslem 0-1023. Význam těmto portům přiděluje mezinárodní orgán IANA a zveřejňuje je na svých webových stránkách (dříve byly zveřejňovány v RFC). Smysl dobře známých portů je v tom, že aplikace se mohou spoléhat na to, že určitá služba bude k dispozici na předem známém portu bez možnosti nečekané změny.

## 7.2 Čím je definováno aplikační spojení? Koncept portů a socketů.

Port je přechodovým bodem mezi aplikační a transportní vrstvou. Tedy entity aplikační vrstvy (procesy, démoni, ...) mohou využívat služby transportní vrstvy prostřednictvím portů. Jedna entita může být asociována s více porty, ale jeden port nemůže být asociován s více entitami.

Socket vznikl v BSD Unixu jako abstrakce souboru. Můžeme na něj nahlížet jako na bránu vedoucí k síťovým službám.

Aplikační spojení (nazveme tak spojení, které vznikne prostřednictvím sítě mezi dvěma aplikacemi) je jednoznačně definováno pěticí (transportní protokol, IP adresa 1, port 1, IP adresa 2, port 2). Díky tomu může být na jeden port na jednom uzlu (třeba port 80 na WWW serveru) vedeno více spojení. Rozliší se pomocí IP adres, pokud se různí uzly vzdáleně připojené, nebo pomocí čísla portů v rámci jednoho připojeného uzlu.

## 7.3 Způsob práce se sockety při nespojované komunikaci (naznačte postup)

Socket je branou k síťovým službám a vzniká nezávisle na portech (ty jsou tu pořád, ale sockety vznikají a zanikají podle potřeby).

Při nespojované komunikaci nejprve vytvoříme socket: mujsocket = **SOCKET(...)**.
Pak musíme socket asociovat s konkrétním portem (např. 80): **BIND(**mujsocket, 80**)**.
Potom můžeme se socketem pracovat. Využíváme k tomu primitivní operace **SENDTO** (odešleme data uzlu zadanému v parametrech) a **RECVFROM** (přijmeme data od vzdáleného uzlu).
Když skončíme naši práci, musíme socket zavřít pomocí **CLOSE(...)**.

## 7.4 Způsob práce se sockety při spojované komunikaci (naznačte postup)

Socket je branou k síťovým službám a vzniká nezávisle na portech (ty jsou tu pořád, ale sockety vznikají a zanikají podle potřeby).

Pro spojovanou komunikaci mají sockety několik operací, pomocí nichž si mohou obě strany posílat data.

Nejprve je potřeba, aby klient i server měli vytvořeny sockety a přiřazeny k portům, kterými budou komunikovat. O to se postarají funkce **SOCKET(...)** a **BIND(...)**. Server pak musí spustit funkci **LISTEN(...)**, která zajistí to, že server bude moci akceptovat klientův pokus o spojení. Klient se pokusí připojit k serveru pomocí **CONNECT(...**, server, **...)**. Pokud server chce klientovi umožnit připojení, vyšle přijetí požadavku pomocí funkce **ACCEPT(...)** a spojení je navázáno. Následně si obě strany mohou posílat data pomocí **SEND(...** data **...)** a přijímat je pomocí **RECV(...** data **...)**. Jedna ze stran pak vyšle požadavek na ukončení spojení (zahodí socket) pomocí **CLOSE(...)**.

## 7.5 Protokol UDP - jeho vlastnosti, formát datagramu, výpočet kontrolního součtu hlavičky

Protokol UDP (User Datagram Protocol) je maximálně jednoduchou nadstavbou nad protokolem IP. Navíc oproti němu poskytuje jen multiplexing a demultiplexing. Funguje nespolehlivě, nespojovaně a komunikace je bezstavová.

UDP datagram se skládá z UDP hlavičky a dat. V hlavičce se nachází port odesilatele, port příjemce, délka UDP datagramu a kontrolní součet.

Kontrolní součet se počítá v jedničkovém doplňku z celého UDP datagramu (tedy i z dat) obohaceného o „pseudohlavičku". To je hlavička, která slouží jen k výpočtu kontrolního součtu. Obsahuje IP adresu příjemce a odesilatele, ID přenosového protokolu a délku UDP datagramu. Díky jedničkovému doplňku má kontrolní součet dvě nuly a tak máme k dispozici jednu speciální hodnotu, která říká, že kontrolní součet nebyl počítán. Použití pseudohlavičky pro výpočet kontrolního součtu má za úkol chránit datagram proti přesměrování (vlivem chyby nebo útoku), protože pokud přesměrujeme datagram v průběhu cesty na jinou IP adresu, nebude sedět v cíli kontrolní součet a datagram bude zahozen (bez generování ICMP zprávy).

## 7.6 Protokol TCP - jeho vlastnosti, způsob navazování a ukončování spojení

Protokol TCP (Transmission Control Protocol) je velmi úspěšný protokol, protože dokáže efektivně poskytovat spojovaný (pouze iluze - sotwarově emulováno) a spolehlivý přenos. Funguje vždy jen na dvoubodovém spojení. Využívá řízení toku a poskytuje ochranu před zahlcením. Používá kontinuální potvrzování, které ale dokáže změnit při riziku zahlcení (jakákoliv ztráta dat je interpretována jako zahlcení). Vůči vyšším vrstvám vytváří iluzi přenosu bytového proudu (přijímá a odevzdává data po bytech). Také dokáže garantovat korektní navázání i ukončení spojení. Používá k tomu 3-fázový handshake, který ošetřuje téměř všechny nestandardní situaci při těchto akcích.

## 7.7 Zajištění spolehlivosti v protokolu TCP

Protokol pro zajištění spolehlivosti využívá kontinuální potvrzování. Pokud dojde ke ztrátě dat, je to vyhodnoceno jako zahlcení příjemce. Ztracený datagram se odešle znovu a pak se vyčká na potvrzení (tedy přejde na jednotlivé potvrzování). Když dorazí v pořádku, pošle dvojnásobek... Postupem času se zase dostane na kontinuální potvrzování.

TCP nečísluje posílané datagramy, ale pozici v bytovém proudu, který přenáší.

Aby se ještě více eliminovala možnost zahlcení příjemce, využívá TCP řízení toku (příjemce v potvrzeních posílá odesilateli volnou kapacitu pro přijímaná data).

## 7.8 Možnosti zajištění QoS na úrovni transportní a aplikační vrstvy

Přenosová infrastruktura dnešního Internetu je relativně laciná a rychlá. To je díky způsobu jejího fungování (přepojování paketů, best effort...). Bohuže nevychází vstříc multimediálním přenosům, které vyžadují QoS. Nejschůdnější cestou pro „zavedeni" QoS na úrovni transportní vrstvy je přístup hrubou silou - zvyšovat přenosovou kapacitu, aby se minimalizovaly výskyty problémů. Dá se však využít i protokolu RTCP. Na úrovni aplikační vrstvy se to dá vyřešit bufferováním na straně klienta, ale např. při IP telefonii nebo videokonferencích se to dá použít jen velmi omezeně (kvůli latenci).

## 7.9 Protokol RTP/RTCP

Určitou podporu QoS bez změny způsobu přenosu nabízejí protokoly RTP a RTCP. Protokol RTP (Real Time Protocol) balí multimediální data do vlastních „paketů" a přidává k nim informace o typu multimediálního obsahu, pořadí paketu, časové razítko a identifikaci streamu. Tento protokol také podporuje multicast. Protokol RTCP (Real Time Control Protocol) slouží spíše jako podpůrný protokol pro RTP. Zprostředkovává informování zdroje multimediálních dat a příjemce o schopnostech příjemce, zpoždění, procentu ztracených paketů atd.

## 7.10 QoS: Differentiated a Integrated Services

Podpora QoS ve smyslu Integrated Services znamená de facto vyhrazení přenosové kapacity pro přenos, který o to požádá (pokud je to v možnostech sítě). Toto řešení odpovídá představě plné podpory QoS. Spojení by muselo fungovat na principu přepojování okruhů. Navíc by musela být podpora už na úrovni síťové vrstvy a to nejen v koncových uzlech, ale i ve vnitřních a páteřních částech sítě.

Princip QoS Differentiated Services spočívá v nalezení kompromisu mezi plnou podporou QoS (jako u Integrated Services) a nulovou podporou. Tímto kompromisem je vytvoření několika tříd provozu. To znamená, že každý přenášený paket se přihlásí k určité třídě a podle toho se k němu po cestě chová síť (může ho upřednostňovat před jinými pakety). V TCP/IP se používá Expedited forwarding (2 třídy) a Assured forwarding (4 třídy, 3 úrovně priorit pro zahození paketu).

# 8. Telnet, FTP a NFS

## 8.1 Možnosti a přístupy ke vzdálenému přihlašování v TCP/IP (rozdíl Telnet vs. rlogin, ICA …)

Telnet je v TCP/IP hlavním (a univerzálním) prostředkem vzdáleného přihlašování. Snaží se být nezávislý na platformě a podporuje terminálové relace i právě mezi různými platformami. Je jednoduchý takže nenabízí například možnost automatického přihlášení a uživatelské rozhraní je pouze znakového charakteru.

Nástroj rlogin pochází z BSD Unixu a je na něj vázán. Umožňuje automatické přihlášení uživatele na vzdáleném počítači díky „trusted hosts".

ICA je řešení od Microsoftu. Podporuje grafické prostředí.

## 8.2 Princip fungování Telnetu, význam a role NVT

Server Telnet je realizován na aplikační úrovni jako úloha a ne jako součást OS. Výhodou takové realizace je snadná modifikovatelnost a nevýhodou jsou neefektivita a nejistota podpory ze strany OS. Pro komunikaci mezi dvěma Telnetovými servery (resp. klientem a serverem) byl vytvořen NVT (Network Virtual Terminal), který sjednocuje právě tuto komunikaci a neohlíží se na konkrétní platformu, na které je Telnet spouštěn. NVT vlastně definuje formát přenášených dat, je minimem, které musí umět všechny implementace Telnetu.

## 8.3 Příkazy Telnetu, mechanismus rozšiřování

Telnet obsahuje také řídící mechanismy. Pro jejich ovládání se používají řídící příkazy, které se uvozují znakem IAC (Interpret As Command) s kódem 255. Příkazy mohou být editační (mazání znaku či řádky), řídící (přerušení procesu, zastavení výstupu, ...) a dohadovací (umožňují oběma stranám dohodnout se na rozšířeních oproti NVT).

Obě komunikující strany mohou mít „lepší implementace" Telnetu a vedle standardního NVT mohou mít možnost pracovat také s nějakými rozšířeními (např. možnost komunikace v plném duplexu, používání vzdáleného echa, různé způsoby využívání řízení toku...). Použití rozšíření je dobrovolné a musí se na něm obě komunikující strany shodnout pomocí příkazů WILL (= „já chci používat rozšíření...") a DO (= „chci abys používal rozšíření..."). Odpovědí na WILL je buď DO nebo DON'T a odpovědí na DO je buď WILL nebo WON'T.

## 8.4 Koncepce protokolu FTP (pohledy na soubor, režimy fungování, uživatelé a práva, srovnání s protokolem TFTP)

Protokol FTP (File Transfer Protocol) zprostředkovává přenos vzdálených souborů skrze protokol TCP. Využívá k tomu dva režimy: textový a binární. V binárním režimu prostě jen přenáší byty tak jak jsou a nestará se o odlišnosti koncových uzlů. V textovém režimu se snaží převádět přenášená data mezi způsoby kódování cíle a zdroje. Data přenáší zásadně jako 8-mi bitové byty a pro text používá stejné kódování jako Telnet. Umí na vzdáleném počítači vykonávat příkazy typu ls, cwd, ...

Soubor chápe FTP jako vnitřně nestrukturovaný (file structure) a přenáší ho jako spojitý proud dat (stream mode). Může však využít i blokový režim přenosu, kdy je mezi jednotlivé přenášené bloky vkládána zarážka, nebo zhuštěný režim přenosu, kdy se používá jednoduchá metoda komprese

eliminací opakujících se znaků. Při přístupu k souborům vystupuje vždy jménem nějakého uživatele. Buď se uživatelé k soukromým datům autentizují pomocí jména a hesla, nebo přistupují k veřejně přístupným datům jako anonymous.

TFTP je „ořezanou" verzí FTP. Využívá UDP a spolehlivost si zajišťuje sám pomocí jednotlivého potvrzování. Nezná pojem uživatele ani aktuálního adresáře. Funguje jen na explicitně zadané adrese.

## 8.5 Implementace a fungování protokolu FTP (spojení a jejich navazování, pasivní režim, druhy klientů)

Protokol FTP vychází z modelu klient/server a je uzpůsoben možnosti takové implementace, která si nárokuje systémové zdroje až v okamžiku jejich potřeby. Zajištění funkcí FTP je rozděleno mezi interpret protokolu a přenosový proces. Používají se pak dvě různá spojení - řídící a datové. Řídící spojení navazuje klient a přežívá po celou dobu relace, datová spojení se zakládají vždy pro přenos jednoho souboru a navazuje je server. V pasivním režimu se klient stará i o navazování datového spojení (kvůli firewallům). Klient je typicky aplikačním programem, který může fungovat na řádkovém principu (plain text) a nebo graficky.

## 8.6 Příkazy a odpovědi protokolu FTP, dialog mezi klientem a serverem

Dialog mezi klientem a serverem probíhá na základě příkazů řídícího jazyka a odpovědí na ně. Pro tyto účely má FTP definován vlastní řídící jazyk, jehož příkazy mají textovou podobu a jsou přenášeny ve stejném tvaru jako příkazy pro Telnet. Rozlišujeme příkazy pro řízení přístupu (např. zadání jména a hesla), nastavení parametrů (např. nastavení režimu přenosu) a výkonné příkazy (např. přenos souborů, přechod mezi adresáři...). Každý příkaz odeslaný klientem vyvolá alespoň jednu odpověď. Ta má číselný charakter a tvoří ji vždy trojmístné číslo (1. = celkový charakter odpovědi; 2. = upřesnění; 3. = bližší specifikace).

## 8.7 Koncepce protokolu NFS, přínosy a důsledky jeho bezestavovosti, význam mount serveru

Nejrozšířenějším protokolem pro transparentní sdílení souborů je NFS (Network File System). Byl původně vyvinut v prostředí Unixu, ale je otevřený a dnes je implementován prakticky na všech platformách. Dokonce připouští, aby klient a server byli na různých platformách.

Funguje jako bezestavový, což výrazně zjednodušuje zajištění korektnosti komunikace např. při výpadku. Zároveň je tento přístup základem robustnosti NFS. Důsledkem bezestavovosti je také to, že operace prováděné mezi klientem a serverem mohou být pouze idempotentní (takové, které lze opakovat vícekrát se stejným efektem). Díky tomu nelze použít některé operace (např. APPEND).

Aby klient získal vstupní bod do sytému souborů serveru, byl vytvořen koncept tzv. mount serveru. Protože NFS je absolutně nezávislý na platformě, nemůže poskytovat přístupovou cestu k nějakému souboru. O to se stará právě mount server. Ten se „namontuje" na adresářovou strukturu a umí řešit další činnosti, které NFS zajistit nedokáže.

## 8.8 RPC - význam a princip fungování, vztah k NFS, způsob implementace

RPC (Remote Procedure Call) vytváří iluzi toho, že vzdálená procedura probíhá lokálně. Pokud tedy lokální program zavolá proceduru na vzdáleném serveru, počká, než se procedura ukončí, tak jak by to udělal s opravdovou lokální procedurou. Klient si tedy ve skutečnosti ani nemusí

uvědomovat, že pracuje na síti. Implementace RPC je řešena samostatně, mimo implementaci NFS.

# 9. Elektronická pošta

## 9.1 Filosofie a architektura SMTP pošty, hlavní protokoly a standardy

Pošta SMTP původně začínala jako jednoduchá pomůcka ve formě „office memo". Postupem času se do ní přidávaly další schopnosti typu formátování textu, podpora národní abecedy či vkládání netextové přílohy. Její architektura vychází z modelu klient/server, kde poštovní server (MTA = Message Transfer Unit) zajišťuje transport zpráv a jejich shromažďování pro aktuálně nedostupné účastníky a poštovní klient (UA = User Agent) vytváří uživatelské rozhraní a umožňuje číst, psát či jinak upravovat zprávy.

Standardy elektronické pošty musí pokrývat přenos zpráv (protokol SMTP), formát zpráv a adres (RFC 822), download zpráv ze schránky na serveru (protokoly POP3, IMAP, ...) a rozšíření zpráv (MIME).

## 9.2 Struktura zprávy el. pošty, hlavička její nejdůležitější položky

Každá zpráva elektronické pošty má jako povinné části hlavičku a tělo a jako volitelnou část má přílohu.

Hlavička obsahuje adresu příjemce (popř. příjemců), adresu odesilatele, datum odeslání, předmět a další atributy (požadavek na potvrzení příjmu atd.). Hlavička je od těla oddělena prázdnou řádkou.

Tělo zprávy obsahuje vlastní text a příloha může obsahovat cokoliv, co je soubor.

## 9.3 Adresy v SMTP poště a doručování podle MX záznamů

Dříve byly poštovní adresy vázány ke konkrétnímu počítači, což velmi komplikovalo přemisťování uživatelů mezi poštovními servery, protože se měnila jeho adresa. Dnes se používají adresy vázané jméno DNS domény a před ním je (oddělen znakem '@') v podstatě libovolný alias.

Protože SMTP přenos má on-line charakter, musí být odesílající server v přímém kontaktu s přijímajícím serverem. Pro zajištění dosažitelnosti přijímajícího serveru se používá tzv. mail spool a MX záznamy. V tabulce jsou zneseny právě MX záznamy o tom, na který mail spool se má doručovat pošta určená nějakému DNS serveru. Zároveň se dá určit více takovýchto serverů a přiřadit jim priority (pokud je první nedostupný, zkusíme druhý...).

## 9.4 Doručování el. pošty s využitím tzv. doménového koše

Doménový koš shromažďuje poštu pro celou doménu (pošta pro xy@abc.cz a uv@abc.cz bude umístěna společně do doménového koše domény abc.cz). Z doménového koše se pak pošta třídí do jednotlivých poštovních schránek uživatelů. K nim se pak mohou připojovat jednotliví uživatelé (držitelé nějakého aliasu) a pomocí POP3 si stahovat poštu. Výhodou je, že doménový koš nemusí být na stejném poštovním serveru jako schránky uživatelů.

## 9.5 Příkazy a odpovědi protokolu SMTP

Dialog mezi dvěma SMTP servery probíhá pomocí příkazů SMTP, které mají textový charakter a odpovědí, které mají číselný charakter (stejné jako u FTP). Nejprve si servery předají identifikační údaje (o příjemci, odesilateli...) a pak teprve dojde k přenosu zprávy, který je ukončen tečkou na

samostatném řádku. Mezi SMTP příkazy patří HELO (zahájení relace), EHLO (výzva k zaslání seznamu podporovaných rozšíření), MAIL (odesilatel), RCPT (příjemce), DATA (signalizuje začátek přenosu dat, končí řádkem s tečkou na začátku), VRFY (ověření existence schránky) a QUIT (ukončení relace).

## 9.6 SMTP dialog mezi poštovními servery

Při dialogu nejdříve servery zahájí relaci pomocí příkazu HELO. Pak odesílající odešle informaci o odesilateli pomocí MAIL. Následuje sekvence příjemců RCPT. Pak se odesílá vlastní zpráva příkazem DATA ukončená tečkou na začátku řádky. Spojení se pak ukončuje pomocí QUIT.

## 9.7 Podstata problému Open Relay v el. poště

Obecně SMTP servery fungují tak, že umožňují přijímat poštu z libovolné sítě od libovolného uživatele a odesílat poštu do libovolné sítě libovolnému uživateli. Toho může být lehce zneužito ke spammingu a proto by se tento postup neměl používat. Správně by měly SMTP servery být omezeny na odesílání pošty z „vlastní" sítě a od „místních" uživatelů a přijímání také jen pro „vlastní" síť a „místní" uživatele.

## 9.8 Problém netextových přenosů v SMTP, příčiny a možnosti jeho řešení

Původní koncepce SMTP počítala s přenosem 7-mi bitových ASCII znaků. Není garantován výsledek při použití 8-mi bitových znaků (např. při používání národní abecedy nebo posílání příloh). Můžeme sice použít konverzi, ale nastává zde nový problém - obě strany (odesilatel a příjemce) se musí dohodnout na stejném způsobu převodu. Jako systematické řešení tohoto problému byl vydán standard MIME (Multipurpose Internet Multimedia Extensions), který řeší jak problém národních abeced (a to i v komentářových částech adres), tak problém příloh (umožňuje i více příloh najednou).

## 9.9 Standard MIME, co řeší a jakým způsobem

Jako řešení problematiky převodu 8-mi bitových dat na 7-mi bitová při SMTP přenosu byl vydán standard MIME (Multipurpose Internet Multimedia Extensions). Ten k převodu používá 2 způsoby - Quoted Printable a Base64. Zároveň zavádí tzv. MIME type, který umožňuje identifikovat data (jejich typ) a podle toho se k nim chovat. Navíc ještě rozšiřuje formát zprávy elektronické pošty o informace související s kódováním příloh apod.

Quted Printable jednoduše konvertuje znaky vyžadující 8-mi bitovou reprezentaci na sekvence znaků 7-mi bitových ('Č' => '=C8').

Base64 kóduje všechny znaky. Nejprve vezme posloupnost jejich bitové reprezentace, tu pak rozdělí na šestice a tím získá posloupnost čísel od 0 do 63. Ty pak podle tabulky kóduje do znaků (7-mi bitových). Např. 'Článek' se zakóduje na 'yGzhbmVr'.

# 10. World Wide Web

## 10.1 Princip hypertextu, historie WWW

Hypertext se snaží zachytit způsob lidského myšlení. Protože člověk nepřemýšlí lineárně, ale přeskakuje mezi myšlenkami, byl vymyšlen hypertext jako text obsahující vodítka (slova či fráze) k dalším textům týkajících se těchto vodítek. Procházení hypertextem nazýváme brouzdání (browsing).

WWW (World Wide Web) byl vyvinut původně v CERNu pro potřeby sdílení informací mezi vědci. Původně byl založen jen jako textová služba. Odtud se pak šířil do zbytku světa a v počátku 90. let si získal velkou popularitu. V roce 1992 se začal vyvíjet první prohlížeč NCSA Mosaic, který byl o rok později vydán jako volně šiřitelný. V roce 1994 vzniká prohlížeč Netcape Navigator a WWW se tak začíná prosazovat i do komerčního světa. Ve stejném roce je také založeno W3C (WWW Consorcium), které se stará o další vývoj a standardizaci WWW. Microsoft vydává Internet Explorer v roce 1995 a v následujícím roce probíhá „válka browserů".

## 10.2 Služba Gopher - koncepce, přednosti a nevýhody, srovnání s WWW

Gopher je takovou alternativou k WWW. Na rozdíl od WWW ale nepoužívá odkazy v textu, ale odděluje nabídku (menu) od textu. Nejprve se uživatel proklikává skrz menu k tomu, co ho zajímá, a pak teprve si může otevřít příslušný soubor (text, obrázek, ...). Položky v menu nemusí reprezentovat pouze místní zdroje, ale i vzdálené (na jiných počítačích po celém Internetu). Komerčně neuspěl v konkurenci WWW.

## 10.3 Koncepce jazyka HTML a jeho vývoj

Jazyk HTML (HyperText Markup Language) je značkovacím jazykem, který zajišťuje formátování WWW stránek. Nijak se nestará o to, jak text vypadá, ale pouze označuje části textu a označuje tak o jaký „druh" textu se jedná (nadpis, tabulka, ...). O výsledný vzhled se pak stará WWW prohlížeč.

Původně obsahoval jazyk HTML jen základní značky označující nadpisy, tučné písmo, kurzívu a podobně. Fungoval jen jako jednosměrná prezentace. Postupem času se do něj zapracovala možnost zpětné vazby (použití formulářů) a zlepšily se také prezentační možnosti (výběr fontu...). Dalšími milníky byly: CSS (Cascading Style Sheets = umožňují samostatně specifikovat vlastnosti zobrazení prvků HTML), skripty (do HTML kódu je možno vkládat výkonný kód, který je interpretován) a Java applety a ActiveX objekty (výkonné programy, ne ve zdrojovém kódu).

## 10.4 Koncepce protokolu HTTP, rozdíly mezi verzemi 0.9, 1.0 a 1.1

Je to protokol určený pro přenos WWW stránek. Data přenáší v textové podobě a využívá k přenosu TCP. Server přijímá požadavky na HTTP přes port 80 (jeden z well-known portů). HTTP funguje bezstavově a pro každý přenášený objekt navazuje spojení zvlášť. Požadavky na server jsou prováděny pomocí jednoduchých příkazů a odpovědi na ně jsou číselné (jako FTP a SMTP).

HTTP 0.9 je jednoduchý přenosový protokol, který neumožňuje širší dialog mezi klientem a serverem. Uměl přenášet pouze hypertextové dokumenty (ne obrázky a podobně).

HTTP 1.0 nabízí pomocí hlaviček více možností o rozšíření dialogu mezi klientem a serverem. Umí přenášet i jiná data než jen hypertext. K identifikaci přenášených dat používá MIME typy. Bohužel

v této verzi přetrvávají kapacitní nedostatky a další nevýhody.

HTTP 1.1 řeší mnohé nedostatky verze 1.0 a navíc přidává ještě další možnosti rozšíření dialogu. Podporuje virtuální WWW servery, možnost využití jednoho TCP spojení pro více přenosů a přináší pipelining („hromadné" odesílání požadavků). Zavádí lepší podporu pro cache a proxy paměti a zlepšuje zabezpečení.

## 10.5 Příkazy, odpovědi a hlavičky HTTP

Příkazy protokolu HTTP jsou GET (požadavek na poskytnutí WWW stránky), HEAD (požadavek na zaslání hlavičky WWW stránky), POST (pošle data na server a současně žádá o zaslání stránky s odpovědí - používá se u odesílání formulářů). Existují ještě příkazy PUT, DELETE, LINK a UNLINK, ale ty se nepoužívají.

Odpovědi na HTTP příkazy jsou číselného charakteru (podobně jako u FTP a SMTP). Jedná se o trojmístná čísla, kde druhá, resp. třetí, číslice upřesňuje význam první, resp. druhé, číslice. První číslice vyjadřuje celkový charakter odpovědi.

Zprávy protokolu HTTP mohou být doplněny o hlavičky, které upřesňují požadavky či odpovědi. Mohou to být obecné hlavičky, hlavičky doplňující dotaz, hlavičky upřesňující odpověď nebo hlavičky popisující tělo zprávy. Například hlavička „Content-Type" určuje MIME typ obsahu zprávy.

## 10.6 HTTP dialog mezi klientem a serverem

Komunikace skrz HTTP mezi klientem a serverem má povahu textových zpráv. Je to podobné jako u SMTP, ale používají se 8-mi bitové znaky. Klient vyšle serveru požadavek a server mu na něj odpoví na dvakrát. Nejprve mu pošle hlavičku požadované stránky, ve které je také zanesena číselná odpověď, a pak mu pošle požadovanou WWW stránku.

## 10.7 Bezstavový charakter HTTP komunikace a možnosti pro uchovávání historie relace

Bezstavovost HTTP komunikace má za následek to, že si server nemůže pamatovat údaje o klientovi (položky nákupního košíku...). Řešení tohoto problému je hned několik.

Informace se mohou vkládat do URL odkazů a tak klient serveru vždy „připomene" co je jeho požadavkem.

Druhým řešením je možnost použití cookies. To jsou malé datové údaje, které vygeneruje server, ale uchovává je u sebe klient. Server odešle v hlavičce odpovědi klientovi cookie, do které zanese všechny údaje potřebné k tomu, aby mohl následně při přijetí dalšího požadavku od tohoto klienta obnovit relaci právě podle cookie, která se k němu zpět přenese v hlavičce požadavku.

Třetím řešením je použití Session ID. Funguje to tak, že server si do své databáze uloží údaje o relaci a klientovi přidělí identifikátor = Session ID. Tím se klient následně prokáže.

## 10.8 Problematika virtuálních WWW serverů a její řešení

V některých situacích je zapotřebí, aby na jednom stroji běželo naráz několik WWW serverů. V takovéto situaci jim říkáme virtuální WWW servery, protože se nejedná o samostatné WWW servery, ale každý z nich má vlastní URL a obsah. Problém nastane, když chceme rozlišit tyto virtuální servery při adresování pomocí IP adresy. Buď můžeme každému virtuálnímu serveru

přiřadit jednu IP adresu (toto řešení se nazývá IP-based) nebo můžeme fyzickému serveru přiřadit jednu IP adresu a jednotlivé virtuální servery identifikuje položka 'host' hlavičky požadavku (toto řešení se nazývá name-based a v HTTP 1.1 je položka 'host' povinná).

## 10.9 Statické, dynamické a aktivní HTML dokumenty

Statický HTML dokument je neměnný a existuje jako soubor na serveru. Při poskytování klientovi muže být transformován například změnou kódování. Také může být poskytnut vyhledávačům k indexaci.

Dynamický HTML dokument sám o sobě neexistuje. Je generován na základě požadavku aplikace. Nemůže být tedy indexován a jeho cachování nemá smysl. Jakmile je však jednou vygenerován, už se nemůže měnit.

Aktivní HTML dokumenty jsou ve své podstatě nedodělané a dotváří se za chodu, např. pomocí skriptů u klienta. Výhodou je, že se aktivní dokument může sám aktualizovat i během svého zobrazení.

## 10.10 Optimalizace fungování WWW, cache servery a jejich řízení

Kvůli snižování zátěže WWW serverů se používají cache a proxy servery. Takže cesta od klienta k WWW serveru je vlastně takový řetězec serverů. Rozlišují se tedy HTTP hlavičky podle typu na „end-to-end” (platí pro celý řetězec) a „hop-to-hop” (platí pouze pro daný přeskok).

Cachování může fungovat na několika úrovních. Sám klient může cachovat, což je pro něj samotného nejefektivnější. Mezipaměť může být také „uprostřed” řetězce ve formě proxy serverů, které cachují data a poskytují je více klientům současně. V neposlední řadě může cachovat také sám WWW server, a to hlavně dynamické stránky.

Důležité je najít správný časový limit pro uchování stránek. Pokud se budou uchovávat moc dlouho, budou neaktuální. Pokud se budou uchovávat příliš krátce, nebude to příliš efektivní.

Pomocí položek hlavičky 'Expires' a 'Cache-Control' určujeme jak dlouho a jestli vůbec se má stránka cachovat.

# 11. VOIP a IP telefonie

## 11.1 Vysvětlete pojmy VOD, VOIP, IP telefonie, internetová telefonie

Zkratka VOD (Voice Over Data) označuje jakoukoliv technologii pro přenos zdigitalizovaného hlasu po datových sítích (VOIP, VOFR, VOATM, ...).

Technologie pro přenos zdigitalizovaného hlasu po protokolu IP se nazývá VOIP (Voice Oer IP). Tato technologie může být realizována různými způsoby (proprietárně jako Skype, dle standardu H.323, ...), může být využita k různým účelům (veřejná služba, technické řešení v páteřních sítích operátorů, privátní služba např. v rámci firmy) a může využívat různou přenosovou infrastrukturu (veřejný Internet, privátní intranet, ...).

IP telefonie je obecné označení pro službu využívající VOIP technologii.

Internetová telefonie je variantou IP telefonie, která využívá veřejného Internetu pro přenos dat.

## 11.2 Architektura H.323, její hlavní části, jejich role a úkoly

Koncept H.323 je doporučení ITU (Mezinárodní telekomunikační unie), které definuje architekturu IP telefonie. Pochází ze světa spojů. Je to robustní řešení, které je ale velmi drahé a komplikované. Obsahuje řadu protokolů a předpokládá infrastrukturu bez podpory QoS. Pokrývá správu terminálů a zóny, kódování hlasu, řízení hovorů, signalizaci a přenos dat.

V IP síti (resp. v jedné zóně IP telefonů propojených IP sítí) je tzv. gatekeeper, který má úlohu správce a řeší tedy administrativní funkce (jako např. vyhledání volaného").

K IP síti je připojena také jednotka MCU (Multipoint Control Unit), která řeší komunikaci více uzlů současně.

K připojení starých (nepodporujících IP) telefonů slouží terminálové adaptéry, které jsou jakousi branou pro tento telefon do IP sítě.

IP síť s klasickou telefonní sítí propojuje voice gateway.

## 11.3 Charakterizujte signalizaci a řízení hovorů v IP telefonii, naznačte jejich řešení v H.323

Signalizace v IP telefonii zřizuje, vede a ukončuje spojení mezi dvěma uzly. Tato činnost zahrnuje překlad adres zúčastněných stran, ověření dostatku přenosové kapacity, hledání cesty pro spojení a vzájemnou identifikaci zúčastněných uzlů. Tomu se v H.323 věnuje hlavně H.225, který definuje formát zpráv pro součásti Q.931 (signalizace převzatá z ISDN) a RAS (komunikace s gatekeeperem).

Řízení hovorů se týká využití spojení dvou uzlů pro potřeby hlasu (eventuálně obrazu). Zahrnuje dohodu o používaných kodecích, dohodu o schopnostech obou zařízení, dohodu o portech pro media streamy a o dalších parametrech přenosu. Řízení hovorů řeší v H.323 hlavně H.245, který může být tunelován skrze H.225.

## 11.4 Popište roli a způsob fungování gatekeeperu v H.323

Funkce gatekeepera je v H.323 nepovinná, ale pokud existuje, musí se u něj všechny terminály

zaregistrovat a využívat jeho služeb. Všechny terminály spadající pod jeden gatekeeper označíme jako jednu zónu. Gatekeeper zajišťuje překlad adres, správu zóny, řízení přístupu, řízení přenosové kapacity. Může se také volitelně starat o signalizaci a řízení hovorů, spravovat přenosové kapacity a autorizovat hovory.

## 11.5 SIP - k čemu slouží, jeho architektura (servery a jejich role)

Protokol SIP (Session Initiation Protocol) je řešení nejen IP telefonie ze světa TCP/IP. Je to vlastně jen signalizační protokol, který se zabývá sestavením a rušením spojení mezi dvěma či více účastníky a dohledem nad používáním tohoto spojení. Nestará se o vlastní přenosy dat a řízení hovorů. Je to jednoduchý textový protokol podobný HTTP, na který navazují další protokoly, které řeší řízení hovoru (nejčastěji SDP = Session Description Protocol).

Architektura SIP počítá s existencí několika serverů se specifickými úlohami. Proxy server (pomáhá hledat spojení), redirect server (říká, kam by se měly přesměrovat žádosti o navázání spojení), location server (zná umístění koncových stanic) a registrar server (terminály se mu přihlašují a sdělují své umístění).

## 11.6 Metody a odpovědi protokolu SIP

V každém SIP terminálu nebo bráně se nachází tzv. User Agent (UA), který obsahuje UAC (User Agent Client) a UAS (User Agent Server).

Styl komunikace mezi UAC a UAS je velmi podobný komunikaci WWW klienta s WWW serverem pomocí HTTP => posílají si požadavky formulované jako metody v textové podobě s hlavičkami, které je upřesňují; opovědi jsou číselné stejně jako u FTP, SMTP a HTTP. Není předepsáno jestli by měla komunikace probíhat přes UDP nebo TCP.

Metody jsou INVITE (žádost o sestavení spojení), ACK (potvrzení iniciátora o přijetí odpovědi na INVITE), BYE (ukončení spojení), CANCEL (ukončení nesestaveného spojení), REGISTER (registrace UA) a OPTIONS (dotaz na schopnosti a možnosti serveru).

Odpovědi mohou mít v tomto případě i číslici 6 na první pozici (označuje globální či nějak zásadní chybu). Jinak jsou stále trojmístné.

## 11.7 Způsob navazování relací v SIP-u, s využitím proxy a redirect serveru

Pro navázání spojení mohou existovat tři možnosti.

Volající UA zná umístění volaného UA. V tomto případě jej osloví přímo a k iniciaci spojení se použije třífázový handshake.

Volající UA nezná volané UA a proto osloví proxy server. Proxy server vyhledá volaného a sám předá žádost o navázání spojení.

Volající UA nezná volané UA a proto osloví redirection server. Ten mu pošle adresu, na které by se měl volaný nacházet.

## 11.8 SIP adresy a jejich překlad, využití při navazování relací

SIP adresy jsou variantami URL adres, kde adresa začíná schématem ('sip'), následuje tečka a pak alias uživatele, zavináč a IP adresa (host) či doména. Pokud je v SIP adrese použita doména, je potřeba ji přeložit. K tomu je využit server registrar.

Když volající nezná IP adresu volaného, zeptá se nejbližšího proxy serveru. Ten si ze záznamu o příslušné doméně zjistí adresu registrar serveru této domény. Zeptá se ho na adresu požadovaného terminálu a pak mu sám pošle INVITE. Redirect server funguje obdobně, jen s tím rozdíle, že získané informace pouze předá tazateli.

## *11.9 Protokol MGCP - jeho koncepce a využití, vztah k SIP a H.323*

Protokol MGCP definuje styl komunikace mezi Media Gateway (zajišťuje přepojování hovorů) a Call Agentem (rozhoduje o směrování hovorů).

Protokoly SIP a H.323 jsou vytvořeny pro komunikaci stylem peer-to-peer, ale MGCP funguje na principu klient/server, kde klientem je Media Gateway a serverem je Call Agent. MGCP „doplňuje" SIP a H.323, nenahrazuje je.