# ISA - příprava na půlsemku/zkoušku

# Kapitola 1 - Architektura sítí, adresování, konfigurace TCP/IP

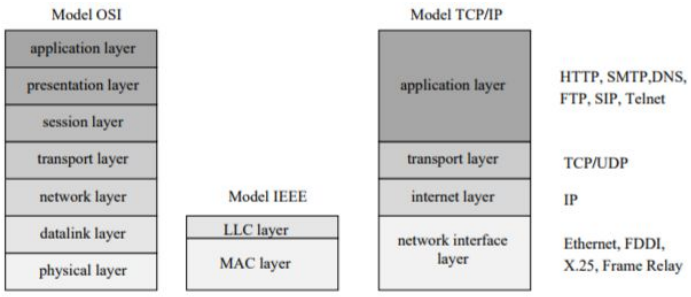
## Přednáška 1

### Architektura sítí

Architektura sítí odlišuje tři základní části komunikačního systému:

- technologie pro přenos signálu (metalická kabeláž, radiové vlny, optická vlákna)
- vrstvu pro zajištění spolehlivého přenosu (TCP/IP, IPX/SPX, AppleTalk, SNA, DECnet)
- aplikační vrstvu, která poskytuje služby uživateli

Je popisována modely, nejpoužívanější jsou ISO/OSI (podle něj má vytvořenou strukturu mnoho dnešních protokolů) a TCP/IP.




Model OSI rozdělen na 7 vrstev, které definují služby dané vrstvy, popisují funkce pro přenos dat mezi procesy stejné vrstvy (partnerská, peer-to-peer komunikace). Na jedné vrstvě není definován jen jeden protokol, ale funkce datové komunikace, které mají protokoly provádět. Při komunikaci daná vrstva využívá služeb nižších vrstev bez znalosti jejich chování či protokolů. Přenos po síti je z pohledu dané vrstvy záležitostí nejbližší vrstvy.

**Síťová vrstva** (network layer) zajišťuje adresování a směrování dat.

**Transportní vrstva** (transport layer) garantuje spolehlivý přenos mezi koncovými uzly. Vrstva implementuje spojované a nespojované transportní protokoly.

**Relační vrstva** (session layer) slouží k udržování relací mezi komunikujícími aplikacemi. Služby zahrnují vytvoření, zrušení relace, obsluhu dialogů, synchronizaci…

**Presentační vrstva** (presentation layer) zajišťuje zobrazení (prezentaci) dat mezi různými aplikacemi a architekturami. Zahrnuje odlišné formáty dat, kompresi dat a kódování.

**Aplikační vrstva** (application layer) definuje uživatelské procesy a aplikace komunikující po síti. Například: elektronická pošta, adresářové služby, virtuální terminál, přenos souborů.

### Architektura TCP/IP

Jednodušší než model OSI. Spojuje služby **presentační** a **relační** do vrstvy **aplikační**. Na úrovni fyzického přenosu bitů spojuje vrstvu **fyzickou** a **linkovou** do vrstvy **fyzického rozhraní**, ta je implementována na **síťové kartě**.

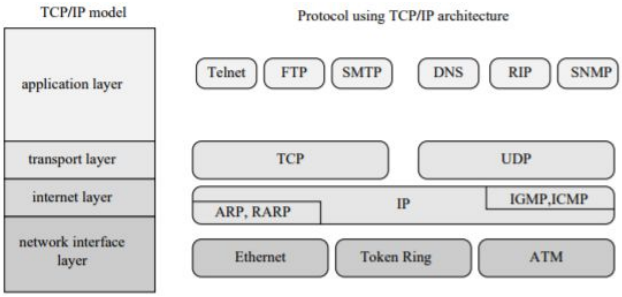
Odstraňuje nedostatky modelu OSI, aneb to co nebylo nikdy plně implementováno.

Implementace architektury TCP/IP rozdělena do **3**částí - **vrstva fyzického rozhraní** je implementována v **síťové kartě**, vyšší vrstvy **internetová** a **transportní** jsou součástí **modulů operačních systémů**.

Při přenosu dat po síti dochází k **zapouzdření** dat vyšších vrstev do jednotek PDU nižších vrstev na straně odesílatele a k **rozbalení** na straně příjemce.

## Vrstvy modelu TCP/IP

Protokolový profil TCP/IP je základem internetu, postaven na modelu o 4 vrstvách.




*Vrstva fyzického rozhraní* (network interface layer, L2) popisuje standardy pro fyzické médium a elektrické signály, zahrnuje funkce pro přístup k fyzickému médiu (ovladače síťových karet). Zajišťuje zapouzdření IP datagramů do rámců.

**Adresování** - používáme tzv. fyzickou adresu (MAC), je to 48-bitová adresa, která se také nazývá hardwarová adresa. Jednoznačně identifikuje síťové rozhraní počítače (kartu NIC, síťovou kartu). První 24 bitů určuje výrobce karty, dalších 24 bitů je číslo karty přidělené výrobcem. Je určena pro adresování na lokální síti. Zařízení na druhé vrstvě si ukládají MAC adresy do přepínacích tabulek (switching tables, MAC tables). Ty obsahují mapování čísla portu na MAC adresu. Dle fyzické adresy může PC zjistit, zda je zpráva adresována jemu. FF:FF:FF:FF:FF:FF je broadcast, zpráva určena všem síťovým zařízením v lokální síti.

*Síťová vrstva* (IP nebo internet layer, L3) vytváří datagramy, adresuje je a směřuje na místo určení. Zajišťuje také doručení s největší úsilím (data poslána nejvhodnější cestou). Ztráta dat => vysílající uzel je o tom informován a musí sám zajistit opětovné přenesení dat. Vrstva kromě protokolu IP používá protokoly ARP, RARP (používají se k mapování IP adres na MAC adresy), ICMP (řízení toku a detekce nedosažitelných uzlů), IGMP (přihlašování do multicast skupin). Tyto protokoly musí být součástí modulu IP v OS. Bez nich nebude komunikace na síťové vrstvě správně pracovat.

Mezi základní operace vrstvy IP patří

- definice datagramu a způsobu adresování
- přenos dat mezi internetovou vrstvou a fyzickým rozhraním
- směrování datagramu na vzdálený počítač

IP vrstva neprovádí kontroly správnosti přenosu ani opravu poškozených dat => nespolehlivý protokol. Kontrola prováděna na vyšších vrstvách modelu TCP/IP.

**Adresování** - k identifikace PC na síťové vrstvě je používá IP adresa. Ve verzi 4 je 32-bitová, u 6 je 128-bitová. Neomezuje se pouze na lokální síť. Kontrolu IP adres provádějí zařízení na L3. Přidělení IP adresy může být dynamické přes DHCP, ale také staticky manuálně v OS. U IPv6 pomocí DHCPv6 nebo pomocí SLAC+RA (router advitersement - router posílá info o prefixu sítě) Každé síťové rozhraní má aspoň jednu IP adresu.

*Transportní vrstva* (transport layer) přenáší data z aplikace na zdrojovém PC do aplikace na cílovém PC. Vytváří logické spojení mezi procesy (na rozdíl od protokolu IP, kterých provádí logické propojení uzlů). Transportní protokoly rozdělují aplikační data na jednotky (pakety), které posílají po síti.

Mezi základní činnosti transportní vrstvy patří

- segmentace aplikačních dat (TCP, UDP)
- posílání dat z jednoho koncového zařízení na druhé koncové zařízení (TCP, UDP)
- ustavení spojení (pouze TCP)

- řízení toku dat mechanismem posuvné okno (sliding window) (pouze TCP)
- spolehlivý přenos pomocí sekvenčních čísel a potvrzování (pouze TCP)

Základními protokoly vrstvy jsou TCP a UDP.

**TCP** načítá data z aplikační vrstvy jako proud dat, seskupuje je do paketů, ty posílá k cíli. Na cílové straně se pakety přeskládávají podle pořadí do datového proudu pomocí sekvenčních čísel, které určují, na které místo v toku dat patří data z paketu.

Zajišťuje spolehlivý přenos dat: řízení toku, řazení paketů, potvrzování, řízení zahlcování. Např. email, vzdálené přihlášení, přenos souborů.

**UDP** umožňuje rychlý přenos paketů bez zajišťování spolehlivého doručení. Implementuje nespojované služby. Zdrojový uzel sám zajišťuje potvrzování a doručení ve správném pořadí. Paket se může ztratit. Je rychlejší než TCP (odpadá režie vytvoření a zrušení spojení, opakovaného doručení ztracených paketů). Používá se u aplikací pro přenos menších objemů dat v paketu s důrazem na rychlost, např. SNMP, DNS (aplikace pro správu a řízení sítě), multimediální přenosy. Při vytížené síti dochází k častým ztrátám, pak by bylo TCP pomalé, protože dynamicky při ztrátách vysílající uzel snižuje rychlost přenosu.

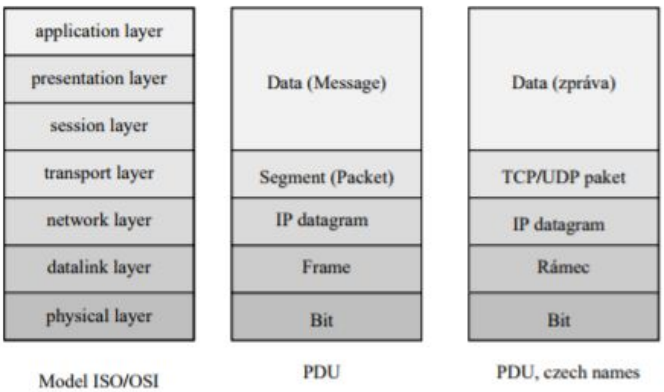
**Programování**

Při programování nemusíme implementovat transportní protokoly, jde nám jen o komunikaci mezi aplikacemi. Využíváme implementované knihovny funkcí, které poskytují přenos TCP/IP. Nejrozšířenější implementací jsou **schránky** (sockets). Jsou to brány, přes které se přenáší data po síti. Schránky jsou jednoznačně identifikované IP adresou a portem. Při komunikaci je potřeba, aby aplikace znaly navzájem jejich IP a port.

**Adresování** - adresujeme služby, ne počítače, které běží na počítačích, např. DNS, emailový server… Službu identifikujeme pomocí 16bitového čísla portu. Jednoznačně identifikuje službu (proces) na PC. Porty jsou rezervované a registrované a dynamické. Rezervované přidělovány organizací IANA standardním službám (HTTP, FTP, …).

*Aplikační vrstva* (application layer) je tvořena procesy a aplikacemi, které komunikují po síti. Zajišťuje zpracování dat na nejvyšší úrovni včetně reprezentace dat, kódování i řízení dialogu. Aplikační protokoly dělíme na uživatelské (vykonávají služby přímo pro uživatele, FTP) a systémové (zajišťují síťové funkce, DNS).

**Adresování** je závislé na konkrétní aplikaci. U elektronické pošty se používají adresy ve formátu user@host, u WWW to jsou URL, HTTP hlavička, SMI hlavička.



# Kapitola 2 - Programování sítí TCP/IP

## Přednáška 2

Díky knihovně BSD sockets nemusíme vytvářet vlastní TCP pakety a zapisovat je na síťovou kartu. To vše zajistí knihovna. Přístup k síťovému připojení je stejný jako k souboru – přes souborové deskriptory (číselné identifikátory souboru či zařízení, které jednoznačně identifikují zařízení pro zápis a čtení).

### Model klient-server, protokol

Je to standardní schéma komunikace mezi dvěma procesy. Komunikace popsána nějakým protokolem. Klient i server jsou aplikační procesy komunikující přes síťové rozhraní (můžou být na stejném PC). Základní činnost klienta je posílání požadavků o nějakou síťovou službu (přijetí a interpretaci odpovědi uživateli), server na tyto požadavky čeká, přijímá je, zpracovává (výhradně on) a posílá zpět odpovědi. Klient obvykle iniciuje komunikaci. Server běží ve smyčce.

**Protokol** je soubor syntaktických a sémantických pravidel určujících výměnu informací mezi nejméně dvěma entitami. Popisuje vytvoření spojení, adresování, přenos dat, řízení toku a zabezpečení. Formálně jej lze popsat stavovými automaty (popisuje všechny scénáře na rozdíl od sekvenčního diagramu), gramatikami, grafovými modely (Petriho sítě, diagramy posloupnosti zpráv) a algebraickými prostředky (komunikační kalkulus CCS, stopy).

Lze navrhnout též vlastní protokol pro vlastní síťovou aplikaci. Výhoda: Přesně implementujeme požadované vlastnosti, aplikace bude rychlá a jednoduchá. Ale pak budeme muset vytvořit klienty a popsat jasně protokol, aby i ostatní uživatelé mohli k aplikaci přistupovat.

### Prostředky pro vytváření síťové komunikace

Základní programovým prostředkem pro vytváření aplikací jsou schránky (sockets). Tvoří aplikační programové rozhraní pro komunikující procesy po síti. Vyvářejí tzv. koncový komunikační bod. Je to nějaká abstraktní datová struktura, která obsahuje údaje pro komunikaci po síti: IP adresu PC a port aplikace.

Pro komunikaci musíme obě schránky propojit na komunikačních uzlech a obě schránky musí znát svou i partnerovu IP adresu a port aplikace. Bez toho se nedají vyplnit údaje do hlaviček IP a TCP/UDP. Tyto informace (IPečka, porty) jsou uloženy v datové struktuře schránka.

Schránky se vytvářejí rozhraní na vrstvě L4 pro přenos aplikačních dat, je identifikována kromě IP, portu také jejím typem (SOCK_STREAM, SOCK_DGRAM, SOCK_RAW)

### Síťový formát pro přenos bitů a bytů

Převod hodnoty na síťový tvar (byte) - htons, htonl, nazpátek - nthons, nthonl.

### Programování komunikace nad TCP

Spojení TCP je perzistentní, přetrvává celou dobu přenosu. Vytvoření TCP spojení pomocí třífázové synchronizace neboli třífázové podávání ruky. Výměna paketů SYN -> SYN+ACK -> ACK. Spojení vyvolává blokující funkce connect() na straně klienta, vysílá paket SYN. Na straně serveru čeká proces na přijetí spojení funkcí accept(). Po 3-way-handshake je ustanoveno spojení. Jeden z hlavních problémů TCP komunikace je blokování. Během toho blokování aktivně proces čeká na potvrzení po síti. Kvůli prodlevě či chybě může dojít ke zpomalení aplikace.

Funkce pro vytvoření spojení: **socket()** - vytvoření datové struktury schránky a její inicializace, funkce vytvoří schránku pro danou rodinu protokolů či doménu, typ komunikace a protokol, při úspěchu vrátí deskriptor schránky, **bind()** - svázání schránky na straně serveru s konkrétním portem (pasivní

otevření), propojí schránku s lokální adresou a portem zadaným jako druhý parametr. Funkce využívána u serverů, pokud je místo adresy obecná adresa INADDR_ANY, OS vybere jakoukoliv platnou IP PC. Užitečné v případě, že PC má více síťových rozhraní, naslouchání na všech pak. U klienta není funkce nutná, při požadavek o komunikaci je mu port automaticky přidělen síťovou knihovnou. Na serveru musíme port zadat, aby klient věděl, na jaký se připojit, **connect()** - aktivní otevření na straně klienta, tato funkce provede připojení klienta TCP k serveru. Vysílá příkaz SYN, **listen()** - pasivní otevření na straně server, čeká na spojení na zadané nepřipojené schránce, OS přijímá požadavky na spojení a směřuje je do této schránky. Tato schránka v jádru systému vytvoří 2 fronty: neúplných spojení (obsahuje záznam pro každý příkaz SYN od klientů, čeká dokončení vytvoření spojení TCP, je ve stavu SYN_RCV), úplných spojení (obsahuje záznam pro každého klienta, který dokončil ustanovení spojení TCP, server je ve stavu ESTABLISHED), **accept()** - přijetí spojení, ustavení komunikace, vybere požadavek z fronty čekajících spojení a vytvoří spojení klient-server. Pokud je fronta prázdná, serverový proces je převeden do stavu sleep, funkce je blokující. Návratová hodnota je deskriptor nové schránky, kterou vytvoří systém pro toto spojení. Původní schránka čeká na další spojení. U všech funkcí musíme testovat návratovou hodnotu, neboť může během komunikace dojít k porušení linky a bez toho nepoznáme, že operace neproběhla úspěšně a budeme čekat na data z nefunkční linky.

Pro přenos dat funkce **read()** a **write()**, **recv()**, **send()**. Poslední dvě umožňují pokročilé nastavení spojení. Je třeba kontrolovat návratové hodnoty!

Ukončení spojení iniciuje klient, z pohledu serveru se jedná o pasivní ukončení, když o to klient požádá, server vyhoví. Používají se funkce **close()** - uzavření schránky, pokud je více referencí na schránku, tak počet sníží o jeden a **shutdown()** okamžité uzavření schránky, uzavírá jen jednu stranu komunikace, close obě. Posílají se čtyři pakety, klient pošle FIN, server pošle ACK, pak FIN a klient zakončí paketem ACK.

### *Iterativní a konkurenční server*

Iterativní zpracovává požadavky postupně a konkurentní je zpracovává souběžně. V praxi spíše konkurentní. Umožňují zpracovat více příchozích požadavků najednou pomocí duplikace procesu serveru funkcí fork() nebo pomocí vláken. Po acceptu se proces zduplikuje, otcovský proces komunikační schránku uzavře, protože ji nepotřebuje a čeká na další příchozí spojení, synovský proces na té schránce komunikuje a přijímá požadavky, schránku pro příchozí spojení uzavře, tam naslouchá otcovský proces.

## Programování komunikace nad UDP

Spojení UDP je jednodušší než TCP, klient nemusí vytvářet spojení se serverem. Vyšle se UDP datagram na server funkce**sendto()**. Ani server neočekává otevření spojení od klienta. Zavolá se jen funkce **recvfrom()**, která čeká, dokud nedojdou data od klienta. Funkce kromě dat vrací adresu klienta, kam se pošle odpověď. Pokud klient nepožádá explicitně o lokální port, je mu přidělen jádrem OS při prvním volání sendto(), zůstává mu i nadále.

4 hlavní výhody UDP: **Není potřeba ustanovit spojení**, TCP užívá 3wh, který je náročný na vytvoření; **Neexistuje stav spojení**, stav spojen vyžaduje paměť pro odesílání a přijímání, parametry pro řízení zahlcení, pro nastavení sekvenčních čísel a čísel pro potvrzování. Server UDP, který to neobsahuje, může obsloužit více klientů než TCP server; **Menší režie v hlavičce paketu**, TCP paket obsahuje 20 bytů informací pro spojení v hlavičce (sekvenční číslo, potvrzení, příznak, velikost posuvného okna, kontrolní součet, volba), UDP jen 8 (délku a kontrolní součet); Rychlejší řízení odesílání dat aplikací, Aplikace připraví data UDP a síťová vrstva je hned odešle. TCP obsahuje mechanizmus řízení zahlcení,

který zpomaluje spojení, pokud je některá linka zahlcena. TCP opakovaně posílá pakety, pokud nedostane potvrzení o jejích doručení.

Pro odesílání dat a přijímání se používá **recvfrom()**, **sendto()**, **recvmsg()**, **sendmsg()**.

Když se jedná o jednoduchý konkurentní UDP server, tak každý požadavek považuje za nový paket a vytvoří nový proces. Po dokončení požadavku se potomci ukončí. Tohle se dá použít, pokud je jeden požadavek pouze v jednom datagramu. Pokud však ve více, tak server pro každého klienta vytvoří nový port a připojí jej k němu, nejedná se o původní rezervovaný, ale o dynamický, klient pak směřuje všechny požadavky na něj.

Nevýhoda UDP komunikace je skutečnost, že recvfrom() je blokující a když se pošle požadavek na neběžící server, tak je klientský proces blokovaný při čekaní na odpověď. Řeší se to connect() funkcí , vytvoří se pomocí ní spojovaná schránka UDP, nemá to nic společného s 3wh. Otestuje se aktuální dostupnost partnera, doplní jeho IP a port a vrátí řízení volajícímu procesu. Pak se ale používají funkce send a write a read a recvmsg místo recvfrom. Nelze zadávat cílovou IP a port při posílání dat, vše je posláno na adresu a port z connectu. Do schránky přicházejí jen data z této adresy a portu. Komunikace omezena jen na jednoho partnera. Schránka přijímá asynchronní chyby na rozdíl od nespojené schránky UDP. Connect je výhodné jen v případě komunikace s jedním partnerem (TFTP).

## Neblokující aplikace

Základní vlastní serveru by mělo být, že při obsluze aktuálního požadavku neodmítne požadavek od nového klienta. Funkce accept(), recvfrom(), read(), write() jsou blokující funkce, které způsobují blokování aplikací. Po zavolání těchto funkcí se předá řízení jádru OS, který čeká na data na síťové kartě určená pro aplikaci. Když data přijdou, jádro je zkopíruje do fronty příchozích dat schránky a předá řízení zpět procesu.

Způsoby předejítí blokování aplikace - nastavit schránku jako neblokující, aby procesu místo čekajícího na data, která nejsou k dispozici, poslal chybovou zprávu EWOULDBLOCK a předal mu řízení. místo převedení do stavu sleep. Pak se opakovaně volá operace čtení ze schránky, tomu se říká polling, je to neblokující, ale spotřebovává procesorový čas. Používá se spíš u connect a accept. Neblokující chování nastavujeme funkcí fnctl. Nastavíme flag schránky O_NONBLOCK.

## Komunikace typu broadcast a multicast

Schránky BSD umožňují zasílat a přijímat data se skupinovým směrováním (multicast) a všesměrovým adresováním (broadcast) na úrovní IP vrstvy. Používají se schránky typu SOCK_DGRAM. Je třeba nastavit speciální vlastnosti schránky. Aneb getsockopt() a setsockopt().

### *Komunikace typu broadcast*

Nastavíme vlastnost schránky SO_BROADCAST u schránky typu SOCK_DGRAM a ta pak posílá pakety UDP na adresu broadcastu. Cílová adresa, na níž posíláme zprávu, musí být typu broadcast. Adresa INADDR_BROADCAST na lokální síti pro broadcast. Broadcast přijímají všichni na lokální doméně. Je to zátěž pro síť. Používá se pro nalezení zdroje, jehož adresu neznáme (DHCP server) nebo zaslání směrovacích informací všem na síti.

Broadcast nad IP je definován pouze pro IPv4 a minimalizuje síťový provoz (jeden paket všem), broadcast IP identifikuje všechny IP rozhraní v podsíti. Broadcast pro síť 147.229.12.0/23 je 147.229.13.255, obecný broadcast je 255.255.255.255 pro jakoukoliv síť. Podporuje pouze UDP transportní protokol. Komunikace omezena jen na lokální doménu. V případě chyby doručení se neposílají ICMP zprávy. IP broadcast vyžaduje podporu na L2, broadcast adresa IP mapována na MAC adresu ff:ff:ff:ff:ff:ff, každé ethernet zařízení tento rámec přijme.

# Přednáška 3

## Komunikace typu multicast

Multicast adresa identifikuje množinu síťových rozhraní. Multicast data se posílají těm účastníkům, kteří o ně mají zájem, ne všem. Multicast není limitován jen na LAN, lze směrovat i na sítích WAN.

### *Adresování*

Multicast přenáší IP datagramy na skupinu PCs, které jsou identifikované IP adresou třídy D (224.0.0.0 - 239.255.255.255). Nižších 28 bitů tvoří identifikaci multicastové skupiny. Pro přenos multicast dat v lokální síti je třeba nastavit mapování multicast IP datagramu nalinkový rámec. Mapování multicast IP adresy na MAC je složena z 01:00:5e (24 bitů), další bit je 0 a zbylých 23 bitů se vezme z multicast IP adresy (nižších 23 bitů). Pět horních bitů z IP adresy se nepoužije, to znamená, že ethernetovou multicast adresu se může namapovat $2^5$-1 multicast IP adres. Dvě skupiny pak můžou mít stejnou MAC adresu a jedna z nich může dostat zprávy té druhé, tomu se říká nedokonalé filtrování, neboť se používá jen část IP adresy v MAC adrese. To se pak předá vyšší IP vrstvě, která porovná celou cílovou IP adresu a nastavenou multicast IP adresu. Tady se už porovnávají všechny bity - dokonalé filtrování. Když je IP adresa paketu jiná, zahodí se.

(IPv6 se mapuje na MAC tak, že první 2 dvojčíslí MAC jsou 33:33, zbytek jsou poslední 4 dvojčíslí z IPv6)

Doručení není spolehlivé. Uživatel se musí do skupiny přihlásit pro doručování zpráv, může se pak kdykoliv odpojit. Členství je dynamické. PC může být ve více MC skupinách, tj. má více MC IP adres. Přihlašování a odhlašování uzlů do MC skupiny zajišťuje protokol IGMP. Každý účastník má právo data přijímat i odesílat všem účastníkům.

### *Programování multicastových aplikací*

Multicast posílá pouze pakety UDP (není garantováno spolehlivé doručení, zachování pořadí). Je teda na úrovni IP podporován jen schránkami skupiny AF_INET typu SOCK_DGRAM nebo SOCK_RAW. Implicitně jsou multicast datagramy vytvářeny s TTL=1, což omezuje doručení mimo lokální síť. Pro doručení mimo se TTL nastaví funkcí setsockopt(). Pro nastavení schránky pro komunikaci MC se používá stejná funkce (IP_MULTICAST_IF, IP_MULTICAST_TTL, IP_MULTICAST_LOOP). Zadání multicast adresy ve struktuře sockaddr_innebo funkci connect(). Posílání datagramů funkcí sendto().

Pro odesílání dat multicast skupině není třeba připojit se ke skupině, to je nezbytné pro čtení. Pro přijímání MC zpráv se používá struktura ip_mreg, která obsahuje info o MC skupině, ke které se připojíme. Čtení dat probíhá za pomocí schránky typu SOCK_DGRAM: inicializace rozhraní pro čtení MC funkcemi socket či bind, připojení do skupiny nastavením vlastnosti IP_ADD_MEMBERSHIP, čtení pomocí recvfrom(), odpojení ze skupiny nastavením vlastnosti IP_DROP_MEMBERSHIP.

## Zpracování paketů na L2

Analýza paketů - zachycení dat v reálném čase a jejich interpretace za účelem zjištění stavu sítě:

- charakteristika síťového provozu
- identifikace špiček během síťového přenosu
- detekce možných útoků a nedovolených aktivit
- nalezení nezabezpečených aplikací

Analýza paketů zahrnuje sběr dat (zachycení binárních dat z přenosového média, např. Ethernet), Převedení binárních dat do srozumitelné formy, rekonstrukce paketů, Analýza paketů, přenosů, konverzací. Nástroje pro analýzu paketů: Wireshark, Microsoft Network Monitor, aplikace tcpdump.

## Komunikace na úrovni IP a na linkové vrstvě

Dosud jen vytváření komunikace na transportní vrstvě. Standardní schránky pracují na úrovni TCP a UDP, kde vytvářejí rozhraní pro aplikace používající pro přenos tyto transportní protokoly. Tyto schránky typu TCP a UDP nezasahují do nižších vrstev(jsou na L4), jen na úrovní IP umožňují nastavit IP. Služby nižších vrstev zajišťuje knihovna BSD sockets s jádrem OS a řadičem síťového rozhraní.

Některé aplikace (týkající se správy sítě, sledování procházejících dat) potřebují přístup k nižším datovým jednotkám, na to použijeme vlastnosti schránek typu raw (L3) nebo knihovnu typu libpcap pro čtení na linkové či síťové vrstvě nebo knihovnu libnet pro vytváření a zápis paketů přímo na síťové či linkové rozhraní (pomocí funkce) nebo k tomu používá libnet schránku typu raw.

### *Schránky typu raw - komunikace na vrstvě IP*

Slouží ke čtení a zápisu dat protokolů ICMP, IGMP, OSPF a IP (mimo TCP či UDP). Využité pro implementaci programů ping a traceroute. Lze vytvářet IP datagramy, které nejsou zpracovány jádrem OS. Přijaté packety jsou defragmentovány jádrem OS před zapsáním do schránky. Zpracovává ICMP pakety kromě typu request, které zpracovává jádro, IGMP a IP, které jádro nezná. Schránky může vytvořit pouze root. Schránka umožňuje vytvořit vlastní hlavičku IPv4 nastavením vlastností schránky IP_HDRINCL. Používá se pro systémové služby, správu sítě, např. ping,traceroute.

### *Přístup k datům na linkové vrstvě*

Nástroje pro zpracování paketů na linkové vrstvě: Paketový filtr BPF (BSD packet filterů), Linuxová implementace schránek typu SOCK_PACKET, PF_PACKET, knihovna Libpcap.

Tyto nástroje zajišťují: Přímý přístup k paketům přicházejícím na síťové rozhraní - zpracování všech paketů na rozhraní (vs. Schránky typu raw); Aplikace pro zpracování na L2 běží jako uživatelský proces (vs. Jádro systému); Aplikace není koncovým bodem komunikace: pracuje pouze s kopií paketu; využívá se u programů typu Wireshark, tcpdump…

*Činnost BPF* - řadič NIC BPF zavolá vždy, když je paket přijat nebo než je odeslán; Aplikace využívající BPF může filtrovat příchozí pakety; Vyfiltrované pakety jsou uloženy v BPF buffery; Když je buffer plný nebo vyprší časovač, data jsou předána aplikaci.

### *Schránky typu PF_PACKET*

Jsou implementovány pouze pro Linux. Slouží ke čtení celého rámce přímo z linkové vrstvy. Pakety jsou uloženy do běžného bufferu schránky (nejsou v kernelu jako u BPF). Podporují filtraci pomocí BPF filtru.

### *Knihovna Libpcap - čtení dat na linkové vrstvě*

Vytváří API pro čtení a zápis dat na síťové rozhraní. Pomocí ní lze přistupovat ke všem paketům na síti i k těm, které jsou určeny pro jinou stanici. Na rozdíl od Libnet neobsahuje rutiny pro vytváření paketů TCP, IP datagramů či linkových rámců. Používá se spíše jen pro čtení dat.

Čtení dat:

- Připojení k síťovému rozhraní: pcap_lookupdev()
- Otevření rozhraní pro čtení: pcap_open_live(), pcap_open_offline()
- Čtení paketů: pcap_dispatch(), pcap_loop(), pcap_next()
- Analýza paketů: provádí aplikace

# Kapitola 3

## Přednáška 4

### Služba DNS

Bez ní bychom nebyli schopni načíst jedinou webovou stránku nebo poslat email. Základním úkolem je **převod doménových jmen na IP adresy**. Pro uživatele není vhodné identifikovat dle IP adresy, lépe se zapamatuje doménové jméno, ale IP adresy jsou zase lepší pro porovnávání nebo prefixové vyhledávání.

Obsahuje DB všech doménových jmen a příslušných IP adres. Tato databáze je distribuována na více počítačů, kde běží speciální server, kterým se říká nameservery, doménové servery nebo **DNS servery**.

Proces vyhledávání v systému DNS nazýváme **rezoluci nebo rozlišení jména**. Využívají ho např.HTTP, FTP, SMTP. První aktivita, co aplikace udělá, když je požádána o připojení na doménové jméno.

Základní služby systému DNS:

- překlad doménových adres na IP adresy (s využitím záznamů typu A, AAAA),
- překlad IP adres na doménové adresy (s využitím záznamů typu PTR),
- překlad aliasů počítačů, překlad na tzv. kanonická jména (s využitím záznamů CNAME),
- určení poštovního serveru pro danou doménu (s využitím záznamů typu MX),
- podpora rozložení zátěže mezi více aplikačních serverů (rotace záznamů),
- sdílení informace v rámci globálního prostoru jmen či
- delegování správy domén na jednotlivé subjekty (pomocí záznamů typu NS).

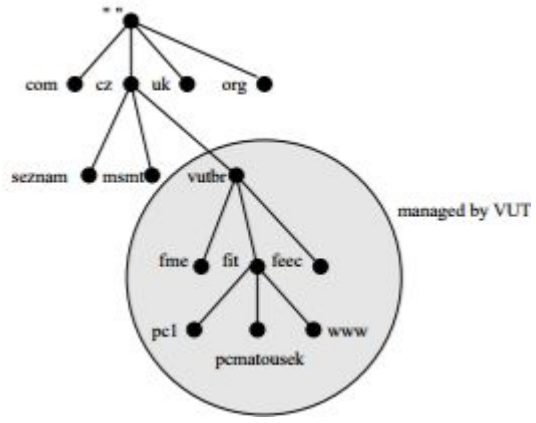
### Architektura systému DNS

Skládá se ze 3 hlavních komponent - **prostoru doménových jmen, serverů DNS a resolveru**.

Prostor zahrnuje strukturu, uspořádání a přístup k datům v systému DNS. Servery ukládají tato data ve svých lokálních databázích. Resolver slouží pro přístup k datům v systému DNS.

#### *Prostor doménových jmen*

Systém DNS tvoří databáze hierarchicky uspořádaná jako **kořenový strom doménových jmen**.




Z pohledu algebry to je acyklický graf - strom obsahující uzly a hrany.

Kořen se nazývá the root, název uzlu max. 63 znaků. Názvy uzlů jsou součástí doménových jmen. Pokud dva uzly mají stejného předchůdce, musí mít rozdílná jména pro rozlišení uzlů. Jméno kořenu je řetězec nulové délky.

**Doména** je podstrom v grafu doménových adres. **Doménové jméno** je cesta mezi uzlem, který tvoří vrchol domény, a kořenem stromu DNS.

Doména s vrcholem v uzlu ve vzdálenosti jedna od uzlu kořene se nazývá **doména první úrovně** atd. Listy stromu můžou označovat konkrétní síťová zařízení patřící do dané domény, např. server pcmatousek, www, pc1. Plné doménové jméno končí ve skutečnosti tečkou.

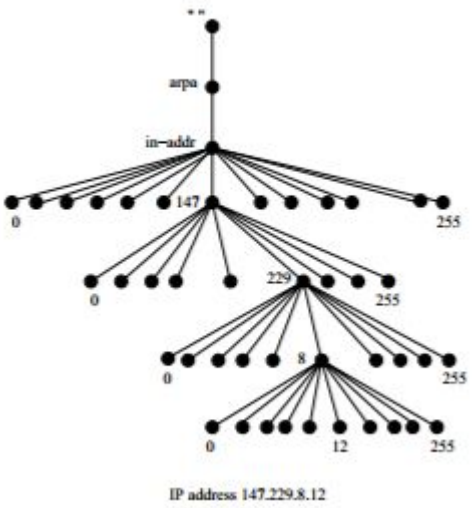
**Správa domén se deleguje** na další organizace, např. doménová jména PC na FITu patří do domény fit.vutbr.cz., kterou zpracuje FIT na svém lokálním serveru DNS kazi.fit.vutbr.cz. Prostě celý strom DNS není na jednom místě, **jednotlivé části** jsou uloženy na **lokálních serverech DNS**. Ty dohromady tvoří **systém DNS**.

Data o objektech v prostoru nejsou jen doménová jména, obsahují informace i o **primárních, sekundárních serverech DNS, správcích domén, poštovních serverech** atd.

Fyzické částí prostoru DNS pod jednou správou = **zóny**. Není totožná s doménou. Někdo může spravovat více domén nebo naopak jen část domény. 2 typy zóny: zóna **stub** obsahuje informace o tom, které servery subdoménu obsahují. Sama neobsahuje žádná data. Zóna **hint** obsahuje seznam kořenových serverů DNS.

### *Reverzní mapování adres*

Další důležitá funkce DNS systému je zpětné mapování IP adres na doménová jména. IP adresy například z logovacích souborů. Použití například při **autorizaci PC**, když kontrolujeme, zda IP adresa má záznam v DNS. Pokud chybí, může to být systémem vyhodnoceno jako použití **podvržené IP adresy** a odmítnout komunikaci. Např. poštovní servery v **boji proti spamu**.




IP adresy uloženy na doméně **in-addr.arpa**. Opět se **deleguje správa subdomén** na vlastníka odpovídajícího prostoru IP adres.

### *Registrace a správa domén*

Koordinaci a správu zajišťuje organizace ICANN. Ta je zodpovědná za správu, přidělování a uložení doménových jmen. Každý záznam o doménové adrese musí být jedinečný, aby příslušné IP adresy byly dohledatelné. ICANN deleguje části prostoru doménových jmen na akreditované registrátory doménových jmen, ICANN spravuje domény nejvyšší úrovně. Domény nižších úrovní jsou delegovány na jiné subjekty.

Národní domény nejvyšší úrovně registruje národní správce domény, který také koordinuje přidělování adres v rámci země. CZ-NIC organizuje .cz a udržuje centrální databázi, do databáze změny zapisují i oprávnění registrátoři.

### *Server DNS (nameserver)*

Jsou další součástí systému DNS. Je to aplikace, která uchovává data z prostoru doménových jmen. Ten prostor je rozdělen do zón a umístěn na jednotlivé servery DNS.

Jejich hlavním úkolem je odpovídat na dotazy směřující na databáze DNS. Uchovávají data ve formě množiny záznamů DNS. Buď jsou záznamy uloženy v lokálním souboru nebo si je server načte z jiného serveru DNS pomocí přenosu zón.

Informace, které server spravuje a je za ně zodpovědný, se nazývají autoritativní.

2 základní typy DNS serverů:

- Primární DNS server - obsahuje úplné záznamy o doménách, které spravuje. Jsou uloženy lokálně v souboru. Server poskytuje autoritativní odpovědi pro tyto domény. Pro každou doménu musí existovat právě jeden primární server DNS.

- Sekundární DNS server - získává data z primárního. Soubor, který obsahuje databázi konkrétní domény, se nazývá zónový. Proces přenosu zónových souborů z primárního na sekundární se nazývá přenos zón. Musí se provádět pravidelně pro aktuálnost dat. Je také autoritativní pro danou doménu. Používá schéma vyzývání pro přenos zón.
- Záložní DNS server - pracuje jako proxy server. Přijímá dotazy od klientů a přeposílá je dalším serverům. Když dostane odpověď na svůj dotaz, uchová si jej a použije jej v budoucnu. Poskytuje neautoritativní odpovědi. Mohou být prostě neaktuální, zrychluje to však proces odpovídání dotazů a rezoluce doménového jména.

Platnost DNS záznamů na sekundárním a záložním serveru je omezená. Hodnota expirace uvedena u každého záznamu. Po expiraci musí být záznam smazán a načten znovu. Bez toho by mohly 2 servery odpovědět jinak, ztráta konzistence dat.

Při vytváření zóny je potřeba vytvořit minimálně dva záznamy: záznam SOA pro specifikaci správy domény a záznam NS, který ukazuje na autoritativní server dané zóny. Do zóny lze přidávat i další záznamy typu A, CNAME, MX, SRV, PTR a podobně.

### *Resolver*

Je klientský program, který se dotazuje na data uložená v systému DNS. Uživatelské programy, které potřebují informace z DNS, přistupují k datům pomocí resolveru.

Základním úkolem resolveru je posílat dotazy DNS serverům, interpretovat jejich odpovědi, předat informace uživatelským programům, které o data žádaly.

Musí přistupovat aspoň k jednomu serveru a ten mu vrátí odpověď, kterou hledal, nebo odkaz na server, který by ji znát mohl.

Resolver si obvykle odpovědi ukládá do cache paměti, totéž dělá i počítač.

### *Rezoluce dotazů DNS*

Je to proces hledání odpovědi v systému DNS. Stačí pouze informace (adresa kořenového serveru DNS) k vyhledání libovolného uzlu stromu, protože prostor doménových jmen je strukturován jako kořenový strom.

Kořenový server DNS je autoritativní server DNS pro všechny domény nejvyšší úrovně TLD. Při dotazu na jakékoliv doménové jméno odpoví buď přímo nebo vrátí odkaz na server DNS, který tu informaci obsahuje. Servery pro domény prvního řádu obsahují informace o příslušných subdoménách druhého řádu atd. je nezastupitelný pro správný běh celého systému DNS. Existuje 13 kořenových serverů, nejedná se obvykle o jeden PC.

2 typy dotazů:

- Rekurzivní - Resolver pošle dotaz na určitý dotaz na určitý údaj ve stromu DNS konkrétnímu serveru DNS. Server odpoví požadovanými daty nebo chybovou hláškou, když nezná odpověď. Pokud server není autoritativním serverem pro daná data, musí se zeptat rekurzivně dalších serverů a čekat. Nebo se může ptát iterativně a získá odkaz na další server, který odpověď zná. Server se ptá tak, jak byl dotázán on. Neptá se nikdy na záznamy NS pro hledanou doménu.
- Iterativní - šetří práci na straně serveru DNS. Server při tomto dotazu vrátí nejlepší odpověď, kterou může dát, více se nedotazuje. Dotazovaný server se podívá do své lokální databáze a když nenajde odpověď, vrátí adresy serverů, které jsou nejblíže hledané adrese.

Většina programů (nslookup, dig) posílá rekurzivní dotazy.

Servery si výsledky dotazů ukládají do cache paměti a příště mohou samy odpovědět na dotaz, ale odpovědi označí jako neautoritativní, je tam riziko, že odpovědi nemusí být aktuální. Negativní záznamy říkají, že ten server nevěděl odpověď, příště se neptám, ty taky nemusí být aktuální, ten server může být doplněn pak o tu odpověď. Proto má každý záznam uvedenu maximální délku platnosti záznamu (TTL), který nastavuje admin zóny pro daný zónový soubor.

Nástroje pro zasílání dotazů: nslookup, dig, host.

## Záznamy DNS

Slouží pro ukládání informací v datovém prostoru DNS. Jsou uloženy v textové podobě v zónových souborech DNS.

Záznam obsahuje jméno (uzlu ve stromu DNS, kde je záznam uložen), typ (záznamu), třídu (záznamu), TTL (maximální dobu platnosti záznamu, pokud je 0, nesmí být uložen v cache paměti, např. SOA), délku dat a data.

### *SOA - start of authority*

Obsahuje informace týkající se uložení autoritativních dat pro danou zónu. Většinou je to první záznam v zónovém souboru. Každá zóna má právě jeden záznam SOA. Záznam SOA obsahuje jméno primárního serveru DNS pro danou doménu (isa.fit.vutbr.cz). Dále obsahuje kontakt na správce domény – jeho emailovou adresu (root@isa.fit.vutbr.cz).

### *NS - name server*

Určuje autoritativní server (či servery) pro danou doménu. Pomocí těchto záznamů dochází k budování hierarchické struktury systému DNS. Záznam NS, který je umístěn v daném uzlu stromu DNS, obsahuje ukazatele na níže připojené následovníky uzlu.

Uvádí autoritativní servery, které obsahují platné záznamy DNS o doméně. Pro zjištění primárního, které jsou primární a sekundární použít SOA.

### *A - IP adrdress*

Obsahuje přímé mapování doménových adres na IP adresy. Je to jediný záznam, který obsahuje na pravé straně IP adresu (spolu s AAAA záznamech). Pokud chceme získat IP adresu například DNS serveru (záznam NS) nebo poštovního serveru (záznam MX), musíme nejdříve získat kanonické jméno obou serverů a teprve poté požádat o A záznam každého z nich.

### *MX - Mail exchanger*

Informuje nás o poštovním serveru, který pro danou doménu přijímá poštu. Oproti ostatním záznamům obsahuje záznam MX navíc prioritu, která určuje preferenci poštovního serveru. Pokud máme pro danou doménu uvedeno více poštovních serverů, použije se ten s nižší hodnotou. Nižší hodnota označuje vyšší prioritu.

### *CNAME - Canonical name*

V DNS můžeme pro jeden počítač vytvořit více doménových jmen. Pokud například na počítači tereza.fit.vutbr.cz poběží služba WWW, můžeme mu přiřadit snadno zapamatovatelný alias www.fit.vutbr.cz. Pokud tam poběží i službaLDAP, přidáme další název ldap.fit.vutbr.cz. Záznamy CNAME pro výše uvedený příklad může vypadat takto: www IN CNAME tereza.fit.vutbr.cz. ldap IN CNAME tereza.fit.vutbr.cz. tereza IN A 147.229.9.22 (Díky A se ke kanonickému jménu přiřadí IP adresu, díky CNAME se k aliasu přiřazuje kanonické jméno)

### *PTR - Domain name pointer*

PTR provádí zpětné (reverzní) mapování. To znamená, že převádí číselnou IP adresu na doménové jméno. Reverzní záznamy jsou uloženy ve speciální doméně in-addr.arpa, a proto jsou uloženy v jiných zónových souborech než přímé.

### *TXT - Text*

Uchovává v DNS textová data týkající se dodatečných informací o doméně, serveru, správci apod. daného uzlu ve stromu DNS. Záznamy TXTse používají pro ověření vlastnictví domény.

### *SRV - Service record*

Slouží pro lokalizaci služeb a serverů. Může se také použít pro distribuce zátěže či zálohování služeb, podobně jako je tomu u záznamu MX. Lokalizace například SIP, XMPP a další.

### *NAPTR - naming authority pointer*

Byl navržen pro mapování řetězců na data. Záznam slouží jako podpora pro dynamicky konfigurovatelné systémy DDDS (Dynamic Delegation Discovery Systems). Lokalizce SIP serveru, umožňuje definovat záložní server pomocí dynamických záznamů.

### *LOC - Location*

Umožňuje administrátorovi zapsat do DNS údaje o umístění počítače, sítě či dalších zdrojů.

### *AAAA . IPv6 Address*

Mapuje doménové adresy na IP adresy verze 6 (IPv6).

## Zabezpečení systému DNS

Služba DNS je veřejná a využívá ji každý uživatel komunikující po internetu. Odpovědi DNS systému jsou většinou přijímány jako důvěryhodné, DNS však komunikuje přes nezabezpečený datový kanál. Odpovědi je možné odchytit, změnit a podvrhnout. Dáse také vnutit serveru DNS informaci do paměti cache, ta ji dale neověřuje a odpovídá uloženou hodnotou bez dotazování autoritativního serveru. Tomu se říká cache poisoning. To vedlo k vytvoření standard na podepisování DNS zpráv - DNSSEC.

### *Bezpečnostní rizika v DNS*

2 důležité úkoly, které je třeba z pohledu bezpečnosti zajistit:

- Integrita dat (tj. Data nejsou změněna během přenosu),
- Autentizace zdroje dat (tj. Odesílateli dat mohu důvěřovat).

K tomu se používá systém veřejných klíčů a podepisování záznamů DNSSEC nebo TSIG.

Základní třídy útoků:

- Odposlech paketů - Útočník sleduje komunikace a v případě dotazu na DNS vrátí nesprávnou odpověď či v odpovědi změní nějaké informace. Tím například přesměruje provoz na server s jinou IP adresou, posílání elektronické pošty. Řešení je zajištění integrity paketů DNS podepisováním záznamů DNSSEC.
- Hádání paketu a predikce odpovědi - identifikační číslo paketu DNS je pouze 16-bitové, číslo cílového serveru je známé, číslo klienta je též 16-bitová hodnota. Existuje $2^{32}$ možných kombinací honot, což je proveditelné pro útok hrubou silou. Sledováním síťového provozu můžeme tyto hodnoty předvídat nebo hádat. Při odchycení dotazu je resolver velmi náchylný k přijetí falešné odpovědi. Relsolver, který kontroluje podpis DNSSEC, pozná neautorizovanou odpověď.

- Zřetězení jmen (otrávení paměti cache) - je podmnožinou útoků typu otrávení vyrovnávací paměti cache. Základem útoku je vložení nesprávné informace do cache. TO lze dosáhnout například změnou informací v poli RDATA odpovědi, zejména v záznamech CNAME, NS, DNAME. Útočník vnutí odpověď na jeho dotaz, která je pozměněná v části RDATA, kde jsou přidány jiná do sekce ADDITIONAL. Například nesprávné IP adresy autoritativních serverů DNS. Zjištění takového útoku je obtížné a většinu takových útoků lze odvrátit kontrolou podpisů pomocí DNSSEC, neboť resolver může ověřit, že odesílatel zná tajný klíč, jehož veřejný klíč je I s ověřením přístupný na veřejném místě DNS.
- Znemožnění služby (Denial of Service, DoS) - systém DNS je zranitelný útokem DoS. Podepisování a autentizace DNSSEC tohle neřeší. Jen to zhoršují, protože je podepisování časově náročné. Řešení je na úrovni konfigurace DNS serveru (omezení počtu dotazů) či vlastní sítě (kontrola počtu navázaných spojení z jedné IP). Jedná se prostě o přetížení velkým počtem dotazů.
- Odmítnutí domény - souvisí to s otázkou ohledně kontrolování neexistence domény. Problém je, zda by měl být resolver schopný detekovat zničení dat útočníkem v odpovědi DNS. Řešení DNSSEC, který obsahuje mechanismus pro určení které autoritativní názvy existují v zóně a které typy autoritativních záznamů existují pro dané názvy pomocí záznamů NSEC a NSEC3.

### *DNSSEC - podepisování záznamů*

Standard definuje rozšíření protokolu DNS pro zabezpečení přenosu dat v systému DNS pomocí asymetrické kryptografie s použitím veřejného a soukromého klíče. Soukromý je pro podepisování a veřejný je pro ověření podpisu.

Používá nové typy záznamů v DNS:

- Záznam DNSKEY pro uložení veřejného klíče (pro ověření podpisů)
- Záznam RRSIG obsahující podpis konkrétního záznamu
- Záznamy NSEC, NSEC3 pro sekvenční uspořádání záznamů v doméně, odkaz na další záznam při dotazu na neexistující doménu
- Záznam DS pro ověření podpisu záznamu DNSKEY (uložen v nadřazené doméně) pomocí vyšší autority

Všechny tyto záznamy se používají pro podepisování zón pomocí DNSSEC. Podepsaná zóna znamená, že zónový soubor obsahuje kromě všech záznamů zóny (A, SOA,…) také elektronický podpis ke každému z těchto záznamů.

Elektronický podpis (podepsaný kontrolní součet (hash) záznamu) se uloží do RRSIG. Použije se k ověření integrity záznamu a autentizaci vlastníka. Podpis se ověřuje oproti veřejnému klíči, ten je v záznamu DNSKEY.

Pro vytvoření podpisu a následné ověření pomocí asymetrické kryptografie potřebujeme vygenerovat klíč. Protože se jedná o asymetrickou kryptografii, používá se dvojice klíčů – soukromý a veřejný klíč.

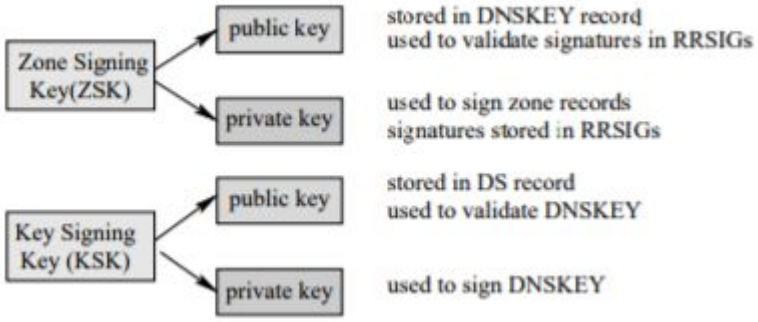
Soukromý klíč slouží k podepisování. Klíč se uchovává na bezpečném místě.

Veřejný klíč slouží k ověření podpisu. Je volně dostupný. U DNSSEC se pravost veřejného klíče ověřuje podepsáním vyšší autoritou pomocí KSK.

Oba klíče jsou algoritmicky závislé - k danému soukromému klíči přísluší konkrétní veřejný klíč. Oba se obvykle generují současně.

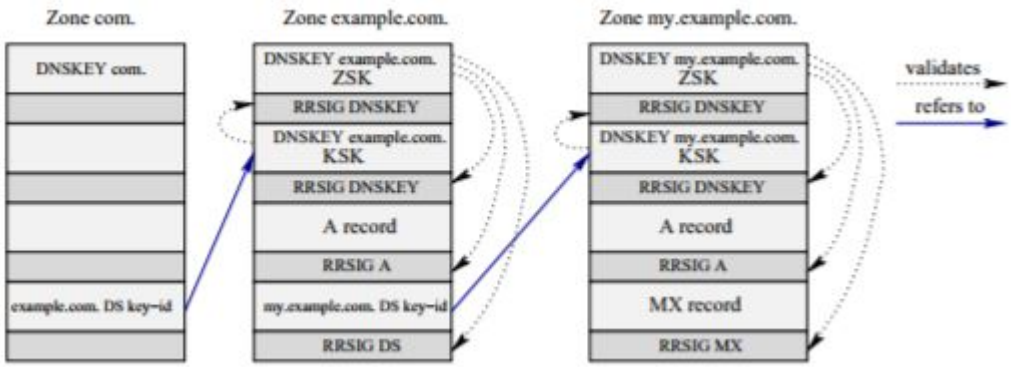
Pomocí dvojice soukromý a veřejný klíč jsme schopni podepsat záznamy v DNS a také zkontrolovat, zda je daný podpis platný. Co však chybí, je potvrzení, že můžeme daným klíčům a podpisům důvěřovat. Co se stane, když si útočník sám vytvoří podvržené záznamy, které podepíše svým klíčem a nabídne nám k ověření svůj platný veřejný klíč? Pak ověříme pravost záznamů a jsme spokojeni. Nicméně jsem získal nepravdivé (neautorizované) údaje.

Z tohoto důvodu nám nestačí jenom klíč pro podpis zóny ZSK (Zone Signing Key). Je potřeba ještě klíč pro ověření těchto klíčů, takzvaný KSK (Key Signing Key). Pro podepisování klíčů opět použijeme asymetrickou kryptografii. Máme tedy dva páry klíčů ZSK a KSK, které se použijí pro vybudování důvěry mezi servery DNS. Jejich vztah a uložení v DNS ukazuje obrázek. Tyto dva páry klíčů, ZSK a KSK, tvoří základ systému zabezpečení DNSSEC. Používají se k podepisování a validaci zón a k podpisování a validaci klíčů pro podpis zón. Klíče KSK (přesněji řečeno veřejný klíč KSK) vytváří tzv. důvěryhodný vstupní bod SEP (Security Enry Point), viz [17]. Společně vytváří propojení KSK a ZSK tzv. řetězec důvěry (chain of trust).




### *Řetězec důvery*

Podepsaná zóna obsahuje veřejný klíč DNS (v záznamu DNSKEY), podpisy záznamů (v záznamech RRSIG) a odkazy na další záznamy (záznamy NSEC), případně záznam DS (měl by být umístěn




v nadřazené zóně) ověřující klíč zóny v DNSKEY. Pokud je zóna neobsahuje, je nepodepsaná.

Záznamy DNSKEY a DS vytváří posloupnost (řetězec) podepsaných záznamů navzájem potvrzující pravost podpisů, tzv. řetězec důvěry. Pravost záznamu DS ověřuje podpis v záznamu RRSIG, který se kontroluje pomocí veřejného klíče uloženého v záznamu DNSKEY dané zóny. Záznam DS obsahuje kontrolní součet (hash) jiného klíče DNSKEY. Tento klíč je tedy autentizován informací v DS. Záznam DS se obvykle nachází v nadřazené zóně, která se nazývá bodem delegace (delegation point). Dvojici souvisejících záznamů DNSKEY a DS nazýváme důvěryhodný pevný bod (trust anchor). Může se stát,

že k dané zóně A neexistuje bod delegace, to jest neexistuje záznam DS v nadřazené zóně, který by obsahoval kontrolní součet záznamu DNSKEY v zóně A. Taková zóna je podepsaná, netvoří však řetězec důvěry. Nazývá se ostrůvkem bezpečnosti (island of security) a podepsané záznamy této zóny bychom měli používat opatrně. Z tohoto popisu vyplývá, že je nutné, aby existoval nějaký počáteční bod řetězce důvěry. Protože je řetězec důvěry budován nad stromem DNS, je vhodné, aby jeho počátečním bodem byla kořenová zóna. V červenci 2010 došlo tedy k podepsání kořenové zóny, čím se vytvořil základní důvěryhodný bod pro vytváření řetězce důvěry (tzv. root trust anchor).

# Kapitola 4

## Přednáška 5

## Elektronická pošta

### *Architektura elektronické pošty*

SMTP - protokol pro přenos emailových zpráv mezi přepravci emailové pošty (MTA).

Základní architekturu elektronické pošty tvoří 2 entity - uživatelský agent UA (User Agent) a agent pro přenos zpráv MTA (message transfer agent). Uživatelský agent je klientská aplikace, která slouží k vytváření, odesílání a čtení elektronické pošty. Agent pro přenos zpráv předává zprávu mezi jednotlivými uzly až na místo doručení. Agent pro přenos může být poštovní server, například postfix.

Používají se také protokoly POP3 a IMAP pro čtení zpráv ze schránek. Pro správné doručení je potřebná služba DNS (záznamy typu MX). Tyto záznamy k dané doméně přiřazují poštovní servery, které pro tuto doménu přijímají elektronickou poštu. Pokud tyto záznamy chybí, odmítne SMTP server poštu doručit a vrátí ji adresátovi.

Při vytváření pošty lze pro vyhledání adresy příjemce použít buď lokální adresář, který bývá součástí emailového klienta, nebo adresářovou službu LDAP. Ta umí podle vlastního jména osoby vyhledat emailovou adresu příslušného uživatele.

Vytvořený email předá poštovní klient protokolem SMTP nejbližšímu SMTP serveru (tzv. server pro odchozí poštu, outgoing SMTP server). SMTP server zprávu přijme a zkontroluje podle emailové adresy, zda je určena pro lokálního uživatele. Pokud ano, uloží ji MDA (mail delivery agent) do schránky příslušného příjemce. Pokud je email určen pro jinou než lokální doménu, zeptá se SMTP server DNS serveru, kdo má být cílovým poštovním serverem (dotaz na záznam typu MX). Poté se přes protokol SMTP připojí s cílovým SMTP serverem a předá mu danou zprávu.

Doručené zprávy se ukládají do schránek s doručenou poštou (tzv. Inbox), do které má přístup pouze uživatele, na jehož adresu byl email poslán. Příchozí poštu spravují servery pro příchozí poštu. Jejich obsah lze získat buď prohlížením souboru z lokálního systému, což není příliš efektivní a praktické, nebo pomocí poštovního klienta přes přístupové protokoly IMAP a POP3.

Pro přenos zpráv slouží MTA. Před vlastním přenosem vytváří odesílající MTA tzv. obálku zprávy (envelope), která nese informace o vysílající aplikaci a počítači. Přijímající MTA zprávu přijme, odstraní obálku a uloží dopis do schránky příchozí pošty daného uživatele. Součástí MTA může být IMAP.

### *Formát zpráv elektronické pošty*

Původní formát – sedmibitový text – už dnes nestačí pro přenos obrazových či multimediálních dat. Proto bylo vytvořeno rozšíření MIME (Multipurpose Internet Mail Extension) [5, 6], které umí strukturovat emailové zprávy tak, aby mohly obsahovat i netextová data, přílohy. Každá emailová

zpráva se skládá ze dvou základních částí – obálky (envelope, vytváří MTA) a zprávy (message, vytváří UA), která je tvořena hlavičkou zprávy (header) a tělem (body) oddělených prázdným řádkem.

Povinné položky hlavičky: datum a čas napsání zprávy (Date:) a adresa odesílatele (From:). Message-ID jednoznačná identifikace zprávy.

### *Standard MIME*

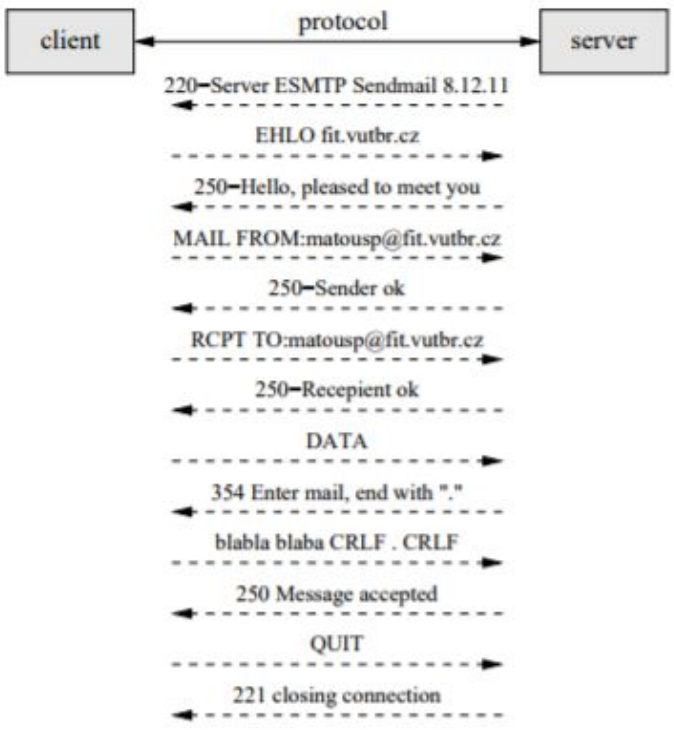
Původně zprávy jen textové a užívalo se 7-bitové kódování ASCII. Uživatelé ale chtěli užívat diakritiku a nelatinské abecedy či binární obrázky. Proto rozšíření MIME. Každou část MIME kóduje různým způsobem. Zpráva může obsahovat textová data, binární digitální podpis a přílohu z Wordu (příklad). Každý typ je zakódován jiným kódováním.

Mezi nejčastěji používaná schemata kódování obsahu patří mechanismus base64 používaný pro netextová binární data a schema quoted-printable doporučované pro textové osmibitové zprávy. Další způsoby kódování jsou uvedeny v tabulce 4.3.

| Typ | Popis |
|---|---|
| 7-bit | 7bitové ASCII znaky s řádky max. 1000 znaků |
| 8-bit | 8bitové ASCII, porušuje definici SMTP (7 bitů) |
| binary | porušuje definici SMTP, není zajištěno správné doručení |
| base64 | kódování 8bitových znaků do 6 bitů (binárních data) |
| quoted-printable | kódování 8bitových ASCII znaků do 7 bitů (textová data) |

### *Přenos zpráv - protokol SMTP*

Protokol SMTP pracuje nad transportním protokolem TCP a využívá port 25. Je to jednoduchý textový protokol pro přenos zpráv. V původním standardu byl definován pro přenos sedmibitových zpráv a i dnes se doporučuje přenášet zprávy v sedmibitovém tvaru. Poštovní server poslouchá na portu 25 a přijímá protokolem SMTP příchozí poštu. Pokud není zpráva určena pro lokální systém, předá ho SMTP server dalšímu SMTP serveru pro cílové doručení. Znamená to, že poštovní server se při přijímání zpráv chová jako SMTP server a při předávání zpráv jako SMTP klient.




Definuje formát příkazů a odpovědí, způsob přenosu.

Komunikace SMTP probíhá anonymně, nepodporuje antentizaci, protože může využít různé SMTP servery dle toho, kde je připojen. Obtížné sdílet heslo. Rozšíření MD5 SASL to pak umožňuje. Chybějící autentizace = spam a podvržené emaily. SMTP by měl u každého spojení určit, zda je autorizované. Pokud to neudělá může být zneužit jako brána pro spammery.

### *POP3*

POP3 načítá data z poštovního serveru a kopíruje je na lokální počítač. Umí pracovat pouze s jedinou schránkou na straně serveru – schránkou pro příchozí poštu INBOX. Při připojení a autentizaci

uživatele umístí POP3 na schránku zámek tak, aby ji mohl využívat pouze jeden klient zároveň. Zprávy přenáší protokol na stranu klienta, kde se zobrazují off-line. Manipulace se zprávami ve schránce (přesuny, mazání) neprobíhají on-line, ale aplikují se až na konci před ukončením komunikace, kdy dojde ke synchronizaci. Nevýhodou tohoto přístupu je, že v případě přerušení spojení se operace nezapíšou správně. Navíc se může stát, že při nekorektním ukončení spojení zůstane na schránce umístěn zámek, který časem expiruje nebo ho musí ručně odstranit správce systému.Protokol neumí pracovat s více schránkami na straně serveru. Neexistuje příkaz na změnu schránky. Přesto umožňují poštovní programy vytvoření více schránek a kopírování příchozích zpráv do nich. Tyto schránky nejsou umístěny na serveru, ale na lokálním systému, kde běží poštovní klient. Pokud si uživatel načte zprávu v práci a uloží na pracovním počítači, nedostane se k ním lokálně například z domácího počítače. Toto je hlavní nevýhoda protokolu POP3 a rozdíl oproti protokolu IMAP, který umožňuje uložení a zpracování došlé pošty na vzdáleném serveru.

#### *IMAP*

Podobně jako protokol POP3 se jedná o aplikační protokol určený ke čtení došlé pošty. Využívá transport TCP, pracuje na portu 143. Protokol IMAP umí pracovat s více schránkami na vzdáleném poštovním serveru. Narozdíl od POP3 umožňuje současnou práci více klientů se schránkami. Pokud se další klient přihlásí ke schránce, označí se jako ”read-only” a pokud chce některý z nich zapisovat do schránky, musí ji znovu otevřít. IMAP umí pracovat nejen z celou zprávou, ale i s jednotlivými položkami z hlavičky dopisu (např. vyhledávání, řazení), čtení části zpráv apod.

Po připojení IMAP klienta k serveru nenačítá klient všechny zprávy v dané schránce jako POP3, ale pouze hlavičky. Teprve když chce ke zprávě přistoupit uživatel, načte i její tělo. Výrazným způsobem tak IMAP minimalizuje datové přenosy mezi poštovním klientem a serverem. Pro off-line zpracování umožňují poštovní klienti načíst vybrané schránky, odpojit se od serveru, pracovat off-line a po dalším připojení synchronizovat schránky.

IMAP server umožňuje ukládat ke zprávám atributy, které definují operace vlastnosti dané zprávy. Mezi základní atributy patří Seen (přečtená zpráva), Answered (odpověď zaslána), Flagged (nastavení příznaků), Deleted (označená zpráva ke zrušení), Draft (návrh zprávy), Recent (nová zpráva) apod. Poštovní klienti umí tyto atributy načíst a zobrazit uživateli, což usnadňuje práci se zprávami.

### Zabezpečení poštovních služeb

Nebezpečí odposlechu přenášených zpráv, změnění či podvržení zprávy. Podobné u protokolů SMTP, POP a IMAP, u nich se hesla a username přenášejí přes protokoly v otevřené textové podobě. U SMTP není autorizace, objevuje se velký problém zneužití služby pro zasílání nevyžádaných nebo komerčních dopisů (spamů).

SSL/TLS na SMTP, SSL/TLS na IMAP, HTTPS, Zabezpečení obsahu PGP a S/MIME, podepisování tím samým.

#### *Požadavky na bezpečnost*

- Důvěryhodnost - zabezpečení proti čtení cizí osobou (odposlech na síti, tcpdump) a podvržení. Zabezpečeno šifrovráním, technika asymetrické kryptografie s veřejným a tajným klíčem.
- Autentizace - aby nešlo poslat email pod libovolnou identitou z libovolného serveru. Nekonzistence mezi adresou odesílatele v hlavičce a skutečnou adresou SMTP serveru, odkud je. Pro zajištění autentizace se používá asymetrická kryptografie, konkrétné elektronický podpis.

- Integrita dat - I když je zpráva chráněna el. podpisem, tak by mohla být cestou změněna, řešení nějaký hash řetězec, který je závislý na každém znaku zprávy.
- Neodtmítnutelnost - ověření odesílatele tak, že nikdo jiný nemohl provést danou operaci. Užívá se k tomu asymetrická kryptografie, tajný klíč, kterým odesílatel podepíše nějakou zprávu, která mu byla předtím doručena v podobě zašifrované veřejným klíčem.
- Dostupnost, kontrola přístupu - omezení dostupnosti na určitý rozsah IP adres, reakce na DoS útoky (generování nesmyslných zpráv na SMTP server), reakce na spamy, podvodné emaily. Pro řešení spamů se užívá antispamová technika, např. blacklist, kontrola reverzních záznamů SMTP serverů.

### *PGP*

Jednou z technik pro zajištění bezpečnosti emailových zpráv je schéma PGP vytvořené roku 1995 Philem Zimmermannem. PGP umožňuje nejenom zprávy šifrovat, ale zajišťuje také autentizaci odesilatele, integritu zprávy i neodmítnutelnost. PGP je soubor programů, které slouží pro šifrování, podepisování, autentizaci zpráv a pro kompresi dat. Ověření cizích klíčů tam provádí na rozdíl od S/MIME uživatel a generuje si sám ty klíče (certifikáty).

### *S/MIME - secure Mime*

Další standardem pro zabezpečení emailových zpráv je rozšíření Secure MIME (S/MIME). Podobně jako PGP poskytuje autentizaci, integritu dat, důvěrnost a neodmítnutelnost. Je to flexibilní technika, která podporuje různé typy kryptografických algoritmů. Využívá přitom standard MIME k zabezpečení různých typů emailových zpráv.

# Kapitola 5

## Adresářové služby

Je to elektronická databáze pro vyhledávání uživatelů (jména, adresy, telefonní čísla, …). Původně jako podpora elektronické pošty.Používá se pro vyhledávání uživatelů, autentizaci a autorizaci, ukládání údajů.

Globální distribuovaný systém s jednotným adresováním. Na mnoha místech už existoval lokální adresář, tak LDAP sloužil k jejich propojení a vytvoření globálního adresáře. Podobně jako DNS má LDAP rovný přístup datům, neupřednostnit žádnou skupinu. Hierarchický přístup k identifikaci a rýchlý přístup k datům.

### *Architektura*

Informace v adresáři se nazývá Directory information Base (DIB). Ten se skládá ze záznamů, které obsahují množinu informací o objektu. Záznamy jsou uspořádány do kořenového stromu Directory Information Tree (DIT), v němž vrcholy jsou záznamy. Záznamy ve vyšší části stromu, které jsou blíže kořeni, reprezentují objekty jako například země nebo organizace, záznamy v nižší části stromu reprezentují osoby či aplikační procesy. Každý záznam má jednoznačný identifikátor distinguished name DN (význačné jméno). DN záznamu se skládá z DN nadřazeného záznamu a hodnoty atributu záznamu. DIB obsahuje 2 třídy - informace o uživatelích a administrativní a operační informace.

### *Jmenný prostor*

Popisuje způsob uložení a uspořádání dat v databázi. Používá se hierarchický způsob uložení dat ve formě kořenového stromu. Záznamy uloženy ve stromové struktuře DIT. Ta podporuje dělení do skupin.

Fyzické uložení dat na serverech je podobné jako u DNS na zóny a data nejsou jen na jednou serveru. Je třeba znát přesnou bázovou adresu vrcholu, odkud začít prohledávání stromu, v případě dotazu.

## Adresářová služba (/ systém) LDAP

Je to internetový protokol pro přístup k adresářovým službám, název LDAP označuje celou adresářovou službu, nejen přenosový protokol. Je to alternativa k X.500, ten měl architekturu moc náročnou na implementaci a i běh klienta a serveru. LDAP je jeho odlehčená verze, zjednodušuje architekturu celého systému a minimalizuje požadavky na klienta a server. X.500 používal víc protokolů, LDAP používá jen LDAP protokol. Používá koncept X.500 pro uspořádání dat: Directory Information Tree (DIT)

Formálně se celý popis adresáře LDAP včetně komunikace, uložení dat a zabezpečení popisuje pomocí tzv. modelů.

Systém LDAP je definován pomocí 4 modelů:

- Informační model popisuje uložení informací v adresáři. Definuje typy dat, které můžeme vkládat, způsob jejich uložení a operace, které nad daty můžeme provádět (porovnání, vyhledání podřetězce). Informační model popisuje záznamy (např. záznam o studentovi na škole), atributy záznamu (např. jméno, příjmení, ročník) a hodnoty (konkrétní položky). Tyto informace pro konkrétní adresář tvoří tzv. adresářové schéma.
- Jmenný model se dívá na uložená data a definuje, jak jsou hierarchicky uspořádaná a jak se na ně odkazovat. Popisuje strukturu adresáře, kterou budujeme z jednotlivých záznamů. Při pojmenování používá jednoznačné jméno nazývané Distinguished Name, DN (význačné jméno), které svou strukturou připomíná DNS adresu.
- Funkční model se zabývá přístupem k datům. Popisuje operace, které se provádějí prostřednictvím příkazů LDAP protokolu. Základní operací nad adresářem je vyhledávání (search). Funkční model například definuje, v jaké části adresáře vyhledáváme (celý strom, podstrom, plochá struktura), kde začínáme hledat, podle jakých atribut a další.
- Bezpečnostní model se zabývá zabezpečením informací uložených v adresáři. Definuje způsob přístupu k datům (autorizovaný či neautorizovaný přístup), práva k vkládání či modifikaci dat a autentizační metody (MD5 SASL, TLS).

Pří návrhu a správě AS se musí admin zabývat těmito oblastmi.

## Informační model

Definuje typy dat a základní jednotky pro uložení informací, tzv. **záznamy**. Ty obsahují informace o objektech reálného světa, které jsou uloženy v adresáři. Je tvořen množinou **atributů**, které obsahují informace o objektu, který záznam reprezentuje. Každý atribut má **typ** a **hodnotu**. Atribut může mít více hodnot. Třída objektů (např. person) popisuje **záznamy** a **objekty**. Třída objektů mimo to také kontroluje operace nad adresářem (způsob porovnání), upravuje umístění objektu ve stromu DIT (hierarchii), definuje, které atributy může nebo musí záznam obsahovat a přiřazuje pro skupiny záznamů způsob přístupu (kontrolu přístupu). Třídy mohou dědit jako v OOP. Nejvyšší generická třída se jmenuje top.

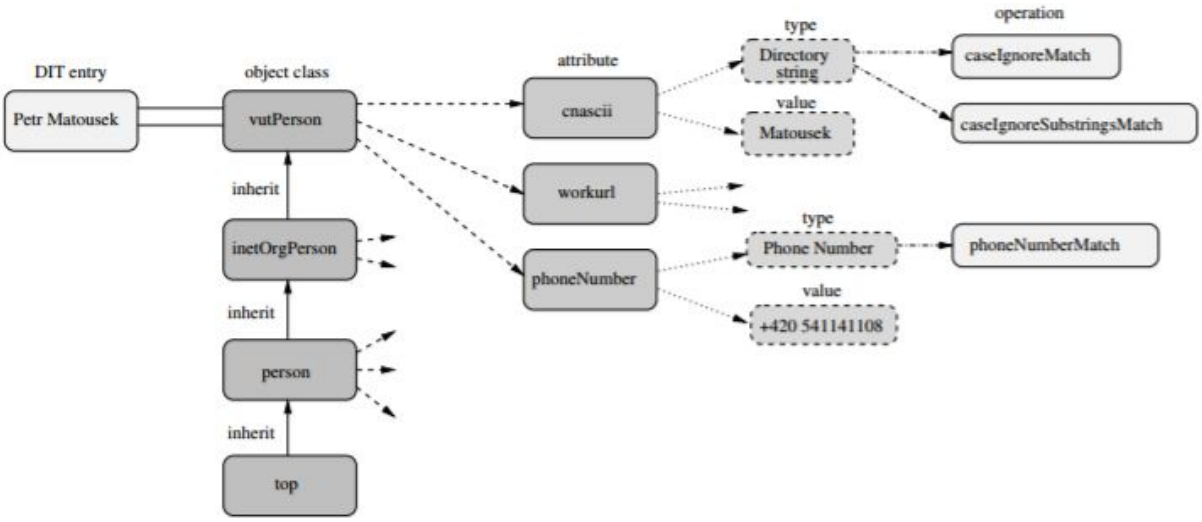
### *Jednoznačné jméno*

Každý záznam má své relativní jméno, které se vztahuje k svému bezprostřednímu předchůdci (rodiči) ve stromové struktuře DIT. Tomu se říká relativní jednoznačné jméno (RDN… relative). RDN je složeno z neuspořádané množiny 1 či více atributů záznamu (např. ou= FIT UIFS či cn=Petr Matoušek+ou=FIT UIFS). Plné kvalifikovaní jméno = RDN a DN svého rodiče, např. uid=matousp, ou=FIT UIFS, dc=FIT, dc=CZ. Toto jméno jednoznačně odkazuje na záznam ve stromu DIT.

### *Adresářové schéma*

Je množina pravidel, které určují, co bude uloženo v adresářové službě a jak mají klient a server zacházet s informacemi při operacích.

Obsahuje definici třídy objektů pro daný záznam (typ třídy, seznam atributů), definici atributů pro danou třídu (typ atributu, operace) a definici pravidel pro operace nad atributy (syntax, odkaz na metodu).




Standardní třídy: person, organization, locality, device, country, applicationProcess, dcObject, groupOfNames, uidObject, organizationalPerson/Role/Unit

## Jmenný model

Popisuje způsob uspořádání dat (organizaci) a vztahy mezi záznamy, identifikaci záznamů v adresáři. Pracuje jen se záznamy, ukládá je do stromové struktury DIT. V něm vrcholy tvoří záznamy adresáře a hrana mezi vrcholy definuje vztahy mezi záznamy. Hrana mezi X a Y znamená, že záznam X je bezprostřední předchůdce záznamu Y. Rodič a potomek. Uzel může obsahovat data i potomky, idetifikuje se dle DN, které tvoří posloupnost relativních DN od uzlu ke kořeni stromu, např. dn: uid=matousp, dc=fit, dc=vutbr, dc=cz.

Speciální záznam typu **alias** - alternativní jméno záznamu odkazující se na jiný záznam (uzel). Například cn=bar, dc=example, dc=com odkazuje na cn=foo, dc=widget, dc=com tím, že obsahuje aliasedObjectName: cn=foo, dc=widget, dc=com. Náročné operace přístupu a vyhodnocení záznamu typu alias = lépe se aliasu vyhnout.

Kromě aliasu je také objekt typu odkaz, je to třída referral s atributem ref. Odkaz typu refferals - speciální URL typu ldap. Např. ref: ldap:// ldap.desy.de/ou=DESY,dc=HEP,dc=NET

Odkaz lze zpracovat na straně klienta (referral followed by client) nebo serveru (zřetězení).

## Funkční model

Popisuje operace, které lze provádět nad adresářem, zejména dotazy (prohledávání adresáře), změny dat (přidávání, aktualizace, rušení záznamu) a řízení přístupu k datům (identifikace klienta).

Popisuje příkazy protokolu LDAP a způsob a rozsahy vyhledávání informací v adresářovém stromě.

Adresář LDAP je klasická síťová služba, která je postavená na výměně zpráv mezi LDAP klientem a LDAP serverem. LDAP klient vytvoří zprávu (dotaz), kterou pošle protokolem LDAP serveru. LDAP server provede akci nad adresářem a vrátí odpověď, která může obsahovat jeden či více záznamů.

Server může odpovědět jednou či více zprávami obsahující odpověď. Veškerá komunikace protokolem LDAP je implementována nad transportním protokolem TCP, port 389. Data uložena ve formátu ASN.1, přenos kódován standardem BER.

Základní operace protokolu LDAP:

- Bind/Unbind: vytvoření/ukončení spojení (autentizace uživatele)
- Compare: porovnání hodnoty atributu u zadaných záznamů
- Abandon: zrušení předchozí operace
- Modify: změna konkrétního záznamu
- Add: přidání nového záznamu
- Delete: zrušení záznamu (v listu)
- ModifyDN: změna nejlevější komponenty v DN

### Použití adresářových služeb

- Vyhledávání e-mailové adresy u poštovního klienta, volajícího účastníka v IP telefonii
- Autentizace uživatelů (Web, unixové přihlášení)
- Autentizace 802.1x (přístup do sítě LAN)

## Přednáška 6

### Videokonference

Multimédia jsou kombinace forem obsahu - současné použití více než jednoho média.

#### *Distribuce multimediálního obsahu*

3 základní rysy:

- Způsob přenosu - počet příjemců dat - unicast (jeden příjemce), multicast (určená skupina přijímajících), broadcast (doručováno všem uživatelům)
- Klient-server (jeden objekt zvolen jako server řídící přístup k mediálním objektům) / peer-to-peer (decentralizovaný přístup k objektům)
- Streaming (live, on-demand) / download

### Streaming / download

Běžně uživáno ve smyslu jednosměrný přenos multimediálních dat.

Streaming multimédií

- Přenos multimediálních dat po síti v reálném čase rychlostí přehrávání
- Vhodný mechanismus pro jednorázové doručení/zobrazení
- Přijímacímu stačí malá vyrovnávací paměť
- Přehrávání začíná po načtení několika prvních sekund (mnohdy i méně)
- Živý přenos (live streaming) - všem přehrávačům jsou zasílána ve stejný okamžik stejná multimediální data, přehrávání nelze řídit.
- Streaming na vyžádání (on-demand streaming) - klientovi je streamem zasílán předem vytvoření multimediální obsah, přehrávání lze řídit

Download (stažení souboru)

- Přenos a uložení kompletního souboru
- Libovolná rychlost přenosu dat
- Přehrávání začíná až po kompletním uložení dat

Přehrávání souboru s přehráváním

- Kombinace streamování a přenosu souborů
- Soubor není na začítku přehrávání kompletně uložen u příjemce
- Stahování vyšší rychlostí než přehrávání
- Přehrávání může začít chvíli po začátku stahování
- Možnost kontrolovat přehrávání
- Je třeba použít větší vyrovnávací paměť než při streamingu

### *Media streaming*

Může to být využito u IPTV, Cable TV, internetových rádií, video on demand (youtube), zabezpečení objektů (IP kamery), videokonference, přenos AV signálů (na projektory a monitory).

### Problémy

Co nám může způsobit problémy při sledování streamovaných médií - omezená šířka přenosového pásma, rozptyl paketů (nepravidelné doručování paketů), zatížení sítě, zpoždění, ztráza paketů, nekompatibilita protokolů, formátů dat.

Částečné řešení problémů - využití vyrovnávací paměti (po cestě mezi serverem a klientem, přímo u klienta, před začátkem přehrávání jetřeba počkat na načtení dat do vyrovnávací paměti).

### Komprese

Obvykle se u streamování multimediální data komprimují. Bez použití komprese by byl objem přenášených dat příliš velký. Před odesláním audia a videa po síti je nutné provést digitalizaci a kompresi (redukce množství přenášených dat).

Standardy u audia - G.711, G.726. G.728, G.729, MP3, WMA, AAC; u videa - H.261, H.263, MPEG-1, MPEG-2.

### *MPEG Transport Stream*

Standard, který popisuje, jakým způsobem jsou části multimediálního obsahu kombinovány do jednoho datového toku - multiplexing. Umožňuje v jednom datovém toku odesílat jeden i více multimediálních streamů. Vhodné pro přenosy, při nichž je možné, že dojde ke ztrátě paketu nebo poškození dat v paketu a/nebo kde je potřeba posílat více než jeden program současně. Data jsou rozdělena na pakety pevné velikosti. Jednotlivé toky s médii se slučují do jednoho toku pomocí multiplexeru a k tomu jsou přidána další data sloužící pro usnadnění výběru programu uživateli (info o tom kanálu - jazyk, titulky atd.).

MPEG TS se skládá ze sekvence transportních paketů o pevné délce obvykle 188 bytů. Každý MPEG-TS paket obsahuje 184 vytů dat a 4 bytovou hlavičku. První byte hlavičky obsahuje vždy hodnotu 0x47, což je sync byte. Klíčovou položkou je 13 bitových packet identifier (PID), která odpovídá jednotlivým elementárním streamům.

Pro přijímaný transport stream musí uživatel určit, o které konkrétní pakety transport streamu má zájem nebo-li musí určit PID paketů. Pro usnadnění výběru PID jsou jako součást TS posílána metadata s informacemi o programu - PSI (program specific information).

### *RTP a RTCP*

Zmíněno u hlasových služeb (další kapitola).

### *RTSP (real-time streaming protocol)*

Je to tzv. signalizační protokol, který má 2 základní funkce - slouží k navázání a ukončení spojení a k řízení jednoho nebo více časově synchronizovaných média streamů (funguje jako síťový dálkový ovladač pro multimediální servery). Je určený pro VOD a multimedia multicasting a broadcasting. Textový protokol, syntaxe je podobná http. Out-of-band protokol - data jsou doručována jiným protokolem (RTP), stateful protokol - protokol si udržuje informaci o stavu relace (ID relace, sekvenční číslo).

Typy zpráv - DESCRIBE - žádost o zaslání popisu médií příslušných k URL (popsáno SDP), SETUP - požadavek na konkrétní objekt (média stream), odpověď obsahuje seznam parametrů nutných pro přenos, PLAY - příkaz pro server aby začal zasílat data specifikovaná zprávami SETUP, PAUSE - dočasně zastaví zasílání konkrétního streamu, TEARDOWN - zastaví zasílání konkrétního streamu, OPTION - používání pro nestandardní žádosti klienta.

### *Videokonference*

Obousměrný přenos synchronizovaného audia, videa a případně dalších médií mezi dvěmi nebo více fyzicky oddělenými lokalitami v reálném čase napodobující situaci, kdy účastnící jednají tváří v tvář.

Z pohledu počítačových sítí je média streaming přenos dat po počítačové síti v reálném čase rychlostí přehrávání, vhodný mechanismus pro jednorázové doručení/zobrazení. Tudíž i videokonference patří mezi streamovaná média.

Na kvalitu přenosu má vliv zpoždění (pro videokonference musí být co nejmenší, pro streaming může být až několik desítek sekund), ztrátovost (občasná ztráta dat způsobí občasnou poruchu při přehrávání videa/audia, v současné době toto dokáží systémy vyrovnat), rozptyl paketů (použití bufferů).

Výhody videokonference - úspora financí, času, vícemožnosti než telefování, možnost komunikovat s místy s omezeným přístupem. Příklady využití – virtuální setkání, výuka, virtuální spolupráce, …

### *H.323*

Doporučení ITU-T popisující terminály a další součásti, které poskytují služby pro multimediální komunikaci v paketově orientovaných sítích, v nichž nemusí být garantovaná kvalita služeb.

Specifikuje součásti vzájemně spolupracující v H.323 prostředí. Ty jsou terminál, gateway, gatekeeper a multipoint control unit (MCU).

Terminál - koncové zařízení, zajišťuje obousměrnou komunikaci v reálném čase s jiným koncovým zařízením, bránou (gateway) nebo MCU. Součásti terminálu - videokamera, zobrazovací plocha, mikrofon, reproduktory, kodek, uživatelské rozhraní.

MCU - umožňuje pořádat vícebodové videokonference, přijímá a přeposílá streamy, přijímá a dekóduje příchozí streamy v různých formátech a vytváří příslušný výstupní stream pro koncové uzly, streamy mohou být zasílány asymetricky. Continuos presence - zobrazení 2 a více účastníků konference současně, Voice-activated switched mode - obraz zasílán v závislosti na hlasitosti příchozího audia na MCU.

Gateway - koncové zařízení v síti, které obykle umožňuje obousměrnou komunikaci v reálném čase mezi H.323 terminály na paketově orientované sítí a jinými ITU terminály na síti s přepínáním okruhů.

Gatekeeper - volitelná součást H.323 architektury, poskytuje služby pro překlad čísel koncových uzlů na IP adresy, řízení velikosti datových toků, řízení přístupu a komunikace s ostatními gatekeepery v síti.

# Kapitola 6

## Přednáška 7

### Hlasové služby

Zahrnují nejen přenos hlasu po internetu (VoIP), ale i služby jako je online textová komunikace (Instant Messaging), zjišťování přítomnosti osoby (Presence Services), vytváření konferenčních hovorů, přenos obrazu (videotelefonní služby) a další.

Přenos hlasu lze implementovat nad více technologiemi - ATM, Frame Relay či přímo nad IP (VoIP).

#### *Klasická telefonní síť*

Vytváří spolehlivé plně duplexní spojení s vysokou kvalitou přenosu hlasu (64 kb/s). Šířka pásma je garantovaná pro každý telefonní hovor. Telefonní přenos se přenáší pomocí přepínání okruhů - vytváření přímého elektrického obvodu mezi volajícími. Telefonní přístroj konvertuje zvuk na elektrický signál a obráceně, zahajuje spojení, oznamuje příchozí hovory. Pro přepojování hovorů slouží ústředny.

Skládá se z: koncová zařízení, ústředny (přepínání hovorů), lokální smyčka (spojení mezi zařízením a ústřednou, elektrický obvod), páteřní linky (spoje, komunikace mezi ústřednami)

Koncová zařízení jsou analogové či digitální, analogové přenášejí zvuk analogovým signálem, digitální obsahují kodek (tím převádí analogový signál na digitální). Jsou připojena na ústředny nebo přímo na veřejné telefonní sítě.

Lokální smyčka tvoří rozhraní do telefonní sítě. Může to být jen pár kabelů pro přenos hovorů.

Přepínače/ústředny mohou být centrální - ukončují lokální smyčku a zajišťují vytváření/rušení hovorů, poskytují signalizaci a přepínání okruhů. Soukromé - jen zákaznické sítě.

#### Signalizace

3 typy - kontrolní, adresová (tónová nebo pulzní volba čísel), informační (stav volajících a stav vytváření hovoru).

Kontrolní pro komunikaci mezi ústřednou a telefonním přístrojem:

- Zavěšeno (On hook) - elektrický obvod s ústřednou přerušen, aktivní pouze vyzvánění
- Zvednuté slucházko (Off hook) - při zvednutí dojde k uzavření elektrického obvodu mezi přístrojem a ústřednou. Proud signalizuje ústředně, že někdo chce uskutečnit hovor, ústředna pošle oznamovací tón oznamující připravenost
- Vyzvánění (Ringing) - Pokud účastník vytočí volajícího, ústředna pošle na jeho přístoj vyzváněcí signál, podobný signál pošle zpět volajícímu.

#### Výhody

Garantovaná šířka pásma, spolehlivý přenos, dobrá kvalita přenosu, napájení telefonních přístrojů z ústředny, spolehlivost, bezchybnost i standardizace. Digitální telefonie nabízí hlasové schránky, konferenční hovory, pozastavení a přesměrování hovorů, lokalizaci účastníka při volání nouzového telefonního čísla.

### *Převod na digitální signál*

Nutnost při přenosu telefonních hovorů přes digitální linky a ústředny. Používán i u IP telefonie. Používají se analogově digitální převodníky (A/D převodník, ADC), převádí do na digitální i zpět. Užívají se procesory DSP (součástí telefonních přístrojů, IP telefonů) provádějící hlasovou kompresi, překódování mezi různými kodeky a umožňují konference mezi více účastníky. Převod signálu se provádí podle vybraného algoritmu pro (de)kódování, kterému se zkráceně říká kodek. U IP telefonů se používá pro překódování (převod kódování hlasu z jednoho kodeku na jiný).

Digitalizace hlasového signálu se skládá z: vzorkování analogového signálu, kvantifikace vzorků, kódování vzorků na binární čísla a komprese.

Vzorkování analogového signálu - probíhá v intervalech, vzorkování musí být 2x větší než nejvyšší frekvence analogového signálu, aby byla možná rekonstrukce. Ucho zachytí 20 Hz až 20 kHz, řeč 200 až 9000 Hz, tradiční telefonie 300 až 3400, digitální až 4000. Pro vzorkování hlasu o 4000 Hz se tedy užívá vzorkování 8000 vzorků/s, vzorek za 125 ms. Výstupem vzorkování je signál pulzní amplitudové modulace (PAM).

Kvantifikace vzorků - přiřazuje nasnímaným vzorkům hodnotu, která odpovídá amplitudě vzorku. Techniky kvantifikace - lineární, logaritmická či metody μ-law a α-law.

Kódování hodnot a komprese - Kódování slouží k převodu kvantifikovaných hodnot na binární vyjádření. Společně s tím dochází ke kompresi hlasových dat. Kodedy (kódovací algoritmy) lze rozdělit na - kodeky používající kódování tvaru vlny (algoritmny PCM, DPCM) a kodeky používající kódování parametrů zdroje řečového signálu(hlasové kodeky).

Kodeky například G.711, G.726, G.728, G.729. Velikost vzorku ovlivňuje potřebné přenosové pásmo.

Proces převodu digitálního signálu na analogový zahrnuje dekódování, (převod kódováného vyjádření vzorků na velikost amplitudy) a filtrování (slouží k rekonstrukci signálu na analogový signál).

### *Kvalita přenosu*

Ozvěna (odraz hlasového signálu zpět k volajícímu, potlačení echa se dělá mikrofonem s potlačovači echa), ztrátovost (nestabilní síť, zahlcení, zpoždění, malé ztráty mohou být rekonstruovány kodeky využívající algoritmy PLC, jiné kodeky ty algoritmy obsahují), zpoždění (čas mezi mluvením a posloucháním volaného, pevné i proměnlivé, eliminace prioritizací hlasových paketů pomocí hodnoty QoS v IP datagramu, < 150 ms je super, > 400 nelze provozovat hovor), kolísání zpoždění (doba mezi očekávaným a skutečným příchodem paketu, toto zpoždění vzniká během přenosu, každý IP datagram se posílá nezávisle na ostatních, po sobě jdoucí datagramy mohou jít jinou cestou, lze elminovat použitím vyrovnávacích bufferů u příjemce), kodek.

### *Hodnocení kvality přenosu*

#### ACR - Absolute Category Rating

Metoda založena na statistickém měření kvality hlasového přenosu, využívá subjektivního ohodnocení posluchači. Je nezávislá na parametrech zkreslení (kodeky, ztrátovost paketů, šum…). Výstupem je hodnota MOS, hodnota z pětibodové stupnice dané ITU-T P.800, pomocí které testovací subjekty vyjadřují své hodnocení (1 až 5). Dle MOS je nejlepší volbou kodek G.711.

#### E-model a R-faktor

Objektivní metoda. Zohledňuje vliv šumu, hlasitosti, kvantizačního zkreslení, způsobu kódování, ozvěny, zpoždění. Vypočítává se hodnota R (R-faktor), kvalitu označuje v rozsahu 0 až 100. Pokud je hodnota menší než 50, nedoporučuje se provozovat telefonní spojení.

### *IP telefonie*

Integrace datových hlasových služeb na platformu IP přenosů, přenášení hlasu přes datové sítě, snižují se náklady na vytvoření, údržbu a rozšíření sítí, možnost centrální správy systému, mobilita účastníků, snížení nákladů na telefonování.

Základní komponenty: telefony, ústředny (gatekeeper, SIP server) a brány do analogové sítě (gateways). Propojeny síťovou infrastrukturou nad IP. Pro uskutečnění hovoru se musí navázat spojení s ústřednou pomocí signalizačního protokolu (SIP, H.323), dale se vytvoří spojení mezi účastníky a probíhá přenos hlasu transportním protokolem RTP.

Ústředna je síťové zařízení nebo aplikace, zařizuje řízený přístup do IP telefonní sítě včetně registrace uživatele a nastavení jeho konfigurace. Vyhledávání účastníků, směrování hovorů, přidělování přenosového pásma, překlad adres.

Brána kromě propojení IP telefonní sítě s veřejnou telefonní sítí umožňuje propojit IP telefonní sítě s jinými signalizacemu (brána SIP/H.323). Může být součást ústředny.

MCU (multipoint control unit) je řídící jednotka pro komunikaci vice koncových zařízení a bran. Poskytuje prostředky pro vytvoření vícebodové conference.

Aplikační servery - nabízejí další služby pro uživatele IP telefonie (správa identit, hlasová pošta, přidělování IP adres - DHCP, překlad enum - DNS).

Mezi funkce IP telefonie patří převod hlasu na IP datagramy (IP telefon), základní činností je řízení hovorů pomocí signalizačních protokolů, to se skládá z vytvoření spojení, údržby telefonního kanálu a z ukončení hovoru napříč IP telefonní sítí.

### *Signalizace SIP*

Protokol SIP je signalizační protocol pro IP telefonii. Textový protokol s hlavičkami podobnými protokolu HTTP. Komunikace SIP probíhá nad protokolem UDP. Hlavním úkolem je autentizace a autorizace uživatele, sestavení spojení, řízení hovoru a ukončení spojení. Pro adresování používá SIP speciální identifikátor URI ve tvaru sip: user@domain. Směrovací informace uloženy v SIP hlavičce(Via, Route, Record-Route). Neprovádí rezervaci zdrojů pro přenos ani nastavení parametru QoS.

Architekturu SIP tvoří několik component: **SIP server (UAS, User Agent Server)**, který obsluhuje požadavky uživatele na spojení, ten je tvořen:

- Registrační server - registruje uživatele (autentizace a autorizace), informace o registraci se používá pro směrování hovorů určených danému uživateli. Provádí aktualizaci dat na lokalizačním serveru
- Server pro směrování hovorů - informuje klienta o dalším skoku při připojování na alternativní SIP server.
- Lokalizační server slouží k vyhledání volaného klienta, spolupráce s registračním serverem.
- Proxy server analyzuje zprávy SIP, přepisuje hlavičky a přeposílá zprávy na další SIP server.

Dále architekturu tvoří **SIP telefon (UAC, User Agent Client)**, jenž je softwarová aplikace nebo HW zařízení, která iniciuje SIP požadavek. Hlasová data posílá protokolem RTP přímo volanému účastníkovi. Koncové komunikační zařízení.

Při vzniku spojení protokol SIP dojednává mezi účastníky parametry jako například kódování, slouží I k určení IP adresy volaného, během hovoru řídí přenos hovoru, zapojení více účastníků nebo změnu kódování. Vlastní komunikace se skládá z dvou základních akcí - registrace klienta a ustanovení

spojení. Při registraci UAC překládá ID uživatele na adresu zařízení a klient posílá příkazem REGISTER žádost na registrační server, v příkazu uvádí svou IP adresu a port. Server to zkontroluje a přidá do DB přihlášených uživatelů. Po přihlášení následuje vytvoření spojení pomocí příkazu INVITE, v příkazu je adresa volaného ve tvaru sip:user@hostname, pokud je přihlášený, přepošle SIP serveržádost na telefon volanému, volajícímu server vrací potvrzení. Spolupracuje se i s dalšími servery, například s DNS pro překlad vytáčeného čísla či lokalizaci uživatele pomocí lokalizačního SIP serveru. Pokud se vytvoří spojení mezi volajícími vytvoří se datový kanál RTP, je navázán přímo bez účasti SIP serverů.

Příkazy REGISTER - žádost o registraci, INVITE, ACK, CANCEL - vytvoření spojení, BYE -ukončení spojení, OPTIONS - zjištění možností přenosu.

Směrování zpráv SIP - metoda INVITE - směrování žádostí přes SIP servery, Request- URI, hlavičky Route, směrování odpovědí pomocí hlaviček Via, po výměně kontaktů v hlavičce Contact jdou odpovědi přímo.

Součástí zpráv SIP typu INVITE a OK je SDP protokol, který nese informace potřebné pro přijímání hlasového toku.

### *RTP a RTCP*

Protokol RTP je transportní protokol pro přenos multimediálních dat zpracovávaných v reálném čase. Pracuje nad UDP. Obsahuje typ přenášených dat (RTP profil), sekvenční číslo a časovou značku. Podporuje také monitorování přenosu, ale on sám jej nemonitoruje, to dělá RTCP. Pro každý směr a typ médií samostatný tok.

Stadard RTP definuje take kontrolní protokol RTCP, který slouží k monitorování kvality distribuce dat a přenosu řídících informací. Sleduje počet přenesených paketů, ztracených paketů, zpoždění, kolísání rychlosti a další. Provádí zpětnou vazbu aktuálního stavu sítě. Slouží pro předávání řídících informací pro tok RTP.

### *Bezpečnost VoIP*

Základní bezpečností hrozby: Odposlech (může dojít spíše z vnitřní sítě, protože každý IP datagram může z dvou toků - signalizační a transportní - směrován jiným způsobem, není jednoduché zachytit VoIP provoz z vnější sítě), Viry, SPIT (spam over internet telephony), PHIT (phishing over IPT), přerušení služby útoky DoS, DDoS, neautorizované použití služby a výpadky napájení.

Zabezpečit lze pomocí: Řízení přístupu k síťovému médiu technologií 802.1x, oddělení hlasového provozu pomocí VLAN (umožňuje logicky oddělit jednotlivé rámce na úrovni Ethernetu a portů přepínače, přestože se hlasový přenos přenáší fyzicky po stejné kabeláži jako ostatní datový přenos), zabezpečení spojení pomocí IPSec (zajišťuje zabezpečení na úrovni síťové vrstvy pomocí protoklů AH a ESP. Lze vyvořit virtuální privátní sítě, umožňující bezpečný datový přenos přes nezabezpečené sítě) a Secure RTP (rozšíření RTP protokolu pro zajištění důvěryhodnosti, autentizaci a ochranu proti podvržení přenosů RTP a RTCP).

### *Skype, WhatsApp, Viber*

Na rozdíl od VoIP typu SIP nebo H.323 mají proprietární nestandardní protokoly, částá změna koncepce. Decentralizované řešení: peer-to-peer, hybridní architektura. Proprietární správa uživatelských účtů (centrálně). Služba není garantovaná, omezené možnosti propojení s PSTN či klasickými sítěmi VoIP. Společné vlastnosti: vytváření uživatelských účtů, registrace uživatelů, adresování a směrování hovorů, vytváření a udržování hovorů, přenos hlasových dat, návazné služby: Instant Messaging, přenos souborů, obrazu, sdílené plochy.

#### *Výpočet přenosového pásma*

Je třeba brát v úvahu: rychlost generování paketů, velikost hlasového vzoru (počet bytů, které jsou potřeba pro uložení hlasové informace pro digitalizaci hlasu do jednoho paketu RTP), režie IP (definuje počet bytů, které se k hlasovému paketu přidávají během vytvoření IP datagramu. Zahrnuje IP, UDP a RTP hlavičky), režie linkové vrstvy (udává počet bytů přidávaných na linkové vrstvě), režie tunelování (použije se v případě přenosu zabezpečeným kanálem).

# Kapitola 8

## Přednáška 8

### Zajištění kvality služeb

Původní návrh provozu internetu založený na protokolu IPpočítal se standardním modelem provozu služeb "doručení s největším úsilím" (best-effort delivery). Jenže to nestačí, potřebujeme odlišovat třídy provozu a jejich požadavky na přenosové pásmo. Vznikl tedy model integrovaných služeb (rezervace přenosového pásma a výpočetních zdrojů na síťových prvcích pro jednotlivé toky) a model diferenciovaných služeb (značí pakety pomocí polí DSCP v hlavičce IP datagramu, pomocí něj je řadí do několika tříd důležitosti).

Zajištění kvality vychází z požadavků uživatele sítě, jsou často definovány v SLA smlouvě. Obsahuje požadavky na fyzické připojení a síťový přenos (například smlouva mezi zákaznickou firmou a poskytovatelem síťového připojení ISP).

Pro zajištění požadavků SLA se používá klasifikace provozu pomocí například značení paketů, rozložení provozu a ořezání provozu. Rozložení provozu slouží k regulaci rychlosti a objemu provozu toků, přizpůsobuje přenos paketů dané rychlosti specifikované v SLA. Shluky se rozprostírají a tím se lépe využije přenosové pásmo. Lze implementovat pomocí modelu tekoucího vědra nebo token bucket.

Ořezání provozu slouží k omezení maximální rychlosti toku. Každý tok ma přiřazen profil přenosu dat, ten stanovuje maximální rychlost přenosu, pokud tok překročí rychlost, pakety porušující SLA jsou zahozeny nebo označeny nižší třídou přenosu. Lze implementovat pomocí modelu tekoucího vědra.

Rozdíl je v tom, že rozložení provozu ukládá přesahující provoz do paměti (buffers) a vyžaduje dostatečnou paměť pro ty fronty přesahujících, ořezávání ne, pouze špičky ořezává, není tolik efektivní. Rozložení většinou pouze na vstupních bodech, ořezání na vstupních I výstupních bodech síťového provozu.

#### *Mechanismy plánování*

U síťové komunikace se různé toky prokládají a vytvářejí fronty na výstupech síťových prvků, pokud má více portů, každý port ma vlastní frontu. Pro způsob řazení paketů a vybírání z front je název plánování. Provádí to speciální proces plánovač. Plánování hraje důležitou roli pro zajištění kvality služeb. Paket vždy přijde na rozhraní a každému rozhraní náleží 1 HW fronta. Pokud je fronta plná, je poslán do SW fronty, kde se uloží. Pokud i tato je plan, dojde k zahození paketu. Plnost může být kvůli nižší kapacitě výstupní linky, délky paketu či vytíženosti procesoru.

#### FIFO

Používá se pro HW frontu, ostatní typy se užívají pouze v SW. FIFO ve stylu, kdo dřív přišel do fronty, dřív i odejde. Pokud je výstupní fronta plná, tak mechanismus pro zahazování rozhodne, zda bude nově příchozí paket zahozen nebo nějaké pakety z fronty budou odstraněny. Výhoda FIFO je předvídatelné zpoždění paketu. Rychlost linky je R a B je maximální velikots fronty, tak zpoždění D je $D < B/R$. Neřeší priority. Pokud je agresivní tok s příliš mnoha pakety ve shlucích, může fronta FIFO

způsobit odepření služby pro ostatní toky, dokud tento tok není obsloužen. FIFO se používá jako implicitní metoda správy fronty výstupní linky, pokud není implementována žádná jiná správa front. Pokud je ve frontě příliš velký paket, může způsobit zpoždění ostatních menších paketů za ním.

### Prioritní fronty

Klasifikace paketů do jedné či více prioritních tříd ve výstupní frontě. Mohou se užít značky paketů jako bity v poli Type of Service, pole DSCP v hlavičce IP datagramu nebo na základě IP adres, čísel portů… Každá třída má svou frontu. Třídění provádí klasifikátor. Vybírají se (plánovačem) nejprve pakety ze tříd s vyšší prioritou. Až je prázdná fronta, jde se na nižší prioritu. Výhodou je rozlišení toků do různých tříd a nastavení priorit obsluhování.

Může však dojít k vyhladovění, kdy pakety s vyšší prioritou pořád předbíhají ty s nižší. Říká se tomu striktní prioritizace, je vhodná například pro přenos VoIP, když je třeba zaručení nezahození paketů. Další variantou jsou těchto front jsou fronty řízení rychlosti, prioritnější pakety jsou upřednostňovány do té doby, dokud prioritní fronta nepřesáhne daného rychlostního limitu.

### Cyklické fronty

Řeší vyhladovění. Cyklicky stejný počet paketů je odebírán z každé fronty, řazení do front dle priority. Upřednostňovány toky s většími pakety. Proto jsou často užívány spravedlivé fronty (zvláštním typem cyklických front), počet paketů z jednotlivých front je normalizován na délku paketů.

V jednom cyklu se odebere z každé fronty stejný počet paketů či bitů. Pokud je rychlost linky R, tak při n frontách je rychlost zpracování toku R/n bitů za sekundu. Pokud nejsou všechny fronty plně obsazené, rychlost bude větší. Rychlost však neklesne pod tuto hodnotu. Zaručení, že nějaká fronta nemůže být opakovaně předbíhána. Počet toků se dynamicky mění.

### Váhové fronty WFQ (Weighted Fair Queues)

Pakety jsou klasifikátorem řazeny do toků a každému toku je přiřazena fronta. Tok je identifikován na základě informací z hlaviček IP a TCP/UDP (zdrojové a cílové adresy, čísla protokolu, typu služby …). Na základě těchto informací se vytvoří hashovací funkcí ukazatel do fronty. Pokud fronta pro tok neexistuje, vytvoří se, pokud ano, paket se do ní zařadí. Fronty jsou obsluhovány plánovašem pro řízení váhových front WFQ. Váhové fronty rozšiřují vlastnosti cyklických front, počet odebraných paketů závisí na váze, která je přiřazena jednotlivým frontám. U cyklických Round Robin se z každé fronty vybírá jeden paket. Tedy u váhových front to v závislosti na váze fronty může být i více paketů.

U linky s rychlostí R bude rychlost obsluhy paketů $r_i$ toku I dána vztahem $r_i = R.\frac{w_i}{\sum_j^n w_j}$. Lze garantovat požadovanou rychlost či maximální zpoždění. Zajištěno mechanismem Leaky Bucket ($D_i <= b_i/r_i$), kde b je velikost vědra, r je rychlost generování žetonů a D je maximální doba zpoždění toku i. Počet front závisí na počtu aktivních toků. Dynamicky se vytváří a ruší toky => není umožněno přesné zajištění přenosového pásma. Pokud jsou fronty zaplněny, pakety jsou zahazovány.

### Mechanismus Leaky Bucket (tekoucí vědro)

Slouží k řízeni toku dat, zejména pro rozložení provozu a také pro ořezání. Vnutí vstupnímu provozu, který je tvořený obvykle nepravidelně přicházejícími shluky dat, konstantní přenosovou rychlost.Aneb vyhladá vstupní provoz na tok se stálým výstupem. Pakety nejsou posílány vyšší rychlostí, než je povolená. Velikost vědra ohraničuje maximální rychlost. Pokud nic ve frontě (vědru) není, rychlost je nulová. Pokud přichází nové pakety rychleji, než je rychlost na výstupu, tak se fronta (vědro) zaplňuje, a pokud je plné, tak se pakety navíc, co se nevlezou do fronty (vědra), zahodí. Vědro je implementováno jako FIFO fronta s časovačem a čítačem.

Velikost prvního paketu ve frontě se porovná s hodnotou čítače, pokud je hodnota čítače větší, tak se sníží o velikost paketu a paket se přepošle. Pokud by byl paket větší, tak přenos zastaven na sekundu a čeká se na vypršení časovače a inkrementaci čítače. Čítač se inkrementuje každou sekundu (vypršení časovače). Pokud se nově příchozí paket nevejde do fronty, tak se zahodí.

### Mechanismus Token Bucket (zásobník žetonů)

Slouží k omezení rychlosti na konstantní rychlost bez ohledu na to, jaké shluky dat se mohou objevit v sítí objevit. Umožňuje krátkodobě překročit výstupní rychlosti tak, aby vstupní hluk prošel bez zahození. K tomu se užívá mechanismus zásobníku žetonů. Žeton = povolení přeposlat 1 byte. Žetony se generují určitou rychlostí. Zásobník umožňuje odeslat shluky dat až do velikosti vědra naráz a bez čekání (tohle leaky bucket neumožňuje). Na výstupu se mohou objevit omezené shluky dat, což tekoucí vědro neumožňuje.

Algoritmus zásobníku žetonu dovoluje propouštět určité množství shluků při dané průměrné rychlosti CIR (velikost dat, které se za daný časový interval pošlou). Dále je definována maximální velikost shluků CBS, která definuje počet bytů, které mohou být krátkodobě přeposlány po síti. Časový interval T definuje dobu trvání shluku. Rychlost ve špičce definuje maximální počet bytů, které lze přenést během kratší časové periody.

Průměrná rychlost CIR je poměr CBS/T, kde T je časový interval trvání shluků. Maximální rychlost ve špičce je dána PIR (maximální počet bytů, které lze poslat během kratší perioudy, u CIR je to během relativně delší) = CBS/t + CIR.

Zásobník má velikost CBS (maximální velikost shluku), žetony se generují rychlostí CIR žetonů za časový interval. Pokud dojde paket o velikosti P a zásobník obsahuje X žetonů a X >= P, tak se odebere P žetonů a paket se přepošle na výstup. Pokud je zásobník prázdný a nebo počet žetonů je X < P, paket čeká na vygenerování potřebných žetonů. Při shluku paketů se pošle tolik paketů, kolik mám pro ně žetonů.

Může se užít kombinace zásobíku žetonů a tekoucího vědra o rychlosti r, kde r je větší než CIR, ale menší než PIR.Tímto omezíme výstupní rychlosti na námi definovanou hodnotu.

Pokud by se jednalo o ořezávání, tak by se v případě málo tokenů paket zahodil, prostě zásobník žetonů neexistuje frontu pro ukládání paketů, co přijdou na vstup.

### *Integrované služby*

Jeden ze způsobů jak zajistit požadovanou kvalitu služeb. Poskytují společnou (integrovanou) službu množině provozních požadavků v určité doméně (části internetu). Úkolem integrovaných služeb je zajistit podporu QoS v sítích nad IP (jak sdílet dostupnou kapacitu síťových zdrojů v době zahlcení). V HW části směrovače je optimalizována nějaká rychlost a probíhá zde klasifikace paketů do tříd provozu, ta může odpovídat jednomu čí více tokům. Klasifikace na základě dat z hlavičky IP datagramu. Plánovač paketů obsluhuje jednu či více front pro každý výstupní port, určuje pořadí vybírání z front a případné zahození. Rezervace zdrojů probíhá v SW části.

Tvoří několik součástí - rezervace zdrojů (pomocí RSVP protokolu), řízení přístupu, způsob řízení front a způsob zahazování paketů.

Zajišťují 2 základní funkce: Rezervaci zdrojů (směrovač musí vědět, kolik zdrojů je rezervováno pro probíhající komunikaci), vytvoření spojení.

### Vytvoření spojení

Relace vyžadující určitou kvalitu služeb musí být nejdříve schopná rezervovat si dostatečné množství zdrojů na každém síťovém směrovači od zdroje k cíli. Tomu se říká vytvoření spojení (call setup, call admission). Musí spolupracovat každý směrovač po cestě, musí se určit lokální zdroje, kterou jsou potřeba.

Skládá se ze specifikace provozu a požadavků na kvalitu služeb QoS (relace je musí oznámit), signalizace pro vytvoření spojení (pro přenos parametrů Tspec a Rspec relace slouží protokol RSVP) a předběžné přijetí spojení (jakmile jsou ovdrženy požadavky, tak směrovač určí, zda je přijme, pokud není dostatek zdrojů, odmítne).

Architektura integrovaných služeb definuje 3 základní třídy služeb:

- Garantovaná služba - definuje striktní zpoždění ve frontách během zpracování ve směrovači. Přenášený provoz popsán mechanismem Token Bucket. Požadovaná kvalita je charakterizovaná přenosovou rychlostí R. Zajišťuje tedy požadovanou datovou rychlost a nedochází k žádným ztrátám z důvodu přeplnění front.
- Síťová služba kontrolované zátěže - poskytuje datovému toku kvalitu služby, která se s velkou pravděpodobností blíží kvalitě nezatíženého síťového prvku. Velké procento paketů projde úspěšně bez zahození a zpoždění se bude blížit nule. Není však uvedeno, jak velké procento je, které projde. Nezajišťuje kvantitativní parametry přenosu dat.
- Služba s největším úsilím - sem spadají všechny pakety, které nepatří do žádného toku s rezervovanou službou, většina toků v této kategorii užívá mechanismus TCP pro řízení rychlosti v případě zahlcení.

### RSVP

Rezervační protokol RSVP je protokol, který umožňuje skupině vysílajících stanic přenést vysílané toky v požadované kvalitě k přijímacím stanicím tak, aby přenosové pásmo bylo optimálně využito a předešlo se zahlcení přenosových zdrojů. Umožňuje rezervaci zdrojů na síťových prvcích.

Požadavek na rezervaci provádí koncová stanice pomocí protokolu RSVP nejbližšímu směrovači, ten ji posílá dále po cestě ke zdroji. Poskytuje rezervaci přenosového pásma v multicastových stromech - předpoklad, že většina aplikací, které budou žádat o rezervaci, komunikují přes multicast. Rezervační zprávy posílá přijímací stanice, i když jsou data generována odesílatelem. Protokol orientovaný na příjemce.

### Zhodnocení integrovaných služeb

Provádí alokaci prostředků pro jednotlivé datové toky v síti (per flow), což je výpočetně náročné (pro každý tok procházející směrovačem vyžaduje zpracování rezervačního požadavku + alokaci zdrojů), ale je zde garance kvality přenosu pro jednotlivý tok.

Dalším problémem je malý počet tříd služeb. Množina tříd neumožňuje popsat kvalitativní či vztahové požadavky (Služba třídy A má přednost před službou třídy B). To je implementováno v diferenciovaných službých.

Je to nevýhodné pro páteřní směrovače. Řízení kvality přenosu pomocá integrovaných služeb se dnes spíše nepoužívá. Používá se model rezervace přenosového pásma pomocí RSVP, užívá verze RSVP-TE (Traffic Engineering) u sítí MPLS.

### *Diferenciované služby*

Architektura těchto služeb se zaměřuje na rozšiřitelnost modelu a flexibilitu - schopnost pracovat s různými třídami provozu. Rozšiřitelnost počítá i s užitím na páteřních sítích. Hraniční směrovače

provádějí výpočetně náročné operace DiffServ, páteřní směrovače provádějí jednodušší funkce. Tím se rozkládá zátěž. Používají se předdefinované třídy, ale je umožněna definice vlastních tříd - snadná rozšiřitelnost a flexibilita.

Architekturu těchto služeb tvoří 2 typy prvků:

- Hraniční prvky - koncová zařízení nebo nejbližší směrovače podporující diferenciované služby. Značkují procházející pakety (nastavují hodnotu DS (differentiated services) v hlavičce IP datagramu). Značení identifikuje třídu provozu, do které patří. Po označení může být paket poslán do sítě, vložen do fronty nebo zahozen.
- Páteřní prvky - Když přijde označený paket na směrovač s podporou DiffServ, je přeposlán na další uzel podle definovaného chování příslušející jeho třídě (tzv. PHB, Per-hop Behavior). Nastavené chování PHB ovlivňuje sdílení paměti a přenosového pásma mezi různými třídami provozu. Podstatné na architektuře DiffServ je, že na páteřním směrovači se zpracování paketu řídí pouze značením v hlavičce IP. Pakety z různých PC, ale se stejným značením nejsou pak směrovači rozlišovány a jsou brány jako jeden tok.

### Klasifikace provozu pomocí DSCP

Toto značení paketu se ukládá v poli DS (Differentiated Services) v hlavičce protokolu IP. Pole není v původní definici protokolu IP, bylo to Type of Service. Pole se skládá z 6-bit kódu DSCP, který obsahuje typ chování PHB (Per-Hop Behavior), a dvoubitové hodnoty CU (Currently Unused), která se momentálně nepoužívá. DSCP hodnoty nastavuje hraniční prvek sítě na základě klasifikace provozu. Klasifikace může být provedena podle zdrojové/cílové adresy, čísla portu, typu protokolu či obsahu paketů. Například poskytovatel internetu dle SLA může určitá data z nějaké adresy označovat kódem DSCP označujícím vyšší prioritu ve své síti.

Klasifikátor je ten, který dle hlaviček IP či obsahu paketů přicházející pakety rozděluje do tříd. Meter taky sleduje příchozí provoz a přiřazuje ho do předdefinované třídy provozu. Taky kontroluje, zda provoz nepřekročil stupeň garantované služby. Marker značkuje pakety novým DSCP kódem. Překročivší provoz přeznačkuje. Sharper provádí rozložení provozu v čase tak, aby provoz nepřekročil dohodnutou úroveň služby dle SLA. Překročivší část toku (pakety) je zahozen Dropperem.

### Definice chování pomocí PHB

Popisuje způsob zpracování toků označených pomocí DSCP. Pro různé třídy provozu je definováno různé zpracování. PHB pouze popisuje definici chování, ale nezabývá s vlastní implementací, která ji zajistí. Tzn. neřeší typy použitých front apod. Příkladem PHB může být garance přenosového pásma o velikosti M procent kapacity výstupní linky pro třídu provozu A.

2 speciální typy PHB:

- Přednostní posílání EF (Expedited Forwarding) - výstupní rychlost odesílání paketů dané třídy ze směrovače musí být stejná nebo vyšší než nakonfigurovaná rychlost. Nebere se ohled na intenzitu ostatního provozu, pokud provoz ostatních tříd zahltí směrovač, musí mít směrovač stále nějakou rezervu zdrojů pro zajištění minimální požadované rychlosti odpovídající požadovanému chování. Po přednostní posílání se používá 1 třída EF s doporučenou hodnotou 46 DSCP.
- Garantované posílání AF (Assured Forwarding) - rozděluje provoz do 4 tříd AF1 až AF4, z nichž každé třídě je zaručena určitá přenosová rychlost. U každé třídy jsou definovany tři úrovně pravděpodobnosti zahození v případě zahlcení. V případě zahlcení se začnou nejdříve zahazovat pakety s vyšší pravděpodobností.

- Implicitní třída BE (Best-Effort) První tři bity nastaveny na nuly. Tato třída se použije, pokud není hodnota DSCP v paketu namapovaná na nějakou politiku PHB. Jedná se o doručení s největší úsilím.

V dnešní době se diferenciované služby využívají častěji nez integrované. Rozšíření diferenciovaných služeb je obvykle omezeno na jednu administrativní doménu (DS doménu), což většinou zahrnuje síť jednoho poskytovatele internetového poskytovatele. Zaručení kvality služeb mezi 2 libovolnými koncovými stanicemi je však závislé na všech poskytovatelích, přes které se tok přenáší. V dnešní době zpoždění ovlivňuje spíš kapacita přístupové linky a počet skoků na cestě než zpoždění ve frontách na směrovačích.

### *Předcházení zahlcení pomocí RED a WRED*

Každá fronta je konečná a je ji možné někdy zahltit. Když dojde k zaplnění, tak se začnou zahazovat všechny nově příchozí pakety. Pokud tyto pakety patří k toku TCP, tak mechanismus řízení toku TCP změnší přenosové okno a sníží rychlost toku. To způsobí výrazný pokles nároků na přenosové pásmo i na uvolnění front. Až se průhodnost linky zlepší, TCP zase zvyšuje rychost, dokud se zase nezahltí fronta. A pak přijde zase snížení rychlosti - cyklus. Je to tzv. problém globální synchronizace TCP.

Při zahlcení může dojít k problému, pokud o přenosové pásmo soupeří TCP a UDP. Pokud sníží svou rychlost kvůli zahlcení TCP, tak UDP nemá žádný mechanismus řízení rychlosti a pokud přes zařízení prochází agresivní tok UDP, tak snížení TCP využije a obsadí další přenosovou kapacitu linky. To donutí ostatní toky TCP ještě k většímu snížení rychlosti až je vytěsní, tomu se říká vyhladovění TCP.

To se řeší mechanismem pro prevenci zahlcení, který sleduje nárůst paketů ve frontách. Když velikost fronty přesáhne minimální práh Qmin, začnou se náhodně zahazovat pakety. Pravděpodobnost se zvyšuje v čase s délkou fronty. Pokud přestane růst, dál pravděpodobnost neroste ani neklesá. Pokud roste dál, pravděpodobnost se zvyšuje. Pokud se dosáhne maximálního prahu Qmax, pravděpodobnost je 100%. Všechny pakety jsou zahazovány. Říká se tomu algoritmus RED. Vybírá pakety pro zahození bez ohledu na jejich vlastnosti (prioritu). Pravděpodobnost zahození Pa závisí pouze na aktuální délce fronty a je popsána lineární funkcí: Pa = Pmax * (Qavg - Qmin) / (Qmax - Qmin).

Detekce zahlcení pomocí algoritmu RED má vliv i na plynulejší využití přenosového pásma. Nemůže eliminovat výkyvy kvůli chování TCP při ztrátách paketů (snížení rychlosti), ale přizpůsobí výkyvy v rychlostech kapacitě přenosové linky. Lépe se využívá celková kapacita linky.

Qmin a Qmax určují, jak efektivně budeme linku používat. Pokud by byla Qmin moc nízká, povede výskyt shluků paketů v toku dat k zahazování a k nevyužití kapacity linky. Musí být velká dost na to, aby směrovač byl schopen přijmout dostatečně velký shluk paketů bez potřeby zahazování. Qmax určuje zpoždění paketu během přenosu ve směrovači. Velká Qmax znamená, že více paketů bude čekat ve frontě, ale zvýší to zpoždění paketu. Qmax by mělo být aspoň 2x tak velké jako je Qmin.

Rozšířením algoritmu RED je přidání vah pro jednotlivé třídy paketů. Modifikace se nazývá váhový RED (weighted RED), tedy WRED. Nezahazuje všechny pakety se stejnou pravděpodobností, ale rozlišuje jejich důležitost pomocí hodnot IP precedence nebo DSCP. Pro různé třídy jsou definovány různé fronty s odlišnou politikou zahazování, která je definována pomocí hodnot Qmin a Qmax. Pakety s nižší prioritou budou zahazovány dříve, čímž umožní průchod paketů s vyšší prioritou.

WRED pro každý paket ve frontě vypočte aktuální délku fronty. Pokud přijde paket, dojde ke klasifikaci podle hodnoty IP precedence nebo hodnoty DSCP. Podle typu třídy a aktuální délky fronty se určí pravděpodobnost zahození. Jestliže není zahozen na základě průměrné délky fronty, může být

umístěn do výstupní fronty FIFO. Když je plná, je zahozen. Pokud ne, je přidán do fronty a přepočítá se aktuální velikost fronty.

Použití algoritmu RED a WRED řeší problém globální synchronizace TCP a vyhladovění TCP, protože zahazování je přímo uměrá dravosti toku. Agresivní toky budou zahazovány s větší pravděpodobností. V konečném důsledku přináší tyto algoritmy efektivnější využití přenosového pásma.

# Kapitola 9

## Přednáška 9

### Klasifikace paketů a filtrování dat

Klasifikace paketů do tříd dle různých kritérií je jednou ze základní činností síťových zařízení (směrovače a firewally). Klasifikace slouží ve firewallech pro filtrování paketů. Většinou pro směrování se používá klasifikace dle cílové IP adresy. Směrovač přijme paket, podívá se na cílovou IP adresu a ve směrovací tabulce najde nejvhodnější záznam s adresou sítě, kam se to má směrovat. Směrovač pak paket přepošle.

Pakety ze vstupního rozhraní se rozdělují na pakety, které vyhovují podmínkám a projdou síťovým rozhraním, a na pakety, které ne a jsou zahranozeny. Pro popis filtrovací funkce se používají filtrovací pravidla, těm se říká ACL. Filtrujeme dle informací z hlaviček pomocí algoritmů, které si uvedeme.

Firewally používají pro klasifikaci také hodnoty z hlaviček IP, UDP, TCP - zdrojové IP adresy, typ protokolu, třída kvality služeb (ToS), porty, příznaky TCP atd. Klasifikace se používá k filtrování paketů, rezervaci zdrojů, zajištění kvality služeb (QoS) či směrování na unicastu i multicastu.

Hlavní problém klasifikace je požadavek na průběhu na základě co největší množiny kritérií, na druhé straně chceme co nerychlejší průchod paketu zařízením. Jsme ovšem omezeni parametry zařízení (velikost vstupních a výstupních linek, rychlost procesoru…).

Hraniční směrovače většinou obsahují desítky filtrovacích pravidel, páteřní však to může být až ve stovkách tisíců. Je třeba proto tedy využívat co nejefektivnějších datových struktur pro uložení pravidel. Lineární seznam není ideální z hlediska toho, že nemusíme pouze hledat jeden první konkrétní záznam, ale více nebo ten nejlepší ve shodě. Paměťová složitost tedy závisí pouze na počtu pravidel, časová složitost roste nejen s počtem pravidel, ale i s počtem polí v pravidlech. Čím více pravidel je, tím významněji narůstají paměťové nároky, ale také doba zpracování. Je třeba využívat výhodnějších struktur v efektivitě. Například: metoda rekurzivní klasifikace toků, metoda bitového vektoru, rozhodovací stromy ři hierarchické rozhodovací stromy.

Záleží však také na to, zda pro vyhledávání a klasifikaci se využívá pouze jedno políčko z hlavičky či více. Podle toho rozdělujeme metody na jednodimenzionální pro vyhledávání pouze jednoho parametry (například stromy trie) a vícedimenzionální.

### Klasifikace paketů

Algoritmický problém, kdy se snažíme roztřídit (klasifikovat) pakety dle zadaných pravidel. Cílem je co nejrychleji vyhledat ke každému přichozímu paketu pravidlo, jehož položky se shodují s hodnotami v hlavičce paketu.

2 fáze klasifikace - **předzpracování** (příprava a uspořádání databáze do datových struktur optimalizovaných pro vyhledávání, tato fáze pouze v případě přidávání, úpravy či mazání pravidel, probíhá to na centrálním procesoru zařízení), **klasifikační fáze** (zpracování příchozího provozu,

vyhledání dát z hlavičky paketu, průchod datovou strukturou pravidel vytvořené v předzpracování, porovnání hodnot hlavičky s těmi z pravidel a vyhledá se nejlepší klasifikační pravidlo, probíhá to na HW síťového zařízení).

Každá hlavička obsahuje n-tici polí například IP paketu. Obvykle procházíme hlavičku IP datagramu na 3. vrstvě či hlavičku TCP/UDP datagramu ze 4. vrstvy. Ale můžeme procházet i z jiných vrstev (2. či 5.). Většinou se užívá pro klasifikaci IP adresa, typ protokolu a port.

Ke klasifikaci se užívá konečná množina pravidel, u firewallů jsou uspořádané do posloupnosti (záleží na pořadí), u směrovačů nezáleží. Každé pravidlo obsahuje k-tici hodnot, které opovídají polím v hlavičce paketu.

Každé pravidlo specifikuje tok dat, do kterého paket patří. Provádí nad tímto tokem dat akce. Nemusí to být pouze zahození či povolení paketů, ale třeba například přesměrování. Každé pravidlo také obsahuje typ porovnání, který se provádí nad daným pravidlem:

- Přesné porovnání – Hodnoty se musí shodovat (pole z hlavičky a z pravidla).
- Porovnání na shodu prefixu – Hodnota v pravidlu musí být prefixem hodnoty paketu v hlavičce.
- Intervalové porovnání – Hodnota z hlavičky musí být v intervalu, který je zadán v pravidle
- Porovnání regulárního výrazu – porovnání pomocí regulárních výrazů – při analýze obsahu paketu

Množina pravidel je uložena databázi pravidel daného směrovače nebo firewallu. Každé má ID, které označuje pořadí pravidla v seznamu. Při vyhledávání se hledá první vhodné pravidlo. Poslední pravidlo bývá většinou implicitní (např. když se to neshodlo s ničím přede mnou, zahoď, tzv. inkluzivní firewall).

Počet polí hlavičky, které se používají při porovnávní s pravidly, určují dimenzi. Jednodimenzionální vyhledávání většinou ve směrovačích, vícedimenzionální vyhledávání ve firewallech.

Metriky hodnocení klasifikačních algoritmů: rychlost, paměťová náročnost, rychlost aktualizace, dimenze, rozšiřitelnost, složitost implementace.

### *Klasifikace v 1 dimenzi*

Použití pro směrování dle cílové IP adresy (vyhledávání IP adres nebo prefixové vyhledávání).

### Prefixové vyhledávání IP adres

Prefix max 32 bitů (IPv4), především na směrovačích se provádí, procesor rozhoduje, kam bude IP datagram dále odeslán. Dívá se do tabulky FIB, ta obsahuje množinu prefixů a odpovídajících výstupních rozhraní (next-hop). Při porovnávání se vyhledává záznam FIB s nejdelší shodou prefixu IP adresy a s tím danou výstupní linku. 2 operace směrovače: vyhledání adresy dalšího uzlu a přeposlání na příslušné výstupní rozhraní. Prefixy mají v tabulce napsanou délku masku sítě za lomítkem, jenže cílové IP adresy nepřicházejí s maskou, takže se opravdu vybere jen nejdelší shoda prefixu.

Časově náročnější je vyhledávání IP adres ve směrovacích tabulkách. Při vyhledávání prefixu hraje roli také maximální délka prefixu W – určuje výšku stromu při reprezentaci pravidel ve stromové struktuře trie.

### Binární stromy trie

Binární stromová struktura, do které ukládáme prefixy IP adres ze směrovací tabulky pro klasifikaci v jedné dimenzi. Uzly tvoří části IP adres, hrany odpovídají jednotlivým bitům adresy. Cesta z uzlu ke kořeni = prefix. Výška stromu = nejdelší prefix (lze i 32). Ne každý uzel obsahuje prefix ze směrovací

tabulky. Struktura trie obsahuje pouze větve, které odpovídají pravidlům ve směrovací tabulce. Časová složitost vyhledávání, vkládání i rušení je O(W), kde W je maximálnídélka prefixu v tabulce. Paměťová složitost je O(N.W), kde N je počet prefixů zadných v pravidlech vyhledávání. Využívá se pro 1D vyhledávání ve směrovacích tabulkách, hledá se nejdelší shoda, vždy si pamatujeme prozatimní nejdelší shodu. Vybere se nejdelší shoda, takže nemusíme řešit detekci společných prefixů.

### Vícebitové stromy trie

Nevýhoda binárních stromů trie je, že porovnání každéhobitu adresy či prefixu je velmi pomalé a často zbytečné. Vícebitové umožňují porovnat více bitů v jednom kroku, což snižuje časovou náročnost a zredukuje počet přístupů do paměti. Krok může mít proměnlivou velikost bitů. Pokud všechny uzly jedné úrovně mají stejně velký krok, tak je to vícebitový strom trie s pevným krokem, jinak s proměnlivým. Počet potomů určuje tzv. krok.

Na vstupu nesmí být prefixy libovolné délky, vstupní prefixy se musí transformovat na ekvivalentní prefixy větší délky (nejbližší větší) tak, aby odpovídali délce prefixů vícebitového stromu. Rozšíření probíhá tak, že u přidaných bitů musíme zvolit všechny možné kombinace jedniček a nul, pokud už v tabulce daný prefix je, vyškrtneme jej, přednost mají existující prefixy.

Když máme pevnou délku kroku a prefixy jsou délky 3 a 5, tak nejdříve v jednom kroku se porovnávají první 3 bity, v druhém zbylé 2.

Složitost vyhledávání je O(W/k), kde W je maximální délka prefixu v tabulce a k je počet bitů jednoho kroku. Aktualizace je složitější O ($W/k + 2^k$), protože kromě hledání ještě zahrnuje čas modifikace $2^k$ – 1 záznamů. Tohle také zároveň celková časová složitost tohoto stromu.

### *Klasifikace ve více dimenzích*

Například v páteřních směrovačích (pravidla cílová – zdrojová IP adresa)

### Struktura trie ve dvou dimenzích

Klasický trie struktura se používá pro 1D vyhledávání. Pro 2D lze využít její zobecnění „síť stromů trie“. Pro 1. dimenzi vytváří jeden strom trie (např. strom s cílovou IP adresou). Uzly obsahující prefixy odkazují na další strom trie pro vyhledávání v 2. dimenzi (např. zdrojová IP adresa). Není to však efektivní kvůli rozšiřitelnosti a prostorové složitosti. Dochází k exponenciálnímu nárůstu prostorové složitosti, protože stromy se kopírují do více koncových uzlů prvního stromu. Paměťová náročnost je pro N*N prvků a při K dimenzí O($N^K$). Exponenciálně roste s počtem dimenzí. Řešením je použití odkazů na opakující se strukturu ve druhé dimenzi.

### Strom trie se zpětným vyhledáváním (backtracking)

Dalším upůsobem, jak se vyhnout exponenciálnímu nárůstu paměťové složitosti je vytvořit strom trie pro 2. dimenzi tak, žer nebude kopírovat pravidla druhé dimenze z předchpzích prefixu, ale každé pravidlo bude uloženo pouze jednou. Algoritmus je časově náročnější, protože prohledává nejen trie druhé dimenzepro prefix první dimenze D, ale i další trie druhé dimenze, na které se odkazují předchůdci D. Aneb pokud nenajdu shodu v 2. dimenzi, začnu se vracet v 1. dimenzi a procházet 2. dimenze pro kratší prefixy. Paměťová náročnost je O(N.W) a časová je při nejhorším O(W.W), kde W je maximální počet bitů jedné dimenze. V praxi tohle může pracovat dobře. Prostorová náročnost menší, ale časová větší.

### Strom trie s ukazateli

Zlepšuje algoritmus prohledávání 2D struktury trie. Prohledávání zkráceno tím, že místo zpětného prohledávání uzlů první dimenze máme v druhé dimenzi předpočítaný přímý ukazatel na jiný možný

strom v 2. dimenzi, kde bude vyhledávání úspěšné. Zlepší se složitost vyhledávání na O(2.W) pro dvě dimenze. Tohle se jmenuje zpětné vyhledávání s ukazateli. Stromy druhé dimenze se neduplikují, tudíž zachována lineární paměťová složitost. Snížení časové náročnosti.

### *Klasifikace ve více dimenzích*

Používají se k tomu různé heuristické algoritmy vytvořené na základě analýzy činnosti firewallů a směrovačů. Používá se pro implementaci filtrovacích pravidel. Časově i prostorově náročné, je třeba zvolit kompromis. Ne všechny položky se používají stejně často. Metody se většinou zaměřují na optimální vyhledání ve 2D. Předpoklady pro využití heuristických metod: Prefixy IP adres se porovnávají do maximálně 24 bitů. Průnik prefixů není obvyklý (mít prefixy, které jsou součástí jiných prefixů) - například prefixy 00* a 00001*. Intervaly se v praxi příliš nepoužívají. Počet disjunktních klasifikačních regionů je relativně malý. Téměř všechny pakety se porovnávají při klasifikaci DstIP - SrcIP na pěti pravidlech.

### Lineární seznam pravidel

Jednoduchý způsob pro klasifikaci paketů ve více dimenzích vycházející z pozorování, že většina porovnání pravidel používá k vyhledávání převážně dvojici zdrojová IP - cílová IP adresa (S, D). Pro reprezentaci k-rozměrných pravidel najdeme vhodnout 2D reprezentaci (síť stromů trie), na níž navážeme pro různé dvojice (S,D) lineární seznam pravidel, které odpovídají daným hodnotám (S,D) v pravidle. Nejdříve se prohledává 2D prostor, pak pro vyhovující dvojice prohledávání jejich lineární seznam pravidel a porovnáme hodnoty paketu s položkami ve zbývajících dimenzích. Výhodou je, že každé pravidlo je reprezentováno jen jednou. Čas vyhledávání závisí na čase 2D vyhledávání a na době průchodu lineární seznamem o maximální délce c <= 20. Příkladem může být využití stromů tre a jejího rozšíření o další dimenze do tak zvané rozšířené sítě stromů trie.

### *Metody typu rozděl a panuj*

Využívá myšlenku rozdělení problému na jednodušší podčásti, které pracují nezávisle a celkový výsledek se složí z dílčích výsledků. Například rozdělení do jednotlivých dimenzí, kde každá položka obsahuje disjunktní prefixy dané dimenze. Každá dimenze je pak prohledávaná pomocí 1D metod (metoda pro vyhledávání nejdelšího shodného prefixu). Pro každou dimenzi lze využít jinou metodu. Lze využít paralelismu. Každá metoda by měla vrátit spíše množinu pravidel, abychom v kombinaci s ostatními dimenzemi našli co nejlepší pravidlo vyhovující všem dimenzím.

### Lineární prohledávání pomocí bitového vektoru

Metoda typu rozděl a panuj. Obsahuje výskyt prefixu v jednotlivých pravidlech. Základní myšlenkou algoritmu je nejprve hledání příslušného pole z hlavičky paketu v seznamu pravidel podle jedné dimenze. Výsledek tohoto dílčího hledání se uloží do bitové mapy. Celkový výsledek je pak průnikem všech dílčích výsledků - průnikem vektorů reprezentující bitové vektory (bitové mapy) jednotlivých dimenzí (aneb kde po operaci AND bude 1, to pravidlo se aplikuje). Lze aplikovat paralelismus. Časová složitost O(N), N je velikost bitového vektoru, je nezávislá na počtu dimenzí. Pro lepší využití paměti se používá modifikace bitových vektorů, která se nazývá agregované bitové vektory. Doporučeno používat tuto metodu pro databáze střední velikosti, přidávání nových pravidel je pomalé a vyžaduje aktualizaci celé databáze.

# Kapitola 10

## Přednáška 10

### Správa sítí

Pojem zahrnuje instalaci kabelů, připojování PCs a konfiguracu IP adres, sledování a monitorování aktivit na síti, bezpečnost, detekci chyb, útoků, vyvažování výkonu sítě, plánování, zálohování a obnova zdrojů dat… Správný administrátor by se měl o chybě dozvědět nejdříve.

#### Koncepce a cíle správy sítě

Výchozím bodem pro správu sítě je zjištění aktuálního stavu sítě. K tomu potřebujeme informace o topologii sítě (jaká zařízení jsou zapojena, jaké mají síťové adresy), aktuální stav zařízení (přehled běžící rozhraní, procesů a poskytovaných služeb), statistické údaje o přenosu (počet přenesených bytů, paketů, broadcastových paketů). Někdy i obsah přenášených dat. Zjistíme dle analyzátorů síťového provozu - tato data načítá a analyzuje.

#### Získání metadata o stavu sítě (metadata)

Může být:

- Pasivní - sledování probíhající komunikace na síti, sbírání informací o stavu sítě a zařízeních na sítích, o chybách, nedostupnosti zdrojů, pokusy o neoprávněný přístup k síťovým službám. Využívají se logovací informace aplikací a služeb sbírané Syslogem. Dále hlášení o stavu sítě asynchronními zprávami SNMP nebo záznamy Netflow.
- Aktivní - administrátor nečeká pouze na zprávy, ale aktivně se dotazuje na dostupnost linek, aplikací či síťových prvků. Například testování dostupnosti síťových zdrojů pomocí ICMP zpráv, dat získaných protokolem SNMP, také pomocí telnetu na TCP port sledovaných služeb. Aplikace na aktivní monitorování běží v nekonečné smyčce a pokud zařízení neodpovídá, tak během časového intervalu se opakovaně dotáží, pokud stále nic, je zařízení prohlášeno za nedostupné a správce informován.

#### Analýza provozu

Proces, který se zabývá podrobnějším zkoumáním a vyhodnocováním nasbíraných dat. Využijí se dříve nasbíraná data nebo se sledují procházející datové toky. Analyzuje jak hlavičky protokolů na vrstvách L2 a L3 tak i obsah paketů až do vrstvy L7.

V reálném čase může být těžké data zachytávat, neboť rychlost přenosu je vysoká. Proto se data většinou nejprve nasnímají a až pak se provede analýza, která zahrnuje klasifikaci, dekódování a prezentaci dat. Výsledkem může být detekce a zobrazení obsahu ARP datagramů, které vysílá špatně nakonfigurovaný přepínač. Používají se nástroje jako Wireshark, Microsoft Network Monitor či tcpdump. Ty umožňují i jednoduchou analýzu odchycených dat (interpretaci hlaviček protokolů).
K programování aplikací, které komunikují přímo se síťovým rozhraním a čtou všechna příchozí data, lze použít síťové knihovny libpcap. Sledování v reálném čase je náročné na procesor. Pokud data ukádáme, tak i na prostor.

Obvykle nás zajímají i dlouhodobé statistiky získané analýzou provozu - sbíráme hodnoty a ukládáme je. Sběr monitorovacích dat: SNMP, RMON, NetFlow. Sledování stavu linek, počtu přenesených dat, apod. Vyhledání nejvíce komunikujících uzlů (top ten hosts). Statistiky přenosů TCP/UDP/ICMP, unicast/multicast/broadcast.

## Cíle analýzi a monitorování

Výsledkem můžou být statistické údaje ve formě grafu, které pomáhají detekovat slabá místa v síti. Dále údaje důležité pro plánování dalších investic. Lze sledovat i nejvíce komunikující stanice či protokolové statistiky.

### *Správa sítě na úrovni IP*

Pro správu sítě na úrovni IP vrstvy se používá protokol ICMP. Je součástí implementace protokolu IP (je implementován v každém TCP/IP stacku síťových zařízení). Používají jej PC, směrovače či síťové prvky pro předávání síťových informací - chybových hlášek, protože protokol IP nezaručuje spolehlivé doručení. V případě zahození není uživatel informován o nedoručení IP datagramu, to zaručí ICMP. Protokol pracující na IP vrstvě, musí jej podporovat všechna L3 zařízení. Například u zadání špatné URL webu, ICMP vrátí hlášku, kterou prohlížeč interpretuje. Informuje též o ztrátě paketů například v důsledku směrování, zahlcení front na směrovači a zahození IP datagramu. Tento protokol užívají PING a TRACEROUTE.

## Funkce a struktura ICMP

Slouží k detekci chyb, detekuje nedostupnost zařízení. Poskytuje informace o zahlcení, umožňuje přesměrovat data při výpadku implicitní brány či sleduje stav linky. Struktura ICMP je podobná IP datagramu. Je to speciální IP datagram. V těle jsou informace pro ICMP, typ protokolu je 1. Tělo obsahuje typ ICMP zprávy, kód chyby, prvních 64 bitů původního IP datagramu, nakterý ICMP reaguje.

| Zpráva | Typ | Popis |
|---|---|---|
| Destination Unreachable | 3 | Cílová síť je nedostupná |
| Time Exceeded Message | 11 | TTL dosáhlo hodnoty 0 |
| Parameter Problem Message | 12 | chyba při zpracování IP hlavičky |
| Source Quench Message | 4 | přeplnění bufferu brány, kontrola zahlcení |
| Redirect Message | 5 | informace o přesměrování datagramu (brána) |
| Echo, Echo Reply | 8,0 | data posílaná pomocí Echo se vrátí |
| Timestamp, Timestamp Reply | 13,14 | posílání časového razítka |
| Info Request, Info Reply | 15,16 | zjišťování adresy počítače |

Mezi nejčastější zprávy patří Destrination Unreachable, dále zpráva 11 a zprávy Echo 8.

## Aplikace využívající ICMP

Nástroje pro monitorování pomocí ICMP:

Ping - posílá zprávy ICMP Echo (type 8, code 0) na cílovou adresu. Vzdálený uzel odpovídá zprávami ICMP Echo reply (type 0, code 0).
Traceroute - posílá IP pakety se zvyšujícím se TTL, očekává zprávy ICMP typu Time Exceeded Message (type 11, code 0). Pro každý odeslaný paket se zapne časovač a čeká se na zprávu ICMP. Když n-tý paket dojde na n-tý směrovač, TTL se sníží na 0. Směrovač zahodí paket a pošle odesílateli zprávu ICMP typu 11. Po přijetí zprávy vysílající zjistí čas přenosu a IP adresu vzdáleného směrovače, který jej zahodil.

Za bezpečnostní rizika se dají považovat prohledávání sítě pomocí zpráv ICMP Echo (ping) a podvržení implicitní brány (ICMP Redirect Message).

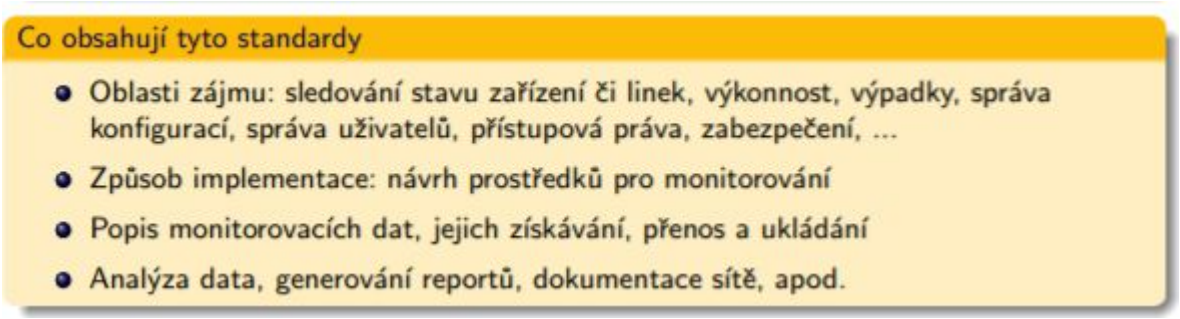
U IPv6 jsou ICMP zprávy zapozdřeny v protokolu IPv6, next header 58.

### *Standardy pro správu sítě*

Prvním byl standardem byl SMAE, který umožnil správci sítě vzdáleně spravovat síťové objekty (protokoly, přepínače, směrovače) nezávisle na typu a architektuře zařízení či sítě. Součástí bylo

rozhraní služeb pro správu sítě CMISE a přenosový protokol CMIP. CMISE a CMIP se staly základem pro standard IETF SNMP, který patří dodnes k nejvíce rozšířeným systémům pro správu sítě.

Dále se využívá model FCAPS a Internet Manafement using SNMP (popis monitorovacích objektů SMI, struktura dat MIB, protokol SNMP).




## Model FCAPS

Standard ITU-T definuje základní požadavky na správy sítě pomocí tzv. modelu FCAPS:

- Správa poruch - zajišťuje detekci, izolaci a opravu poruch. Porucha je obvykle charakterizována jako neobvyklý stav vyžadující zásah správce. Například fyzické porušení linky. Správa poruch zahrnuje logování událostí,detekce poruch a reakce.
    - Když se objeví porucha, potřeba provést tyto akce: Určit, kde je porucha, izolovat poruchu v síti tak, aby ostatní části mohly pracovat, upravit síť tak, aby to mělo minimální dopad na zařízení postižené poruchou, opravit či nahradit porouchané zařízení a nastavit počteční bezchybový stav.
    - Systémy pro monitorování poruch by měly sloužit i k izolaci poruch a diagnostice. Příklad testů: test konektivity, integrity dat, odezvy, funkčnosti.
- Účtování - zajišťuje správu uživatelských účtů a poplatků za služby. Obsahuje specifikaci, logování a řízení přístupu uživatelů a zařízení k síťovým zdrojům. Definuje uživatelské kvóty, alokaci prostředků. Statistiky přenosů, identifikace přenosů
- Správa konfigurací - správcemprováděné sledování připojených zařízení a jejich konfigurace (jméno softwaru). Konfigurace se mohou měnit během aktualizace a rozšiřování sítě. Instalace fyzických zařízení. V DB konfigurací monitorovaných objektů jsou základní informace o zařízení (typ, verze, procesor, paměť). Správa konfigurace zahrnuje tyto informace: konfigurační data (názvy zdrojů a jejich atributy), nastavení a modifikace atributů objektů, inicializace a přerušení síťové operace (vzdálené spuštění webového serveru, nastavení filtrování na směrovači), distribuce softwaru (SK by měla zajišťovat i DS na koncové stanice, s tím souvisí správa verzi, zálohování softwaru a nastavení systémů a aplikací).
- Správa výkonnosti - zaměřuje se na 2 oblasti: monitorování a řízení. Monitorování sleduje chování na síti, sbírá informace o stavu monitorovaných objektů, pak analyzuje a vyhodnotí. Monitorují se metriky dostupnost, odezva (doba, za kterou systém odpoví), propustnost (rychlost, při které se objevují události) a využití (parametr vyjadřující v % poměr využívané kapacity vůči teoretické hodnotě, využití přenosového pásma). Měří se kvalita přenosů, monitorují se prahové hodnoty a vytváří se charakteristické chování sítě (baseline). Řízení výkonnosti a plánování zdrojů.
- Bezpečnost - se zaměřuje na řízení přístupu k síťovým zdrojům dle bezpečnostní politiky (stanovuje hrozby). Zahrnuje distribuci klíčů a certifikátů, používání firewallu atd. Zabezpečení zařízení, sledování událostí, audit. Nástroje: Autentizační služby (802.1x), sítě

VPN (IPsec, SSL). V případě události reaguje, což může být hlášení adminovi (systémy IDS), nastaví protiopatření (filtruje útočníkův provoz).

### *Architektura SNMP*

Správa sítích vytvořených nad TCP/IP byla založena zpočátku jen na protokolu ICMP, ale s roustoucím počtem uzlů v síti nebylo možné kontrolovat stav zařízení jednoduchými dotazy ICMP a bylo potřeba systému pro správu sítí.

Architekturu SNMP tvoří 4 základní prvky: **řídící stanice NMS** (serverová aplikace a tvoří jí nástroje pro sběr a analýzu dat, ukládání statistik a prezentaci stavu sítě), **Agent na monitorovaném zařízení** (aktivní proces sbírající provozní informace, odpovídá na dotazy řídící stanice, zasílá jí informace, může sám poslat zprávy typu *trap*), **DB monitorovaných objektů MIB** (soubor všech objektů na monitorovaném zařízení. Objekt je datová struktura, která reprezentuje sledované informace, řídící stanice monitoruje síť načítáním vybraných objektů ze sledovaných zařízení, může je měnit), **přenosový protokol SNMP** (slouží k předávání zpráv o sledovaných objektech mezi řídící stanicí a agentem, protokol obsahuje 3 základní typy příkazů: *get* (vyžádá si hodnoty daného objektu od agenta), *set* (nastaví hodnotu objektu na agentovi), *trap* (agent řídící stanici ohlašuje důležitou událost na monitorovacím zařízení).

Systém správy SNMP tvoří **jazyk SMI** pro popis objektů, který obsahuje pravidla pro vytváření monitorovacích objektů. Je podmnožinou jazyka ASN.1 pro výměnu informací u heterogenních systémů (komunikace zařízení od jiných vyýrobců). Definuje datové typy pro objekty (integer, octet string, object identifier, null) a způsob vytváření. Popis objektu tvoří jméno objektu, což je jednoznačný identifikátor OID, dále syntax, který definuje abstraktní datový typ spojený s objektem (integer, ocetet string, object identifier, null), a kodování pro přenos po síti (používá se BER).

**Kódování objektů BER** definuje reprezentaci hodnot objektů při přenosu po síti. Každá přenesená hodnota se zapisuje pomocí struktury TLV (type-length-value), která slouží nejen k přenosu vlastní hodnoty, ale i k identifikaci obsahu. Přijímací stanice pak ví typ přijatých dat a kolik paměti vyhradit. Strukturu lze využít rekurzivně například u složených objektů.

Dále systém správy SNMP tvoří **identifikace objektů pomocí OID**, **uspořádání databáze objektů MIB**, **způsob kódování objektů** popsaných ASN.1 pro přenos a **komunikační protokol SNMP**.

**MIB databáze** obsahuje jednotlivé objekty a skupiny monitorovaných. Původní MIB databáze obsahovala 8 skupin s celkem 114 monitorovanými objekty. Základní skupiny byly systém, interface, address translation, ip, icmp, tcp, udp, egp. Dle názvů vidíme, které informace jsou pro správu sítí nejdůležitější. Tato databáze rozšířena na základě požadavků. Například o skupinu transmission (zohledňuje typy přenosových médií). Každá skupina obsahuje množinu objektů, které obsahují další proměnné. Nejčastěji sledované objekty jsou ip a interface. Databáze MIB je strukturována hierarchicky ve formě stromu, kde listy jsou konkrétní monitorované objekty, ty jsou identifikovány pomocí číselného identifikátoru OID.

### Komunikační protokol SNMP

Aplikační protokol pro práci s monitorovanými objekty, nestavový protokol typu dotaz-odpověď. Využívá vyzývání (port 161) a asynchronních zpráv typu trap (port 162).

### Praktické nasazení systému SNMP

Parametry ovlivňující výkonnost systému SNMP - počet monitorovaných agentů, frekvence dotazování, objem přenášených dat -> intenzivní monitorování může významně zatížit zařízení i síť.

Omezení protokolu SNMP - chybí potvrzování -> nelze detekovat ztrátu zpráv trap. SNMPv1 a v2 využívá pouze jednoduché autorizace -> využít pokročilého zabezpečení SNMPv3.

SNMP - nelze vykonávat příkazy na monitorovaném zařízení, monitorování omezeno na L1 až L4, SNMP monitoruje různá zařízení, zpracování je na jedné stanici NMS, umí zjistit stav daného zařízení, chybí celkový pohled na síť -> potřeba decentralizace a komplexního pohledu na síť -> rozšíření RMON.

# Přednáška 11

## Network and system logging

### *NTP protokol*

Pro synchronizaci PC času ve všech PC. Je to důležité pro dohledávání například stížností, logů problémů. Časový formát může být vždy jiný (notice date, timestamp). Přesný čas je potřeba i pro provoz sítě - zjištění přesného času události. Potřebné i pro synchronizaci událostí. Best practice je, když jsou všechna zařízení na UTC (universal time). Je to protokol pracující nad UDP, port 123. NTP server referencuje na externí hodiny, posílá z nich čas. Ruční nastavení času není dostatečně přesné pro síťové zařízení. Nejlepší způsob je použití NTP.

NTP protokol má také hierarchii serverů podobně jako DNS - úrovně = stratum. Stratum 0 - atomové hodiny, které jsou přesné. Stratum 1 až N se synchronizují se stratum 0 (1 od 0, 2 od 1 atd.), zvyšuje se s každou úrovní nepřesnost kvůli zpoždění sítě, to je v řádu ms, takže je to ještě pořád přesné.

NTP protokol a přesný čas z něj je potřebný i pro zjištění, kdy nám vyprší certifikáty, klíče (autentizace klíče, klíčové řetězce). Pokud 1 zařízení NTP je mimo synchronizaci s ostatními, ostatní se odmítnou synchronizovat s ním, což je obrana proti rozbití synchronizace. To znamená, že nesynchronizované zařízení si může od nich čas vzít, ale oni si čas od něj nevezmou.

Zařízení - server, klient. Peers jsou servery stejné úrovně mezi sebou. NTP servery již byly zneužity i pro DDOS útoky.

### *Syslog*

Protokol jako standard pro poskytování komplexního mechanismu podávání zpráv (pro logování). Má jej řada zařízení - PC, tiskáry, servery. Je standardem pro posílání informací nebo logů. Pracuje nad UDP na portu 514 (reálně se TCP moc nepoužívá). UDP, neboť se nechceme zabývat vytváření spojení.

Logy jsou posílány v plain textu, neboť se typicky zasílá přes lokální síť, přes síť zřídkavě. U logů rozlišujeme stupně důležitosti informace - 0 až 7, kde 0 je nejvíce důležitá. 7 je nejméně, jedná se o debugging (informace, že se něco stalo), 6 je informační, 5 je poznámka, 4 je varování, 3 je chyba, 2 je kritická událost, 1 je upozornění a je pohotovost.

Formát syslog zpráv se skládá z facility (2 a více místný kód pro označení, odkud pochází zpráva (hardware, protokol), severity (stupeň důležitosti, 1 znak), mnemotechnická (unikatně identifikuje error message) a samotný text zprávy (popis situace).

Obecně se Syslog užívá pro logování aplikací a služeb (DNS, DHCP, RADIUS, WEB, EMAIL). Logovací zprávy mohou být uloženy v lokálních log souborech nebo poslány do vzdáleného zařízení.

### *SNMP*

Obsahuje 3 části: agent (na monitorovaném zařízení), NMA (network management app, řídící stanice) a databáze MIB (popisuje informaci, co poslal agent k NMA).

SNMP módy - pull model (NMA periodicky vyzývá agenta k poslání objektů), push model (agent informuje NMA o určitých událostech).

Objekty u agenta mají jednoznačný identifikátor OID (Object identifier), objekty uloženy v hierarchické stromové struktuře, každý uzel má jméno + číslo, do OID se počítají čísla předků + číslo objektu.

#### *Nástroje pro přenos souborů (konfigutací)*

Například TFTP (malá efektivní binárka), FTP, SFTP (bezpečnější a robustnější), HTTP, REST API. Přenášejí se konfigurace, zálohy a uploadují se a zálohují se firwary. Někdy pro synchronizaci databáze interních switchů nebo routerů (DHCP binding).

#### *Analýza logů (abychom byli schopní problém dohledat)*

Je třeba nějak logy uchovávat, protože je to zákonem dané. Uložiště logů obsahuje velký počet logových souborů, navíc velikost logů se zvětšuje, protože se loguje víc a víc informací. Velká množství dat se těžce interpretují a monitorují, je tedy třeba je nějak logovat a označovat. Užívají se jednoduché nástroje pro pomoc s analýzou debuggingu, monitoringu a správy chyb. Podporují agent monitoring, SNMP monitoring, Graphs reporting, upozorňování. Například zabbix nebo nagios.

#### Agent

Je software na zařízení posílající data (logy) o zařízení SNMP serveru

#### SNMP monitoring

Užíván a implementován u směrovačů a routerů. Přes OID se dotazuje agenta na něco z MIB. Získávají se informace od agenta o přenesených datech, CPU zatížení, době běhu, hostnamu atd.

#### Konfigurace monitorování a zálohování

Je lepší mít přístup ke konfiguraci celé sítě na jednom místě, protože je pak lehčí hledat a najít poslední configy. Revize historie je užitečná pro trouble shooting (řízení chyb), užívají se nástroje - rancid a oxidized.

## Přednáška 12

### Monitorování toků NetFlow

NetFlow se dá přirovnat k sledování silniční dopravy - lze zjistit vytíženost silnic, sledovat zdroj a cíl jednotlivých tahů, parkování atd. Dle úrovně detailů je možné zjistit aktivitu osob v dané oblasti, lze identifikovat oblastni ucpání vozovek a neobvyklé události, lze sledoat konkrétní auto na cestě z místa A do místa B (cíle), lze také stanovit denní aktivitu na silnicích, v případě trestného činu dohledat záznamy pro dohledání pohybu pachatele.

Přesně jako u sledování provozu na silnicích lze sledovat a monitorovat tok dat. Logují se metadata o každém paketu, který prošel sítí. Záznamy o tocích se využívají pro správu síťového provozu.
V případě bezpečnostího incidentu lze identifikovat komunikaci v daném čase.

#### *Síťový tok (Flow)*

Tok je posloupnost paketů mající společnou vlastnost a prochazející bodem pozorování za určitý časový interval. Záznam o toku obsahuje informace o toku - zdrojovou a cílovou IP adresu, zdrojový a cílový port, typ protokolu, datum a čas zaznamenání, počet přenesených dat toku.

Router si zaznamenává Flow cache komunikace mezi body, když jde paket od stejného zdroje na stejný cíl (shodují se hlavní pole v tabulce Flow cache), tak se jen inkrementuje počet paketů a velikost dat toho toku. Pokud takový tok ještě není, tak se vytvoří nový řádek v tabulce. Pakety jdoucí jedním a druhým směrem tvoří 2 toky. Pokud se tyto dva toky spojí, tak máme biflow. Na to je třeba

ale mít podporu. Z paketu se vlastně vypočítává hash, podle kterého se zjistí, na který případně existující řádek paket patří.

Záznamy NetFlow se exportují v krátkých intervalech - monitorování provozu.

### *Cisco NetFlow*

Je to otevřený protokol vyvinutý společností Cisco pro monitorování sítí využívající informace o tocích. Provádí analýzu metadat - získávání statistik z hlaviček paketů.

### *Architektura NetFlow*

#### Exportér

Je to sonda nebo router pro získávání statistik o tocích. Je to zařízení (nebo SW), které monitoruje procházející provoz. Vytváří záznamy o tocích (Flow records). Vytváří nový nebo aktualizuje starší záznam v paměti NetFlow cache. Záznamy jsou uloženy s určitou expirací, může se provádět agregace dat. Export pomocí UDP, což znamená možnou ztrátu dat.

Export záznamů o toku probíhá, když je detekce konce toku (u TCP příznak RST nebo FIN) nebo když je neaktivita toku (neaktivní timeout) nebo když je příliš dlouhý tok (aktivní timeout) nebo když je zaplněná paměť NetFlow cache.

Data se vzorkují podle pořadí, času a velikosti. Důvodem je snížení nároků na hardware - 1:100 snížení nároků přibližně o 75 %. Flow filtr - vzorkování podle provozu, není tak zatížená cache. Vzorkování může být použito jak u exportéru, tak kolektoru.

Filtrování dat - klasifikace provozu na základě hodnot v hlavičce paketů. Nezávisí na pořadí, času apod.

Slučování dat podle klíčových položek, nové záznamy se ukládají do aggregation cache.

#### Komunikační protkol NetFlow

Verze 5 - obsahuje hlavičku a záznam o tocích (src, dst IP a port, ext hop router - pro kontrolu směrování, počet paketů, bajtů - pro účtování, příznaky u TCP paketů, protokol, ToS). Verze 8 umožňuje agregaci toků na směrovači. Verze 9 umožňuje nadefinovat, které informace chci v paketu přenášet (šablony). Verze 10 vychází z verze 9 a standardizuje a nahrazuje ji.

#### Kolektor

Je to nějaký software (netflowtracker, nfdump, calligare), který přijímá pakety NetFlow z jednoho či více exportérů, zpracovává záznamy o tocích a případně agreguje data. Dále ukládá statistiky na disk nebo do databáze. Dotazování nad daty - grafická prezentace dat, skriptové zpracování nebo webové zpracování.

#### Nástroje pro zobrazení dat

Grafy a statistiky.

### *Použití NetFlow*

Sledování toků a provozu konkrétních uzlů (nikoliv obsah komunikace), plánování sítě (detekce provozu), detekce útoků v reálném čase (DoS, viry, červy), dlouhodobé ukládání informací o přenesených datech (ze zákona půl roku pro možnosti vyšetřování zločinů), sledování a analýza aplikací/uživatelů (kdo nejvíc vytěžuje síť, jaké aplikace se používají), účtování (měření IP provozu, počet přenesených dat, kontrola dodržování SLA), vyvažování směrování mezi ISP (optimalizace směrování).