# Rozumíme počítačovým sítím

Neděle, 09.05.2010 21:27 | Samuraj - Petr Bouška | sítě

Počítačové sítě, obecný termín, který zahrnuje velice širokou oblast technologií, protokolů a metod. Mnoho věcí z této oblasti je považováno za samozřejmost, což vede k tomu, že se těžko hledá jednoduchý popis či logické zařazení do celé skládačky, která tvoří síť. My se dnes na sítě podíváme obecným globálním pohledem s přihlédnutím k vzájemným vazbám. A v dalších dílech se podrobněji seznámíme s některými oblastmi z praktického hlediska.



*Tento článek jsem napsal pro časopis Connect a vyšel v čísle Connect 04/10, zde jej publikuji s laskavým svolením redakce.*
*Jedná se o první díl seriálu o počítačových sítích. Na tomto webu již existuje obsahově shodný (a rozsáhlejší) seriál Počítačové sítě - Computer networks, ale tento článek jsem napsal s určitým časovým odstupem a z trochu jiného pohledu.*

## Síťové technologie přehledně a stručně

Na začátku bychom si měli určit, co taková počítačová síť (dále budeme používat pouze termín síť) je. Běžně používaná definice říká, že ***počítačová síť vznikne spojením dvou nebo více počítačů tak, aby mohli navzájem komunikovat a sdílet zdroje*** (informace, software, zařízení, apod.).

Sítě můžeme dělit podle velkého množství parametrů. Dle velikosti, rozlohy a účelu máme Local Area Network (LAN), Metropolitan Area Network (MAN), Wide Area Network (WAN), Virtual Private Network (VPN), Storage Area Network (SAN) a mnoho dalších. Podle síťové topologie máme zapojení jako sběrnice (bus), hvězda (star), kruh (ring), mřížka (mesh) apod. Podle technologie zapojení máme drátové (wired) sítě, které mohou používat kroucenou dvojlinku (twisted-pair), koaxiální kabel, optické vlákno, a bezdrátové (wireless).

Dále můžeme sítě dělit podle technologie komunikace na Ethernet, Token Ring, FDDI, Frame Relay, atd. A případně i podle použitého komunikačního protokolu, nejrozšířenější skupina protokolů TCP/IP (Transmission Control Protocol/Internet Protocol) a další jako IPX/SPX (Internetwork Packet Exchange/Sequenced Packet Exchange), AppleTalk.

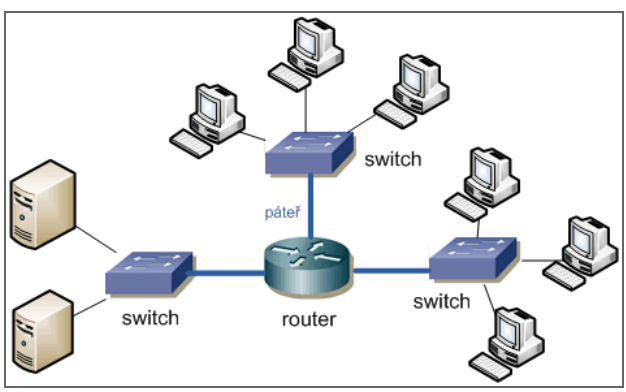
My se dále budeme hlavně věnovat tomu, co je nejrozšířenější a nejběžnější, tedy lokálním sítím LAN s technologií Ethernet, používající protokoly z rodiny TCP/IP a v některých oblastech ještě s přihlédnutím k použití kroucené dvojlinky v zapojení do hvězdy.

## Aktivní prvky a kabeláž

Síť stavíme z určitých fyzických prvků. Jednou částí jsou samozřejmě počítače (stanice, servery, tiskárny a další síťová zařízení), kterým chceme umožnit komunikaci. Druhou jsou zařízení, která zajišťují propojení, to je strukturovaná kabeláž (jinak řečeno pasivní prvky) a aktivní síťové prvky.

Do infrastruktury ***strukturované kabeláže*** patří horizontální i páteřní kabeláž, konektory, patch panely, datové (komunikační) a serverové rozvaděče (racky), telekomunikační zásuvky, apod. Mezi ***aktivní prvky*** patří zařízení, která odesílají, přijímají a přeposílají informace přes komunikační kanál. Jedná se o síťové adaptéry (Network Interface Controller - NIC), přepínače (switch), směrovače (router), přístupové body bezdrátové sítě (WiFi AP), případně i starší zařízení jako opakovače (repeater), rozbočovače (hub), mosty (bridge) a další.

Použité aktivní prvky a komunikační protokol určují, jakým způsobem data fyzicky putují po síti. Připojení na sdílené médium (mimo bezdrátové sítě) dnes asi již nikdo nepoužívá, takže pro komunikaci v rámci stejného subnetu se používá ***přepínaný ethernet*** s využitím switchů (ten je při komunikaci transparentní, nemění odesílaná data). Pro doručení dat do jiného subnetu se používá ***směrování*** pomocí routeru (ten se musí při komunikaci přímo adresovat a upravuje odesílaná data).




## Komunikační model

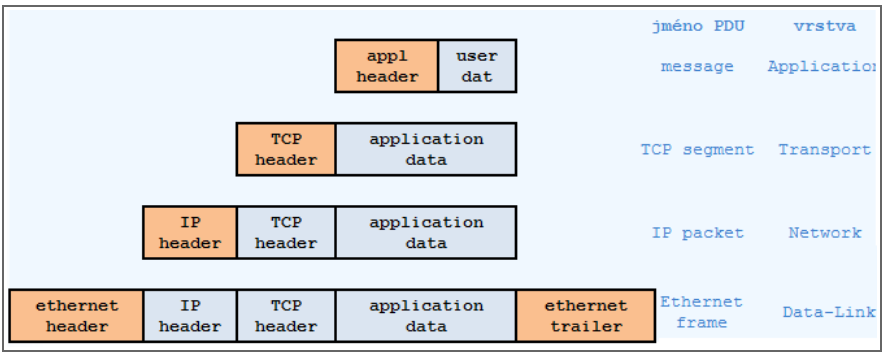
Pro podrobnější pohled na síťovou komunikaci můžeme použít standardizovaný abstraktní popis, který se označuje jako ***OSI model*** (Open Systems Interconnection Basic Reference Model). Tento model se skládá ze sedmi vrstev a každá vrstva zajišťuje určitou část komunikace či služeb. OSI model ovšem pro praxi nenabízí potřebnou volnost, přesně definuje vrstvy a každé vrstvě povoluje komunikaci pouze se sousední nad či pod. To by pro praxi přinášelo zbytečnou zátěž.

V praxi nejpoužívanější soubor protokolů se jmenuje Internet Protocol Suite (většinou se označuje jako TCP/IP). Tato sada protokolů používá vlastní, upravený model, který se označuje jako **TCP/IP Reference Model** (případně

také Internet Model). Ten se skládá pouze ze čtyř vrstev, nazvaných linková (link), síťová (internet), transportní (transport) a aplikační (application), často se ovšem setkáme i s jinými názvy. Horní vrstvy jsou blíže uživateli a pracují s více abstraktními daty, nižší vrstvy upravují data do formy, která se dá fyzicky přenášet.

Při odesílání začne pracovat nejvyšší vrstva, ta má uživatelská data, doplní je o hlavičku a předá nižší vrstvě. Tehdy se provede ***zapouzdření*** (encapsulate), z dat a hlavičky vyšší vrstvy se stávají data nižší vrstvy (zabalují se). Do hlavičky se doplňují komunikační údaje a další informace pro identifikaci dat na dané vrstvě. Pokud jsou uživatelská data velká, tak se také provádí rozdělení na segmenty (na 3. vrstvě).

Data na 2. vrstvě se označují jako **paket** (packet - balíček) a v případě IP protokolu obsahují IP adresu zdrojové a cílové strany. Na síť data odesílá 1. vrstva, která v hlavičce doplní MAC adresy a tehdy se z paketu stává **rámec** (frame). Měli bychom tedy rozlišovat, kdy mluvíme o paketech (například při směrování, které používá data z 2. vrstvy) a kdy o rámcích (například při přepínání, používají se údaje z 1. vrstvy). V praxi se ale často oba termíny zaměňují. Můžeme také používat obecný termín ***PDU*** (Protocol Data Unit).




## Sítě a adresování

Řekli jsme si, že budeme převážně mluvit o lokálních sítích. Mezi sítě ***LAN*** se většinou počítá síť, která se nachází na menším (omezeném) geografickém území (může se jednat o budovu případně areál nebo jen patro či místnost). LAN je sama o sobě uzavřená síť, ale většinou spojujeme více LAN sítí dohromady. Buďto napřímo nebo přes větší vzdálenosti pomocí WAN (internetu či privátních linek).

Největší a nejznámější (WAN) síť postavená na TCP/IP je ***internet***. Zjednodušeně můžeme říci, že se jedná o velké množství vzájemně propojených LAN. Situace je ale o něco komplikovanější. V internetu se používá adresování pomocí IPv4 (Internet Protocol Version 4) a k tomu se pomalu začíná prosazovat IPv6. Z řady technických důvodů není možné, aby všichni účastníci této sítě komunikovali napřímo mezi sebou.

Takže z principu komunikace, ale také kvůli přidělování rozsahů adres, celou síť rozdělujeme hierarchicky na menší části, těmto částem říkáme ***podsítě*** (subnet). Protože se jedná o hierarchické dělení, tak jednotlivé podsítě můžeme znovu dělit na ještě menší subnety. V případě internetu je hlavní adresní prostor spravovaný organizací IANA (Internet Assigned Numbers Authority). Dále se dělí na velké bloky, které jsou přiřazeny regionálním registrátorům RIR (Regional Internet Registries), ti spravují adresy pro různé části světa. RIR přiděluje menší bloky adres poskytovatelům služeb ISP (Internet Service Providers) a velkým organizacím. Od ISP pak může získat menší blok adres nějaká organizace.

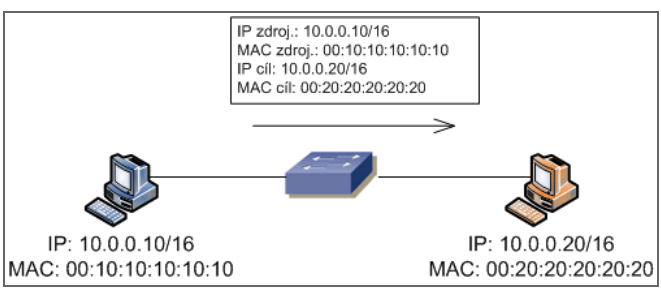
V rámci organizace většinou nepoužíváme pouze veřejné IP adresy přidělené od ISP, ale také neveřejné rozsahy (ty by v rámci internetu neměli komunikovat). Celá síť firmy, kterou můžeme označovat jako LAN, se v praxi skládá z řady subnetů. Do jiných subnetů patří sítě, které chceme vzájemně oddělit (a také je logicky nebo fyzicky oddělujeme), třeba provozní, testovací, DMZ. Ale rozdělení na menší subnety využíváme i z důvodu výkonu, protože rozdělíme broadcastové domény (logická část sítě, kde všechny zařízení komunikují přímo na linkové vrstvě a přijímají vzájemné broadcasty).

V dávnějších dobách se pro vytvoření oddělených sítí museli používat různé aktivní prvky (a tedy budovat infrastrukturu vícekrát). V dnešní době je ovšem naprosto běžné využití ***VLAN*** (Virtual LAN), které dovolí provozovat více oddělených sítí (subnetů) v rámci jednoho (samozřejmě i více) aktivního prvku. Zjednodušeně řečeno, switch můžeme rozdělit na několik logických switchů tak, že určité porty zařadíme do určité VLANy. Při tom je zajištěno, že není možno komunikovat přímo mezi VLANami.

## Komunikace v rámci subnetu

Zařízení, která jsou připojena do stejného subnetu, spolu mohou komunikovat napřímo. V případě přepínaného ethernetu to znamená, že pro propojení se použije jeden nebo více switchů. Pro komunikaci se použijí údaje z první vrstvy dle TCP/IP modelu, tedy v případě ethernetu jde o **MAC** (Media Access Control) **adresu**, což je fyzická adresa rozhraní.

Switch si v paměti vytváří tabulku (CAM - Content Addressable Memory), za jakým portem se nachází jaká MAC adresa. Potom komunikaci určenou nějaké MAC adrese posílá přímo na daný port. Pokud ještě nemá záznam, tak data odešle na všechny porty mimo příchozího. CAM tabulka se vytváří z příchozích dat.



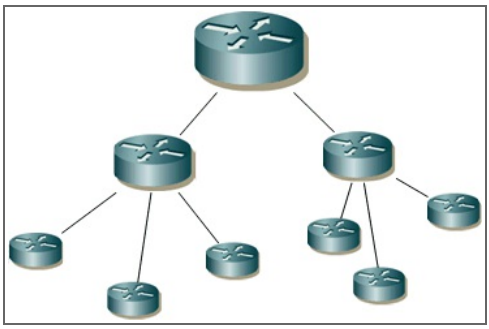

Na switchích se řeší několik různých problémů, hlavním jsou smyčky ve fyzickém propojení. Ty mohou vést ke vzniku broadcastových bouří (a dalších problémů). Na obranu proti smyčkám můžeme běžně použít Spanning Tree Protocol (STP), který blokuje redundantní cesty.

## Propojení subnetů

Tím, že jsme provozní síť rozdělili na menší subnety a zařadili do různých VLAN, jsme sice zmenšili množství

broadcastů, které každé zařízení zpracovává. Ale také jsme ztratili možnost komunikace mezi zařízeními v různém subnetu. Aby spolu zas mohli komunikovat, musíme připojit zařízení, které bude provádět ***routování*** a pro jednotlivé subnety bude výchozí branou (default gateway). V praxi nejčastěji využijeme tuto funkci na L3 switchi případně routeru. Komunikaci mezi sítěmi odlišného charakteru, jako je provozní síť a DMZ (Demilitarized zone), většinou zajišťuje firewall. Ten zároveň tvoří bezpečnostní bariéru mezi sítěmi.

Tak jak jsme mluvili o rozdělení sítě na subnety, tak tomu odpovídá umístění routerů, které se nachází v každém bodě, kde spolu mají komunikovat jednotlivé subnety. Dohromady tvoří hierarchickou strukturu, zjednodušeně strom. Když spolu mají komunikovat dvě stanice v různých sítích, tak komunikace musí projít přes jeden nebo více routerů. Délku označujeme jako počet hopů (skoků), tedy počet zařízení, přes která musí komunikace projít (přímé propojení dvou PC je jeden hop).



Za to, jakou cestou bude komunikace směrována (na jaké další zařízení), jsou zodpovědné routery. Ty používají určitý ***routovací protokol***, který sestavuje routovací tabulku a podle ní se provádí rozhodnutí o směrování na další hop. Routovacích protokolů existuje celá řada, ať již standardizovaných nebo proprietárních. Příkladem je RIP (Routing Information Protocol), EIGRP (Enhanced Interior Gateway Routing Protocol), OSPF (Open Shortest Path First) či BGP (Border Gateway Protocol).

Použití routovacího protokolu není vždy potřeba. Ten se používá pro ***dynamické routování***, které automaticky detekuje změny v síti a upravuje routovací tabulku. V LAN nám často stačí použít statické routování, kdy záznamy do tabulky zadáme buď ručně, nebo se vytvoří automaticky z přímo připojených rozhranní (subnety, které jsou připojeny na tomto routeru).

## Metody vysílání

K předchozímu popisu komunikace je ještě třeba doplnit, že máme tři typy komunikace a podle nich se mění detaily v doručování. Jde o to, zda chceme doručit data pouze jednomu příjemci. Tato metoda je nejběžnější a označuje se jako **unicast**. Tomu se věnovala většina předchozího popisu.

Potom máme vysílání jednoho všem, označuje se jako **broadcast** (všesměrové vysílání). To by měla obdržet všechna zařízení v rámci broadcast domény (stejného subnetu). Switch takovou komunikaci odesílá na všechny porty mimo příchozího. Toto vysílání využijeme nejen tehdy, když chceme poslat informaci všem okolo, ale také když neznáme adresu příjemce. Ten přijme broadcast, podle obsahu pozná, že je určen jemu a odešle odpověď. Takovým příkladem jsou protokoly DHCP či ARP. Velké množství broadcastů ale zpomaluje síť a zatěžuje stanice.

Ještě máme **směrovaný broadcast** (Subnet Directed Broadcast), ten by měl po síti doputovat do cílového subnetu a teprve tam se rozeslat jako broadcast. V praxi je však tento způsob z bezpečnostních důvodů často blokován.

Poslední možnost je vysílání určené skupině příjemců, tzv. **multicast**. To má zajistit efektivní doručení dat několika různým příjemcům tak, aby data v síti putovala pouze jednou.

## Síťové protokoly

Ještě si doplníme krátkou informaci k protokolům. Zde se bavíme o **Internet Protocol Suite**, souboru síťových protokolů, kterému se často říká TCP/IP. To ovšem může být zavádějící, protože do této skupiny patří mnohem více protokolů. ***IP*** je sice hlavní komunikační protokol na 2. vrstvě TCP/IP modelu, který zajišťuje doručování paketů. Ale existují zde i další důležité protokoly, jako je ICMP (Internet Control Message Protocol, zajišťuje v síti třeba informace o nedostupnosti cílové sítě nebo echo request/reply) či IGMP (Internet Group Management Protocol, zajišťuje správu skupin pro multicast).

Stejně tak na 3. vrstvě TCP/IP modelu je důležitý ***TCP***, protokol, který operuje nad IP a zajišťuje rozdělení odesílaných dat na segmenty a jejich spolehlivé doručení příjemci ve správném pořadí. I na této vrstvě máme další protokoly, hlavním zástupcem je ***UDP*** (User Datagram Protocol), který také pracuje nad IP. Ten používá jednoduchý vysílací model, který nezajišťuje spolehlivost doručení (data nemusí dorazit, mohou dorazit vícekrát či v jiném pořadí), ale má malou režii.

Na 4. vrstvě TCP/IP modelu jsou již aplikační protokoly. Nad UDP pracuje například DNS, TFTPa VoIP. Nad TCP pracuje velké množství protokolů, namátkou HTTP, SMTP, FTP. Do rodiny IP protokolů patří i několik protokolů z 1. vrstvy TCP/IP modelu, například MAC (Media Access Control, používaný ethernetem), ARP (Address Resolution Protocol) či PPP (Point-to-Point Protocol).



Autor: Petr Bouška

**Související články:**

## Základy počítačových sítí

Tento seriál jsem napsal pro časopis Connect. Většinu informací obsahuje shodně jako můj starší seriál Počítačové sítě - Computer networks, ale je napsán trochu jiným způsobem. Nejprve jsou stručně shrnuty technologie počítačových sítí a pak se trochu podrobněji probírají od nejnižších vrstev nahoru.

- Rozumíme počítačovým sítím [09.05.2010 21:27] ..... **právě čtete**
- Víte, jak pracuje switch? [25.05.2010 10:39]
- Víte, jak pracuje router? [23.06.2010 18:30]
- Adresování v IP sítích [21.07.2010 12:39]

# Víte, jak pracuje switch?

Úterý, 25.05.2010 10:39 | Samuraj - Petr Bouška | sítě

Když chceme pochopit, jak fungují komunikace v počítačových sítích nebo obecně sítě celé, tak musíme začít od spodu. Z fyzického pohledu se jedná o kabeláž a základní aktivní prvky (switche), z logického pohledu se jedná o komunikace na nejmenším logickém celku, tedy v rámci subnetu (přímá komunikace). Obě tyto oblasti jsou definovány a zahrnuty v rodině technologií Ethernet.



*Tento článek jsem napsal pro časopis Connect a vyšel v čísle Connect 05/10, zde jej publikuji s laskavým svolením redakce.*
*Jedná se o druhý díl seriálu o počítačových sítích. Na tomto webu již existuje obsahově shodný (a rozsáhlejší) seriál Počítačové sítě - Computer networks, ale tento článek jsem napsal s určitým časovým odstupem a z trochu jiného pohledu.*

## Ethernet jako základ LAN sítě

V minulém díle časopisu jsme v úvodní části našeho seriálu stručně shrnuli různé technologie, metody a protokoly, které dohromady tvoří něco, co označujeme jako počítačová síť. Dnes se podíváme podrobněji na to, co tvoří základ nejrozšířenějších LAN sítí, tedy technologie Ethernet. Jak funguje komunikace v rámci Ethernetu a jak pracuje nejrozšířenější aktivní prvek switch.

## OSI model

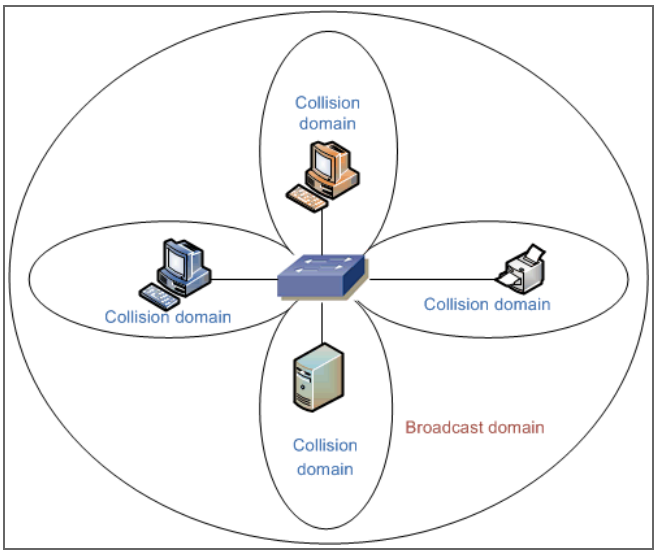
Na začátku se trochu více zmíníme o **Open System Interconnection Reference Model** (OSI model), který jsme minule srovnávali s TCP/IP Reference Model. Tento standardizovaný popis se totiž v praxi velice často používá, když se popisují určité síťové technologie. OSI model je založený na vrstvách, anglicky se používá termín Layer. V praxi často říkáme, že něco pracuje na L2, tím myslíme na 2. vrstvě OSI modelu. Každá vrstva OSI modelu zajišťuje skupinu konceptuálně podobných funkcí a poskytuje služby vrstvě nad ní a přijímá služby od vrstvy pod ní.

| vrstva | jméno | jednotka | funkce vrstvy | příklad |
|---|---|---|---|---|
| L7 | Application | Data | Síťové procesy pro aplikaci, ověření uživatelů, vše závislé na aplikaci. | HTTP, DNS |
| L6 | Presentation | Data | Reprezentace dat a šifrování. Řeší rozdíly v reprezentaci dat mezi aplikací a síťovým formátem - kóduje data pro přenos. | SSL, MPEG |
| L5 | Session | Data | Spojení mezi aplikacemi, správa session. Komunikace jedné aplikace s druhou, posílání více dat po sobě. Udržuje celé spojení mezi dvěma počítači. | NetBIOS |
| L4 | Transport | Segments | End-to-end spojení systémů, spolehlivost - zajišťuje kompletní přenos dat, kvalita služby. Řeší spolehlivé odeslání všech dat ze zdroje do cíle pomocí segmentace a potvrzování. | TCP, UDP |
| L3 | Network | Packets | Logická adresace - routování - určení cesty paketu, přenos dat z bodu do bodu, používá IP adresy, fragmentace. Komunikace mezi zdrojovým a cílovým zařízením pomocí IP adresy. | IP, ICMP, OSPF |
| L2 | Data Link | Frames | Fyzická adresace, MAC - media access control a LLC - logical link control, datový tok, synchronizace rámců, komunikace 1 hop, používá MAC adresy. Detekce chyb, řízení toku a přístupu na médium. Komunikace mezi dvěma zařízeními v jednom subnetu (nebo na bránu) pomocí MAC adresy. Vytváří rámce (hlavička + data + zápatí). | Ethernet, 802.1q, PPP, ARP |
| L1 | Physical | Bits | Fyzické parametry linky - média (kabely, rádio, světlo), signály a binární přenos. Řeší fyzické poslání dat (přenášeným bitům nepřiřazuje žádný význam). | 100BaseT, 802.11g |

## Obecně o Ethernetu

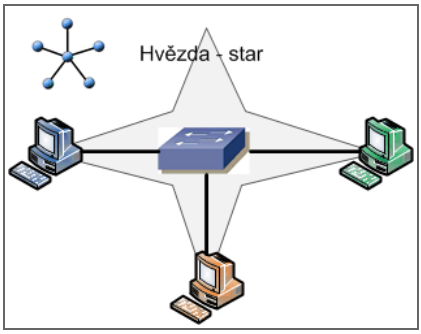
V dnešní době jsou asi všechny lokální sítě založeny na rodině síťových technologií zvané **Ethernet**. Takže když se chceme dozvědět, jak funguje komunikace v počítačové síti, tak se nejprve musíme seznámit s vlastnostmi Ethernetu.

Ethernet je standardizován ve skupině norem **IEEE 802.3**. Definuje parametry přenosového média a signálů pro fyzickou vrstvu (L1) a také obecný adresovací formát a Media Access Control (MAC) pro data link vrstvu (L2). Z pohledu TCP/IP Model se nachází celý Ethernet v první vrstvě, tedy linkové. Tato vazba je důležitá, protože TCP/IP nejčastěji pracuje nad Ethernetem.




**P**oužit jako přenosové médium koaxiální kabel, stejně jako zapojení na sdílenou sběrnici, dnes asi nikdo pro drátovou síť nepoužívá. Takže pouze pro referenci zmíníme metodu přístupu na sdílenou síť ***Carrier Sense Multiple Access With Collision Detection*** (CSMA/CD), která řešila kolize při vysílání. K tomu je důležitý termín **kolizní doména**, což je část sítě, kde může dojít ke kolizi vysílání více stanic. Pokud používáme switch, tak na každém portu je ukončena kolizní doména. Tedy když připojíme do portu pouze jedno zařízení, nedochází ke kolizím. Pokud ovšem používáme hub nebo zapojení na sdílenou sběrnici, tak kolize mohou nastat.

**J**ako médium dnes asi vždy použijeme kroucenou dvojlinku (twisted pair) či optické vlákno (fiber optic). A pro propojení použijeme spoje bod-bod pomocí switche (přepínače), tedy zapojení do hvězdy.




## Varianty Ethernetu

**P**ro Ethernet existuje řada variant, které se liší podle přenosové rychlosti a typu fyzického média. Důležité ale je, že formát rámce (Ethernet frame) je stále stejný. Původní Ethernet, s kterým se dnes těžko setkáme, dosahoval rychlosti 10 Mbps. Jeho nástupce s rychlostí 100 Mbps se označuje jako ***Fast Ethernet*** a pokud používá kroucenou dvojlinku kategorie 5 nebo lepší, tak se označuje jako 100Base-TX, při použití optického vlákna záleží na jeho typu, může se jednat například o 100Base-FX. Další navýšení rychlosti přináší ***Gigabit Ethernet***, na kroucené dvojlince je to 1000Base-T. Dále existuje řada rychlejších variant.

**D**o detailů kabeláže nebudeme zabíhat, pouze krátká zmínka. Nejběžněji se používá **kroucená dvojlinka** (Twisted Pair - TP), která může být buď ***nestíněná*** (Unshielded TP - UTP) nebo s ohranou proti rušení ***stíněná*** (Shielded TP - STP). Dále se TP kabely dělí do určitých kategorií, podle definovaných kritérií. Jako stále hojně používaná kategorie 5e (Cat-5e) či nová kategorie 6a (Cat-6a).

**V** Ethernetu můžeme používat ***Half Duplex***, kdy se vysílá pouze v jednom směru nebo **Full Duplex**, tedy současné vysílání v obou směrech. Využívá se toho, že kroucená dvojlinka obsahuje více vodičů a komunikace v různém směru běží po jiném páru. Podmínkou použití Full Duplex je, že nepoužíváme huby, ale switche. Výhodou Full Duplexu je to, že se nám teoreticky zdvojnásobí propustnost.

## MAC adresa

**E**thernetové stanice komunikují pomocí zasílání **rámců** (frames), což je jednotka komunikace na L2 (tedy to, co se posílá po síti). Jedná se o určitý blok dat, který má doplněnu hlavičku a zápatí (kontrolní součet). ***Hlavička rámce*** obsahuje zdrojovou a cílovou MAC adresu a EtherType, ten určuje jaký protokol je použit na vyšší vrstvě (nejčastěji 0x0800 pro IPv4). Ethernetových rámců existuje více typů, ale nejvíce se setkáme s verzí 2. Pro rozlišení jednotlivých rámců se na začátku vysílá speciální úvod (preamble) složený ze sekvence střídajících se jedniček a nul a SFD - oddělovač začátku rámce.

| úvod | SFD | cílová MAC adresa | zdrojová MAC adresa | typ / délka | data (payload) | CRC |
|---|---|---|---|---|---|---|
| 7B | 1B | 6B | 6B | 2B | 46-1500B | 4B |

**MAC adresa**, jinak označovaná jako ***fyzická adresa*** či ***Ethernet adresa***, je 48 bitové unikátní sériové číslo. Výrobci HW mají přidělený určitý rozsah adres, který používají. MAC adresa bývá zakódována v zařízení výrobcem, ale v dnešní době ji lze často změnit. Tím způsobem se provádí útoky s podvržením MAC adresy ***MAC spoofing***.

**M**AC adresa by měla identifikovat zařízení v síti, ale neříká nic o jeho umístění. Navíc MAC adresy se používají pouze pro komunikaci v rámci stejného subnetu, tedy na L2. Zápis MAC adresy se běžně provádí třemi různými způsoby, buďto pomocí dvojic oddělených dvojtečkou či pomlčkou nebo pomocí čtveřic oddělených tečkou. Příklad 01:23:45:67:89:ab, 01-23-45-67-89-ab či 0123.4567.89ab.

**P**ři běžné unicastové komunikaci by síťový adaptér měl přijímat pouze rámce, které mají jako cílovou adresu jeho

MAC adresu. V případě, že chceme například monitorovat provoz, tak musíme síťovou kartu přepnout do tzv. ***promiskuitního módu*** (promiscuous mode), kdy nevybírá provoz, ale vše odesílá vyšší vrstvě (procesoru) ke zpracování.

## Jak funguje switch

**S**witch, česky řečeno přepínač, nahradil starší zařízení *hub* (rozbočovač) a také *bridge* (most). Hub pracuje na L1 a je velice jednoduchý, veškerý provoz (rámce), který přijde na jednom portu, odešle na všechny ostatní porty (mimo příchozího). V dnešní době se již snad s huby v praxi nesetkáme. Výhoda switche je v tom, že již většinu provozu nerozesílá všude, ale pouze tam, kde se nachází příjemce.

**Switch** již pracuje na L2, takže přistupuje k MAC adresám a podle nich rozesílá provoz. Když přijde nějaký rámec, tak si přečte *zdrojovou MAC adresu* a v paměti si sestavuje tabulku portů a MAC adres, které se za nimi nachází. Tato tabulka se označuje jako ***CAM tabulka*** (Content Addressable Memory). Potom se podívá na *cílovou MAC adresu* a hledá ji v CAM tabulce. Pokud ji najde, tak odešle rámec na daný port. Pokud záznam pro tuto adresu nemá, tak rámec odešle na všechny porty mimo příchozího (většinou pak brzy přichází odpověď, takže takto odeslaných rámců není mnoho).

**V**ýše uvedený popis se týkal ***unicastu***. Pokud jde o ***broadcast***, který se na L2 pozná podle MAC adresy se samými binárními jedničkami, tedy `FF:FF:FF:FF:FF:FF`, tak se vždy rámec odesílá na všechny porty mimo příchozího. Pokud se jedná o ***multicast***, MAC adresa začíná `01:00:5e` a jedná se o MAC adresu skupiny, tak se takový rámec rozesílá stejně jako broadcast. To proto, že z této adresy není nikdy žádný příchozí provoz, takže neexistuje záznam v CAM tabulce. Optimálnější rozesílání multicastu na L2 se dá řešit pomocí ***IGMP Snoopingu***.

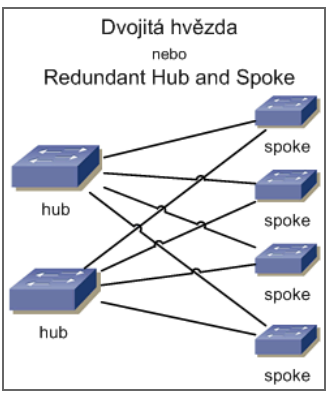
**P**oužití switchů je o něco bezpečnější než připojení na sdílené médium, protože provoz by se neměl šířit všude po subnetu. Ale bezpečnost je relativní, protože existují jednoduché útoky jako zaplavení switche MAC adresami ***MAC flooding***, které zajistí, že switch se začne chovat jako hub. Ale i proti tomu existuje obrana.

**Z** důvodů rychlosti existuje několik metod, jakými switch přeposílá rámce. Nejběžnější je ***cut-through***, kdy switch přeposílá rámec okamžitě, jakmile zná cílovou MAC adresu, což je rychlé, ale nekontrolují se chyby. A ***store-and-forward***, nejprve se přijme celý rámec a uloží do bufferu, ověří se kontrolní součet a teprve potom přepošle (případně zahodí).

## Propojení switchů

**J**edna věc v principu komunikace na L2 je důležitá a to, že switch je naprosto transparentní. To znamená, že nijak nemění procházející data (rámce) a při komunikaci jej nijak neadresujeme. Pouze může rámce kontrolovat a ty poškozené zahazovat.

**P**ři zapojení můžeme použít více switchů a ty navzájem propojovat. Přes propojovací spoj, budeme mu říkat ***uplink***, pak proudí veškerá komunikace, která se má dostat na zařízení na druhém switchi. Princip zůstává stále stejný, pouze pro uplink port vznikne v CAM tabulce větší množství záznamů. Není ovšem optimální zapojovat switche do řady, ale spíše do stromové struktury ideálně s jednou nebo dvěma úrovněmi.



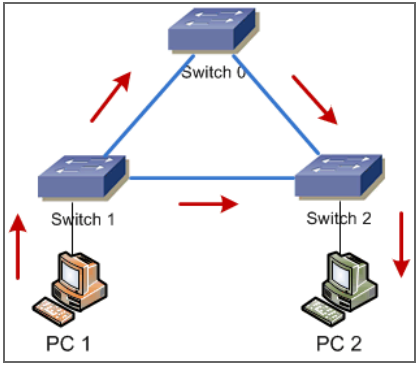

**F**yzická struktura sítě, která vznikne propojením více switchů (mohou jich být desítky), tvoří oblast, kde je možná přímá komunikace mezi všemi připojenými zařízeními. Z logického pohledu tuto strukturu můžeme označovat jako LAN, ale ta se může skládat i zvíce takovýchto oblastí navzájem propojených (přes router). Lepší označení by bylo **subnet**, protože právě v rámci subnetu můžeme přímo komunikovat na L2. Ale subnet vytváříme pomocí adresace na L3. V rámci skupiny propojených switchů můžeme klidně definovat více sunbnetů (v praxi se to takto nedělá!). Potom zařízení z různých subnetů spolu nemohou komunikovat po L3, ale pořád mohou komunikovat po L2, což většinou, když dělíme síť, nechceme.

**P**okud chceme mít fyzicky oddělené subnety, tak máme dvě možnosti. Buďto vybudovat samostatné skupiny switchů, které spolu nebudou přímo propojeny (pouze mohou být spojeny přes router). Nebo využít technologii zvanou ***Virtual LAN*** (VLAN), kterou musí podporovat switche. VLAN funguje na L2 a na fyzické struktuře switchů vytváří logické rozdělení do skupin (vytváří jakoby virtuální switche). Technologii VLAN si podrobně popíšeme v některém z dalších dílů.

## Agregace a redundance

**K**dyž propojujeme switche mezi sebou, tak přes uplink port proudí větší množství dat, proto má většinou tento port větší rychlost než ostatní porty, aby zde nedocházelo ke zpomalování celé komunikace. Můžeme také použít technologii ***Link Aggregation***, kdy propojíme více portů a svážeme je dohromady do jednoho logického portu, který může mít větší rychlost.

**D**ruhá věc, kterou musíme řešit, je možnost výpadku jednoho zařízení, které může způsobit pád celé sítě. Proto je dobré použít zdvojené linky, které směřují k různým switchům. Tím se nám ovšem mění původní zapojení do hvězdy (tedy stromové struktury) a vznikají ***smyčky***. Když máme velké množství switchů, tak může vzniknout smyčka i omylem, při neplánovaném propojení dvou switchů.




Smyčky v síti jsou velký problém, protože k cíli již neexistuje pouze jedna cesta, ale několik různých cest. To vede k tomu, že při odesílání broadcastu (většinou stejně u multicastu) vzniká ***broadcastová bouře***, což je takové zahlcení sítě broadcasty, že se přestane odesílat regulérní provoz. Další problém je několikanásobné doručení a nestabilita CAM tabulky, na switch přichází rámce se stejnou zdrojovou adresou z různých portů.

Smyčky můžeme řešit pomocí výše zmíněné agregace linek, která se ovšem musí konfigurovat a nezabrání tomu, že propojíme switche omylem. Nebo automaticky pomocí protokolu ***Spanning Tree Protocol*** (STP), který obsahují všechny dnešní spravovatelné switche a je defaultně zapnutý (dnes se používají novější verze tohoto protokolu Rapid STP a Multiple STP). Tento protokol zajišťuje blokování redundantních cest a v případě výpadku aktivní cesty zprovozní tu záložní.



Autor: **Petr Bouška**

**Související články:**

## Základy počítačových sítí

Tento seriál jsem napsal pro časopis Connect. Většinu informací obsahuje shodně jako můj starší seriál Počítačové sítě - Computer networks, ale je napsán trochu jiným způsobem. Nejprve jsou stručně shrnuty technologie počítačových sítí a pak se trochu podrobněji probírají od nejnižších vrstev nahoru.

- **Rozumíme počítačovým sítím** [09.05.2010 21:27]
- **Víte, jak pracuje switch?** [25.05.2010 10:39] ..... **právě čtete**
- **Víte, jak pracuje router?** [23.06.2010 18:30]
- **Adresování v IP sítích** [21.07.2010 12:39]

# Víte, jak pracuje router?

Středa, 23.06.2010 18:30 | **Samuraj - Petr Bouška** | ***sítě***

Abychom mohli komunikovat v počítačové síti, musíme používat nějaké logické adresování, které nám určí v jaké části sítě se účastníci nacházejí. A dále metody, které nám zařídí správné směrování dat od odesílatele k příjemci. V dnešních LAN sítích, stejně jako v internetu, se jedná o Internet Protocol (IP). Pro adresování používáme IP adresy a dělení na subnety, pro přeposílání dat používáme routování.



*Tento článek jsem napsal pro časopis **Connect** a vyšel v čísle **Connect 06/10**, zde jej publikuji s laskavým svolením redakce.*
*Jedná se o třetí díl seriálu o počítačových sítích. Na tomto webu již existuje obsahově shodný (a rozsáhlejší) seriál **Počítačové sítě - Computer networks**, ale tento článek jsem napsal s určitým časovým odstupem a z trochu jiného pohledu.*

## Komunikace v síti pomocí IP protokolu

V minulých číslech časopisu Connect jsme se si nejprve stručně shrnuli vše, co taková počítačová síť zahrnuje. Potom jsme začali trochu podrobnější popis od spodních vrstev sítě a napsali si něco o pasivních prvcích, z kterých sítě stavíme. V dalším čísle jsme se věnovali základní technologii, kterou v nejčastějších případech pro komunikaci použijeme, tedy Ethernetu. Dnes se podíváme o vrstvu výše, tedy z pohledu OSI modelu na 3. vrstvu (L3), a budeme se věnovat protokolu IP.

**Internet Protocol** (běžně označovaný jako IP protokol) je součástí rodiny protokolů ***Internet Protocol Suite*** (TCP/IP) a slouží ke komunikaci v počítačové síti. IP protokol může pracovat nad různými sítěmi (myšleno nad různými technologiemi, to je díky abstrakci pomocí enkapsulace), ale v našem seriálu budeme uvažovat speciálně provoz nad Ethernetem, o kterém jsme si povídali minule.

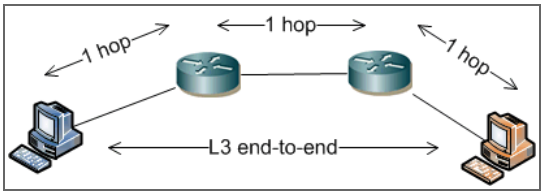
## Vlastnosti IP protokolu

IP protokol definuje adresovací metody a struktury. V současnosti se majoritně využívá Internet Protocol Version 4 (IPv4), ale řadu let se již nasazuje novější verze Internet Protocol Version 6 (IPv6). Data z vyšších vrstev se zapouzdřují (encapsulate) do datové struktury zvané paket (případně datagram). IP je ***nespojovaný*** (connectionless) a ***bezestavový*** (stateless) protokol, což znamená, že před odesláním prvního (a ani žádného dalšího) paketu se nesestavuje spojení (cesta), kterou by pakety putovaly. Pakety putují v síti nezávisle a každý si nese kompletní informaci o odesílateli a příjemci.

Další definující vlastností IP protokolu je, že se jedná o ***nespolehlivou*** komunikaci. Protokol nemá žádné prostředky pro zajištění spolehlivosti. Jednotlivé body na cestě se snaží paket odeslat směrem k cíli, ale není nijak zaručeno, že k příjemci opravdu dorazí nebo že nedorazí vícekrát či v jiném pořadí než byly odeslány. Pro zajištění spolehlivosti se používají protokoly vyšší vrstvy, hlavním příkladem je TCP. To, že se jedná o nespolehlivou komunikaci, má ovšem i pozitivní vlastnosti, což je malá režie a větší rychlost.

Při přenosu dat po síti musíme zajistit i určité vysílací charakteristiky, podle toho, co je síťová cesta schopna přenést. Jedním z parametrů je **MTU** (Maximum Transmission Unit), což je maximální velikost (v bajtech) PDU (v našem případě paketu), který je vrstva schopna přenést. Pro Ethernet je standardní hodnota MTU rovna 1500 B (mluvíme o velikosti paketu, to znamená, že standardní rámec má v tom případě velikost 1518 B). Pokud IP protokol obdrží data (z vyšší vrstvy) větší než je MTU, tak obsahuje mechanizmus zvaný ***fragmentace***, který je rozdělí do více paketů a odesílá fragmentovaně. Tato metoda má ovšem řadu nevýhod, takže například TCP protokol obsahuje lepší metodu zvanou segmentace a předává data v menších blocích.

Klasická počítačová síť (založená na IP protokolu) je stavěna tak, že inteligence se soustředí v koncových bodech (stanicích, serverech, tiskárnách) a mezi nimi se (z pohledu L3) nachází routery. Délku cesty od odesílatele k příjemci měříme počtem hopů, tedy počtem zařízení (routerů) přes které musí L3 komunikace projít. Komunikace (na L3) probíhá end-to-end principem (paket obsahuje IP adresy odesílatele a příjemce), kdy routery pouze přeposílají data směrem k cíli (ale na L2 se komunikuje bod-bod, rámec má vždy MAC adresy pro aktuální hop). Veškerou vyšší logiku řeší koncové body.




## TCP/IP model

Minule jsem si stručně popsali OSI model, dnes se lehce podíváme na **Internet model** (často označovaný jako TCP/IP model). Ten má oproti sedmi vrstvám OSI modelu vrstvy pouze čtyři. V praxi se používá různé pojmenování vrstev (a někdy se hovoří i o jiném počtu), v tabulce jsou uvedeny názvy, které používá RFC standard. Obecně je Internet model dost podobný OSI modelu. Můžeme říci, že vrstvy fyzická a data link OSI modelu jsou spojené do jedné linkové vrstvy TCP/IP modelu. Síťová vrstva odpovídá vrstvě internet, stejně jako transportní je shodná. Vyšší tři vrstvy OSI modelu jsou v TCP/IP modelu obsaženy v jedné vrstvě aplikační.

| vrstva | jméno | příklad |
|---|---|---|
| L4 | Application | SSL, HTTP, DNS |
| L3 | Transport | TCP, UDP |
| L2 | Internet | IP, ICMP, OSPF |

| L1 | Link | Ethernet, ARP |
|---|---|---|

**I** když jsme si nyní popsali Internet model a dříve jsme zmiňovali, že je pro praxi reálnější, tak dále v textu budeme odkazovat vrstvy podle OSI modelu, protože to je v literatuře běžnější způsob.

## IP paket

**P**DU (Protocol Data Unit) na L3 označujeme jako **paket** (packet). Můžeme také používat termín **datagram**, což je paket nespolehlivé služby. Když tedy použijeme na vyšší vrstvě protokol UDP, tak můžeme obecně používat termín datagram (i když se stále jedná o paket). Pokud je na vyšší vrstvě TCP, tak bychom měli používat pouze termín paket.

**P**aket je formátovaná jednotka dat, která obsahuje kontrolní informace (hlavičku) a uživatelská data (payload, zapouzdřená data vyšších vrstev, až k aplikačním datům). Před odesláním po síti se předá nižší vrstvě, u nás Ethernetu, ten provede opět zapouzdření a vytvoří rámec (frame), který se následně po bitech vysílá na přenosové médium.

**I**Pv4 **hlavička** obsahuje vždy zdrojovou a cílovou IP adresu, označení verze IP (IPv4 nebo IPv6), délku hlavičky, Type of Service (prioritu), velikost paketu, identifikaci (pro fragmentaci), zda jsou data fragmentovaná, offset fragmentu v originálním paketu, TTL (time to live), což je počet hopů přes které paket projde než se zahodí (aby nám v případě smyčky nekoloval věčně), určení protokolu vyšší vrstvy a kontrolní součet pro hlavičku. Takováto standardní hlavička má velikost 20B, ale může ještě obsahovat další volitelné vlastnosti.

| bits | 0-3 | 4-7 | 8-15 | 16-18 | 19-31 |
|---|---|---|---|---|---|
| 0 | 4 | header length | Type of Service | total length (header + data) | |
| 32 | identification | | | flags | fragment offset |
| 64 | TTL | | protocol | header checksum | |
| 96 | source IP | | | | |
| 128 | destination IP | | | | |
| 160 | options (if any) | | | | |
| 160/192+ | DATA | | | | |

**T**ak jako v Ethernetu existuje fyzická adresa zařízení, která se označuje jako MAC adresa, tak v IP protokolu máme logickou adresu zařízení, které říkáme **IP adresa**. Tato adresa jednoznačně identifikuje síťové rozhraní v počítačové síti a musí být unikátní. V IPv4 se používají 32 bitové adresy, které se zapisují dekadicky pomocí čtyř oktetů (8 bitů z adresy, tedy maximálně hodnota 255), příkladem je adresa `10.5.127.2`. Adresování v IP sítích je rozsáhlejší a komplexní oblast, které se budeme věnovat v příštím díle.

## IPv4 vs. IPv6

**V** této sérii článků mluvíme o základech počítačových sítí, takže se věnujeme IPv4 (což je první, v praxi používaná, verze protokolu), která se stále používá majoritně. Ale pro úplnost si uvedeme pár informací o IPv6. Hlavní důvod vzniku nové verze IP protokolu je, že v IPv4 existuje cca 4 miliardy adres. Což se může zdát jako velké číslo, ale tento počet adres by již byl vyčerpán, pokud by se nezačali používat různé metody jako NAT, CIDR a lepší způsoby přidělování adres. Přesto hrozí vyčerpání adres IPv4 v krátké době (odhaduje se něco přes rok).

**I**Pv6 používá adresy dlouhé 128 bitů (což dává ohromné množství adres), které se zapisují jako 8 skupin po čtyřech hexadecimálních číslicích oddělených dvojtečkou. Sice existuje řada pravidel, jak se dá adresa zapsat kratším způsobem, ale přesto se jedná o natolik dlouhé komplikované číslo, že pro použití lidmi je potřeba DNS. V IPv6 je integrální součástí bezpečnost pomocí IPsec, zaměřuje se na podporu mobility, obsahuje metodu bezestavové autokonfigurace IP adresy. V IPv6 neexistuje vysílání broadcast, ale pouze unicast, multicast a novinka anycast (skupina příjemců, ale data se doručí pouze jednomu nejbližšímu).

## Směrování v IP síti

**K**omunikaci v IP síti na L3 označujeme jako **směrování** (routing, dále budeme používat počeštěný a v praxi častý termín routování) a provádí jej každé zařízení, které se této komunikace účastní. Koncové body většinou používají pouze jednodušší formu routování, ale i ony musí rozhodnout, jak data odeslat. Hlavní jsou prvky na cestě, které rozhodují jak dále směrovat paket. Těmito zařízeními jsou převážně **routery**. V lokálních sítích se často pro routování využívá L3 switch (označovaný také jako MultiLayer Switch), který pro přepínání paketů využívá speciální HW (takže dosahuje velkého výkonu). Funkce routeru má také firewall a pro routování můžeme použít i běžný počítač osazený více síťovými adaptéry.

**V**lastní routování znamená rozhodnutí kam a kudy odeslat paket, když známe cílovou adresu podle údajů z jeho hlavičky. Kudy znamená přes jaký lokální interface a kam na jakou další adresu (hop) ve stejném subnetu. Pro rozhodování se využívá ***routovací tabulka***, ta obsahuje seznam cest (route) a souvisejících informací. Routovací tabulka se sestavuje různým způsobem. Automaticky se generují záznamy z přímo připojených rozhraní. Například v počítači máme dvě síťové karty, které mají nastavenu IP adresu a masku do dvou sítí. Z těchto údajů se vytvoří dva záznamy, že daná síť se nachází za tímto síťovým rozhraním. Pro jednoduché sítě se používá ***statické routování***, kdy se záznamy do routovací tabulky zadávají ručně. Dynamicky nebo také ručně se vytváří záznamy pro ***defaultní routování*** (související termín gateway), tento údaj se použije ve chvíli, kdy neznáme žádnou cestu do cílové sítě.

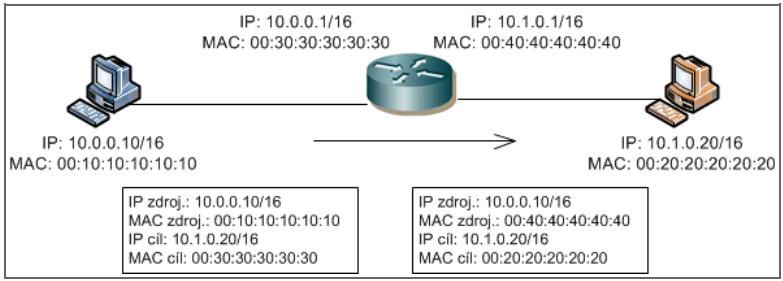
**V**e složitějších sítích a když chceme, aby se automaticky reagovalo na změny topologie sítě, se používají ***routovací protokoly***. Ty zjišťují nejlepší cesty v síti, upravují routovací tabulku a informují okolní routery. V tomto případě mluvíme o ***dynamickém routování***. Příkladem routovacích protokolů je RIP (Routing Information Protocol), EIGRP (Enhanced Interior Gateway Routing Protocol), OSPF (Open Shortest Path First) či internetový BGP (Border Gateway Protocol).

**A**bychom měli terminologii kompletní, tak ještě zmíníme termín ***routovaný protokol***, tedy protokol, který se používá ke komunikaci v routované síti, u nás jde o IP protokol.

## Jak funguje router

**R**outer je zařízení, které pracuje na L3 a pro svou činnost využívá první 3 vrstvy OSI modelu. Slouží k propojení subnetů a oddělení broadcastových domén. Hraniční router se často označuje jako brána (gateway), slouží k propojení LAN sítě s WAN (internetem). Na něj se směruje veškerá neznámá komunikace. Pokud nám brána slouží pro připojení k internetu, tak by neměla propouštět privátní adresy (cílová adresa musí být veřejná).

**R**outer není transparentní zařízení a u každého příchozího rámce odstraní L2 hlavičku a při odesílání vytvoří novou (z údajů aktuálního hopu) včetně kontrolního součtu rámce (vlastní paket zůstává nezměněn). To jsou operace (spolu s vyhledáváním v routovací tabulce a přeposíláním paketů), které stojí určitou režii a v klasických routerech ji provádí procesor.




Když router přijme paket (ve skutečnosti přijme rámec, ale L2 údaje odstraní), tak se podívá na cílovou IP adresu v hlavičce. Hledá v paměti v routovací tabulce, zda má nějakou cestu pro cílovou adresu. Pokud cestu nezná, paket zahodí. Pokud záznam našel (je jedno jakým způsobem v tabulce vzniknul), tak zjistí přes jaký interface data odeslat a případně na jakou adresu. Při odesílání ví, jestli cílová adresa patří do některého z přímo připojených subnetů nebo odesílá na další router (hop) na cestě. Než odešle data, tak připraví rámec, kde uvede svoji MAC adresu jako zdrojovou a jako cílovou buď přímo MAC adresu příjemce nebo dalšího routeru. Pro nalezení MAC adresy používá ARP protokol. Pokud adresu nezjistí, tak paket zahodí.

## ARP protokol

Při komunikaci v sítích, o kterých si zde povídáme, používáme aktivně IP adresy (případně DNS jména, která se na IP adresu převedou). Přesto již z minulého dílu víme, že pro doručení rámce v Ethernetu musí naše stanice znát i zdrojovou a cílovou (pro aktuální hop) MAC adresu. Svoje údaje samozřejmě stanice zná, ale pro cíl zná pouze IP adresu. Tehdy vstupuje na scénu již letitý protokol **Address Resolution Protocol** (ARP) definovaný v RFC 826. To platí pokud používáme IPv4, o které zde mluvíme primárně. IPv6 využívá Neighbor Discovery Protocol (NDP).

Princip ARP protokolu je jednoduchý. Jen na začátku musíme zmínit, že funguje pouze v rámci stejného subnetu. To je ovšem v pořádku, protože MAC adresy potřebujeme pouze pro zařízení ve stejném subnetu. Stanice, která potřebuje zjistit nějakou MAC adresu sestaví ARP žádost (request), která obsahuje hledanou IP adresu, a odešle ji jako broadcast. Všechny stanice v subnetu tento rámec obdrží a ten, kdo má danou IP se sestaví ARP odpověď (response), kde uvede všechny potřebné údaje, a unicastem ji odešle tazateli. Aby se zmenšil počet broadcastů, tak ještě zařízení používají ARP cache, tedy dočasnou paměť, kde si po určitou dobu uchovávají zjištěné hodnoty.



Autor: Petr Bouška

**Související články:**

## Základy počítačových sítí

Tento seriál jsem napsal pro časopis Connect. Většinu informací obsahuje shodně jako můj starší seriál Počítačové sítě - Computer networks, ale je napsán trochu jiným způsobem. Nejprve jsou stručně shrnuty technologie počítačových sítí a pak se trochu podrobněji probírají od nejnižších vrstev nahoru.

- Rozumíme počítačovým sítím [09.05.2010 21:27]
- Víte, jak pracuje switch? [25.05.2010 10:39]
- Víte, jak pracuje router? [23.06.2010 18:30] ..... **právě čtete**
- Adresování v IP sítích [21.07.2010 12:39]

# Adresování v IP sítích

Středa, 21.07.2010 12:39 | Samuraj - Petr Bouška | sítě

Počítačové sítě jsou dnes nejčastěji založeny na rodině protokolů TCP/IP. Z pohledu fyzického i logického nepoužíváme jednu globální síť, ale hierarchické dělení na různé části zvané subnety (podsítě). Aby spolu jednotlivé subnety mohli komunikovat, tak musíme zajistit správné fyzické propojení, ale také přidělit správné adresní prostory. V článku se spolu podíváme na adresy IPv4 (Internet Protocol Version 4), jak se rozdělují a jak se s nimi počítá.



*Tento článek jsem napsal pro časopis Connect a vyšel v posledním čísle Connect 07-08/10, zde jej publikuji s laskavým svolením redakce.*
*Jedná se o čtvrtý díl seriálu o počítačových sítích. Na tomto webu již existuje obsahově shodný (a rozsáhlejší) seriál Počítačové sítě - Computer networks, ale tento článek jsem napsal s určitým časovým odstupem a z trochu jiného pohledu.*
*Bohužel další díly již následovat nebudou, protože časopis Connect byl zrušen. Je to škoda, protože se podle mne jednalo o jediný český časopis na docela technické úrovni (oproti hrůzám jako je Chip a další). Ale vydavatelství CPress Media koupila Mladá Fronta a rozhodla se řadu časopisů, mezi nimi Connect, zrušit.*

## Adresy a jejich výpočty v počítačových sítích založených na TCP/IP

Celou řadu termínů z oblasti počítačových sítí jsme probrali v minulých dílech. Nyní si některé termíny zopakujeme a přidáme několik nových, které jsou důležité pro výpočty adres v IP sítích. Zásadním termínem je IP adresa a maska podsítě. V celém článku budeme hovořit o čtvrté verzi IP protokolu, tedy IPv4.

## IP adresa a subnet

IP adresa je logická adresa zařízení v počítačové síti (na 3. vrstvě podle OSI modelu), která používá IP protokol. V IP v4 je tato adresa velká 32bitů (4 byte) a zapisuje se pomocí ***dot-decimal notation***. To znamená pomocí čtyř dekadických hodnot (každá velká jeden byte), označovaných jako oktet, oddělených tečkou. Příkladem IP adresy je `193.222.5.15`. IP protokol je zde již delší dobu a stále probíhá jeho vývoj, my se budeme bavit o modifikacích, které jsou běžné v dnešních sítích. Veškerý popis se nachází v RFC dokumentech.

IP adresa může nabývat hodnot `0.0.0.0` až `255.255.255.255` (teoreticky, ne všechny adresy je možno v praxi použít), což je adresní rozsah IP sítí a internetu. Sice se zápis adresy provádí standardně v desítkové soustavě, ale vše je optimalizováno pro zpracování počítačem, který používá binární soustavu. Takže pro výpočty adres jsou třeba základy počítání v dvojkové soustavě.

V minulých dílech jsme si vysvětlili základní způsoby komunikace, takže víme, že tento rozsah musíme rozdělit na menší části. Těmto částem říkáme **subnety** (česky podsítě) někdy i sítě. Síťová zařízení uvnitř subnetu sdílí společný prefix IP adresy.

## Maska sítě

IP adresa se skládá z několika částí. V základu jde o dvě části, ***prefix identifikující síť*** a ***adresa uzlu*** v rámci podsítě. U původních classful sítí byl prefix pevně dán třídou, do které adresa patřila. U dnešních classless sítí můžeme prefix určit z masky podsítě. Nebo adresu rozdělit na tři části, adresu sítě, adresu podsítě a adresu uzlu. Kde adresa sítě je dána třídou IP adresy a adresa podsítě je rozdíl masky a třídy.

```
classful    <network-prefix><-------host-number-------->
classless   <network-number><subnet-number><host-number>
classless   <-------network-prefix-------><host-number>
```

Pro to, abychom určily, která část IP adresy je pro podsíť a která pro uzel, se používá **maska podsítě** (subnet mask). Ta v binárním tvaru obsahuje jedničky následované nulami a pomocí jedniček „vymaskovává" část síťového prefixu v IP adrese. Jinak řečeno, tam kde jsou v masce jedničky je v IP adrese část podsítě a kde jsou nuly je část uzlu. Síťová část IP adresy určuje podsíť a používá se ke směrování. Část uzlu identifikuje stanici uvnitř daného subnetu.

Maska se v IPv4 zapisuje stejně jako IP adresa pomocí dot-decimal form s tím, že jsou validní pouze adresy, které v binárním zápisu mají zleva jedničky následované nulami (za první nulou smí být pouze nuly). Příkladem síťové masky je `255.255.255.0`. Maska `255.255.255.254` není platná, protože uvnitř takového subnetu by se nacházelo 0 uzlů. Maska `255.255.255.255` není maskou podsítě, ale určuje jeden uzel.

Když se bavíme o masce sítě, tak ještě musíme zmínit občas používaný speciální zápis, který se označuje jako ***inverse mask*** nebo ***wildcard mask***. Jedná se o obrácenou masku, jednoduše řečeno tam, kde jsou v tradiční masce jedničky jsou nuly a naopak. Pro výpočet v dekadické formě můžeme použít pro každý oktet `hodnota = 255 - hodnota oktetu`. Například pro masku `255.255.255.240` je inversní maska `0.0.0.15`.

## Veřejné a privátní adresy

Prvotní princip IP sítí byl takový, že IP adresa musela být unikátní v rámci celé sítě a jednotlivé uzly (stanice, servery a další zařízení s přiřazenou IP adresou) spolu mohli komunikovat. S rozvojem internetu se ukázalo, že počet adres, které je možno v IPv4 vytvořit, rozhodně není dostatečný. Začaly se tedy používat různé metody, jak se s nedostatkem adres vypořádat. Nejrazantnější je nová verze IP protokolu IPv6, která obsahuje mnohem více adres, ale její nasazování není jednoduché. Dále se objevila technika CIDR a také se IP adresy rozdělili na dva typy, na veřejné a neveřejné IP adresy.

***Veřejné IP adresy*** (public address) tvoří hlavní část adresního rozsahu internetu a tyto adresy jsou routovatelné v rámci celého internetu. Jednoduše řečeno počítač s veřejnou adresou může být dostupný z celého internetu. Tyto adresy musí být unikátní v celé síti (internetu).

Oproti tomu ***privátní IP adresy*** (private adress) by se měli používat pouze v rámci LAN sítí a v internetu by přes ISP neměli komunikovat. Firma pak používá jednu (nebo více) veřejnou adresu a pomocí techniky NAT (Network

Address Translation) se při komunikaci mimo LAN překládají privátní adresy na tuto veřejnou. Stejné privátní adresy se tak mohou nacházet na mnoha místech v internetu, ale nemohou spolu přímo komunikovat. Následující tabulka ukazuje jednotlivé privátní rozsahy.

| síť | adresa sítě | broadcast adresa | adresy uzlů |
|---|---|---|---|
| 10.0.0.0/8 | 10.0.0.0 | 10.255.255.255 | 10.0.0.1 - 10.255.255.254 |
| 192.168.0.0/16 | 192.168.0.0 | 192.168.255.255 | 192.168.0.1 - 192.168.255.254 |
| 172.16.0.0/12 | 172.16.0.0 | 172.31.255.255 | 172.16.0.1 - 172.31.255.254 |

Některé adresy z veřejných i privátních mají od začátku plánovaný speciální význam. Z těch důležitějších se jedná o rozsah `127.0.0.0/8`, což jsou localhost loopback adresy či `169.254.0.0/16`, to jsou adresy pro autokonfiguraci (APIPA).

## Síťové třídy

Metoda, s kterou se již v dnešní praxi nesetkáme, ale jejíž označení se stále používá, je rozdělování sítí do tříd. V minulosti se adresní prostor rozdělil do pěti tříd (hlavní jsou třída A, B a C), v každé třídě existovalo určité množství podsítí. Třída byla určena prvními čtyřmi bity adresy a ke každé třídě byla napevno přiřazena maska podsítě. Masky navíc mohly mít v daném oktetu pouze jedničky nebo nuly. Protože byly masky pevně dány a daly se podle adresy rozeznat, tak se ani neuváděly (což byl později problém u routování). Pokud používáme síťové třídy, tak mluvíme o ***classful network***. Následující tabulka zobrazuje základní parametry jednotlivých tříd.

| třída | určující bity | rozsah adres | maska | CIDR maska | poznámka |
|---|---|---|---|---|---|
| class A | 0xxx | 0 - 127.x.x.x | 255.0.0.0 | /8 | hlavní |
| class B | 10xx | 128 - 191.x.x.x | 255.255.0.0 | /16 | hlavní |
| class C | 110x | 192 - 223.x.x.x | 255.255.255.0 | /24 | hlavní |
| class D | 1110 | 224 - 239.x.x.x | | | multicast |
| class E | 1111 | 240 - 255.x.x.x | | | rezervováno |

Později se ukázalo, že toto pevné rozdělení sítí je neefektivní a velice plýtvá adresami, které začaly ubývat. Tak vznikl síťový design označovaný jako ***classless***. Ten využívá ***Variable Length Subnet Mask*** (VLSM), které povoluje proměnnou délku síťové masky. Na VLSM je založen ***Classless Inter-Domain Routing*** (CIDR), který dovoluje libovolně dlouhou masku a používá zkrácený zápis pomocí počtu jedničkových bitů v masce (délka masky). CIDR zápis adresy vypadá například `10.5.0.2/28`, tomu odpovídá tradiční maska `255.255.255.240`, kterou zapíšeme binárně `11111111.11111111.11111111.11110000`.

CIDR také obsahuje mechanismus agregace, který dovoluje spojit několik spojitých síťových rozsahů do jednoho supernetu. Použití agregace šetří místo a prostředky při routování. Použitím agregace můžeme spojit subnety `192.168.0.0/24` a `192.168.1.0/24` do `192.168.0.0/23`.

## Adresy v subnetu

Každá podsíť obsahuje tři typy adres. První adresa z rozsahu se nesmí přiřadit žádnému uzlu. Tato adresa slouží k identifikaci subnetu o označuje se jako ***network ID*** nebo adresa sítě či základní adresa. Tato adresa obsahuje pouze nuly v části adresy uzlu. Dále následují standardní ***adresy uzlů***, které můžeme přiřazovat síťovým zařízením. Poslední adresa subnetu se opět nemůže použít pro uzel, jedná se o tzv. ***broadcast address***. Tato adresa se využívá pro subnet directed broadcast, tedy zprávu, kterou chceme zaslat všem stanicím v subnetu. Tato adresa má pouze jedničky v části adres uzlu.

V následující tabulce jsou zobrazeny všechny adresy pro síť `192.168.5.12/30`. Masku `/30` můžeme zapsat binárně `11111111.11111111.11111111.11111100`, takže IP adresa se může měnit pouze v posledních dvou bitech.

| IP adresa | binárně | typ |
|---|---|---|
| 192.168.5.12 | 11000000.10101000.00000101.00001100 | network ID |
| 192.168.5.13 | 11000000.10101000.00000101.00001101 | uzel |
| 192.168.5.14 | 11000000.10101000.00000101.00001110 | uzel |
| 192.168.5.15 | 11000000.10101000.00000101.00001111 | broadcast |

## Výpočet maximálního počtu uzlů a podsítí

Maximální počet uzlů a subnetů v určité síti je určen pomocí masky podsítě. Určení, která část IP adresy je částí sítě a která částí hostů, je jednoduché. Když si IP adresu i masku zapíšeme binárně pod sebe, tak tam, kde jsou v masce jedničky, je část síťová a kde nuly část adresy uzlu v rámci subnetu. Jednoduché je to u nejběžnějších masek, kdy v oktetu jsou buď jen jedničky, tedy hodnota 255, nebo nuly.

| | | |
|---|---|---|
| IP adresa | 10.240.5.8 | **00001010.11110000.00000101**.00001000 |
| maska | 255.255.255.0 | **11111111.11111111.11111111**.00000000 |

Když máme masku v CIDR zápisu, tak rovnou víme počet jedniček a počet nul můžeme dopočítat `32 - CIDR hodnota`. Síťová část je dána jedničkami v masce sítě, pokud v této části IP adresy změníme nějakou hodnotu, dostaneme jiný subnet. Takže počet kombinací v této části adresy se rovná maximálnímu **počtu subnetů**. Počet kombinací spočítáme jako $2^{\text{počet jedniček}}$.

Ve starších dobách se nesměli používat subnety, které měli v adrese samé jedničky nebo nuly (to určovalo staré RFC 950), takže podle této normy by počet subnetů byl $2^{\text{počet jedniček}} - 2$. Již dlouho ale platí novější RFC 1812, kde můžeme využít všechny adresy. Celkový počet subnetů také záleží na tom, jestli přihlížíme k síťovým třídám, potom počítám počet subnetů v dané síťové třídě, tedy $2^{\text{počet jedniček v masce - počet jedniček ve třídě}}$.

Maximální počet **adres pro uzly** spočítáme obdobně pomocí vzorce $2^{\text{počet nul}} - 2$. Dvojku odečítáme, protože jako adresu uzlu nemůžeme použít network id a broadcast adresu.

Uvedeme si nyní jednoduchý konkrétní příklad, na kterém snad bude dřívější obecný popis jasně pochopitelný. Máme síťovou adresu zapsanou pomocí CIDR `148.25.3.5/22`. Takže počet jedniček v masce je `22` a počet nul je `32 - 22 = 10`. Pokud budeme uvažovat třídy, tak tato adresa spadá do třídy B (podle prvních dvou bitů IP adresy, první oktet je binárně 10010100) a její maska je /16. Počet subnetů v rámci třídy B s maskou `/22` je $2^{22-16} = 2^6 = 64$. Pokud bychom brali veškeré subnety, tak je to $2^{22} = 4\ 194\ 304$. Počet hostů, který může být v každém subnetu s touto maskou sítě, je $2^{10} - 2 = 1024 - 2 = 1022$.

## Jsou dvě adresy ze stejného subnetu?

Občas potřebujeme určit, zda se dvě adresy (které známe včetně masky) nachází ve stejném subnetu (podle toho pak probíhá komunikace apod.). Pokud máme jednoduchou masku, tak to poznáme na první pohled, ale u některých složitějších musíme použít základní matematiku. Stačí porovnat síťové části adres a pokud jsou stejné, patří adresy do stejné podsítě. Nejjednodušší je převést všechny adresy do binární formy a vše je hned patrné. Důležitým předpokladem je, aby masky podsítí byly shodné.

Příklad si ukážeme na adresách `192.168.5.13/22` a `192.168.7.128/22`, hned vidíme, že masky jsou stejné.

| dekadicky | binárně |
|---|---|
| 192.168.5.13 | **11000000.10101000.000001**01.00001101 |
| 192.168.7.128 | **11000000.10101000.000001**11.10000000 |
| 255.255.252.0 | **11111111.11111111.111111**00.00000000 |

Ze zápisu vidíme části IP adres určených jedničkami v masce a jednoduše porovnáme, že jsou obě stejné, tudíž patří obě adresy do stejné sítě.

## Nalezení Network ID

Nalezení první adresy podsítě se může zdát jednoduché, ale někdy to na první pohled vidět není. Pak musíme použít logiku nebo matematiku. Network ID se dá vypočítat z binárního zápisu adresy a masky, kdy se provede bitové AND, `networ-ID = IP-adresa AND subnet-mask`.

Příklad pro adresu `10.217.123.7/20`

| | |
|---|---|
| IP binárně | 00001010.11011001.01111011.00000111 |
| maska binárně | 11111111.11111111.11110000.00000000 |
| operace AND | 00001010.11011001.01110000.00000000 |
| dekadicky | 10.217.112.0 |

Místo matematiky můžeme použít i logické uvažování. Maska /20 znamená, že dochází ke změně ve 3 oktetu, takže první 2 oktety zůstávají zachovány a poslední je nulový. Ve 3. oktetu máme čtyři jedničky a čtyři nuly, takže adresy podsítí jdou po 16ti (24). To znamená 0 až 15, 16 až 31, atd., pro hodnotu 123 je nejbližší nižší hodnota 112.

## Nalezení broadcast adresy

Broadcastovou adresu subnetu nalezneme podobně jako network ID. Matematicky můžeme použít bitové OR mezi IP adresou a negovanou maskou, tedy `broadcast-adresa = IP-adresa OR NOT(subnet-mask)`. Příklad pro stejnou adresu jako minule `10.217.123.7/20`.

| | |
|---|---|
| IP binárně | 00001010.11011001.01111011.00000111 |
| maska binárně | 11111111.11111111.11110000.00000000 |
| negace masky | 00000000.00000000.00001111.11111111 |
| operace OR | 00001010.11011001.01111111.11111111 |
| dekadicky | **10.217.127.255** |

Jiná možnost výpočtu je, když známe network ID, pak stačí do části uzlů v binárním tvaru zadat samé jedničky.

## Jiné výpočty

Otázky na to, co chceme spočítat mohou být různé, ale vše vychází z předchozích úvah. Například chceme najít masku podsítě, aby se v ní nacházel zadaný počet adres. Převedeme počet adres na binární hodnotu, z toho dostaneme jak dlouhá je část uzlů a doplněk je maska podsítě. Tedy například subnet pro 4000 klientů, binárně `1111.10100000`, to je délka `12`, maska je dlouhá `32 - 12 = 20`. Subnet `/20` může obsahovat až `4094` adres uzlů.

Jiný příklad může být, když chceme rozdělit síť na určitý počet stejných subnetů. Vezmeme hodnotu počtu subnetů, odečteme 1 (protože nabývá hodnot 0 až n-1), převedeme na binární hodnotu, ta musí obsahovat samé jedničky, jinak není možno dosáhnout tohoto počtu subnetů. Pokud počítáme v rámci nějaké třídy, tak připočteme masku třídy a máme cílovou masku jednoho subnetu, kterých může vzniknout zadaný počet.

Například síť `192.168.100.0/24` v rámci třídy C rozdělit na 8 stejných podsítí. `8 - 1 = 7` binárně `111`, v rámci třídy C máme masku `24 + 3 = 27`, takže možné subnety jsou `192.168.100.0/27, 192.168.100.32/27, 192.168.100.64/27, 192.168.100.96/27, 192.168.100.128/27, 192.168.100.160/27, 192.168.100.192/27, 192.168.100.224/27`.

> ***Kalkulačky***
> *Pro výpočty různých hodnot okolo adresování se používají kalkulačky IP podsítí (IP subnet calculator). Takových aplikací nalezneme celou řadu a velká část jich je online na internetu. Příkladem je* ***subnet-calculator.samuraj-cz.com***.



Autor: **Petr Bouška**

**Související články:**

## Základy počítačových sítí

Tento seriál jsem napsal pro časopis Connect. Většinu informací obsahuje shodně jako můj starší seriál Počítačové sítě - Computer networks, ale je napsán trochu jiným způsobem. Nejprve jsou stručně shrnuty technologie počítačových sítí a pak se trochu podrobněji probírají od nejnižších vrstev nahoru.

- **Rozumíme počítačovým sítím** [09.05.2010 21:27]
- **Víte, jak pracuje switch?** [25.05.2010 10:39]
- **Víte, jak pracuje router?** [23.06.2010 18:30]
- **Adresování v IP sítích** [21.07.2010 12:39] ..... **právě čtete**

# Počítačové sítě a jejich typy

Pondělí, 09.07.2007 15:32 | Samuraj - Petr Bouška | sítě

Tímto článkem, který pouze shrnuje obecné základy o počítačových sítích, bych chtěl začít sérii článků o základech počítačových sítí (computer networks), jinak řečeno networkingu. V článku naleznete stručný popis termínu počítačová síť a základní dělení sítí.

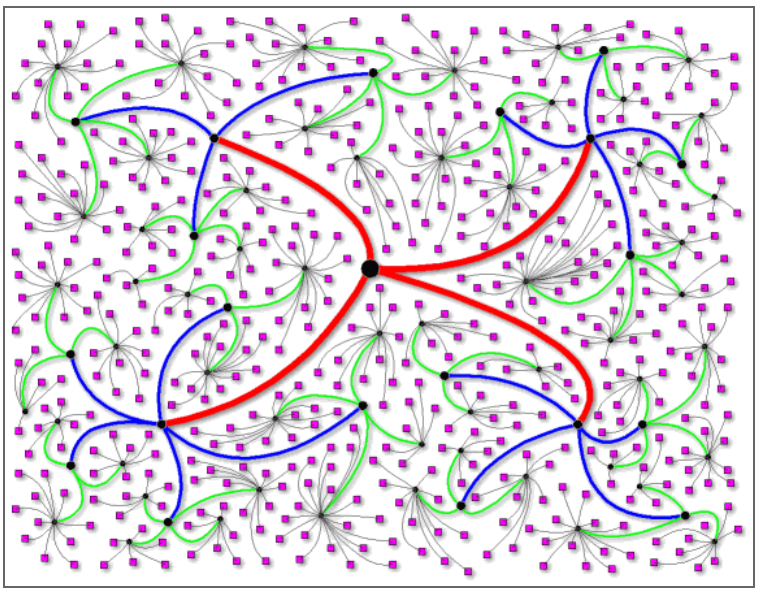


## Počítačová síť - computer network

***Počítačová síť*** vznikne ve chvíli, kdy dva (někdy se říká minimálně tři) nebo více počítačů propojíme dohromady pomocí telekomunikačního systému za účelem sdílení zdrojů. V praxi je dnes nejrozšířenější síť založena na technologii ***ethernet*** a používá protokol ***TCP/IP***.

Počítačové sítě se dělí podle řady kritérií. Například podle ***velikosti/měřítka*** na PAN (Personal Area Network), LAN (Local Area Network), CAN (Campus Area Network), MAN (Metropolitan Area Network) či WAN (Wide Area Network). Podle ***technologie*** použité pro spojení zařízení na Ethernet, WLAN, apod. Nebo podle ***síťové topologie*** na topologie sběrnice, hvězda, apod.

Zjednodušeně můžeme počítačovou síť (jejímž velkým reprezentantem je například internet) zobrazit pomocí stromové struktury. Hezký schematický obrázek i s popisem, jsem nalezl v článku The Internet: How it works. Samozřejmě skutečná situace je složitější, protože řada linek je zdvojená a propojená na více místech.



## Local Area Network - LAN

***Lokální počítačová síť*** se vyznačuje tím, že počítače jsou propojeny na menším geografickém území (tedy v rámci firmy, budovy, místnosti, atp.). V rámci LAN se nejvíce používá ***přepínaný ethernet*** nebo ***WiFi*** (IEEE802.11). Infrastruktura je většinou tvořena metalickými kabely a případně optickou páteří.

LAN může být samostatná síť, které propojuje řadu zařízení, ale v dnešní době je většinou propojena do internetu, tedy WAN sítě.

## Metropolitan area network - MAN

Síť, která spojuje jednotlivé LAN, ale nepřekračuje hranice města či metropolitní oblasti, se označuje jako ***metropolitní síť*** - MAN. V rámci MAN se často používá bezdrátové spojení nebo optická vlákna.

MAN může být vlastněna jednou organizací, ale většinou se jedná o propojení několika nezávislých objektů. Můžeme mít například několik poboček firmy v jednom městě propojených do MAN sítě.

Dříve se využívalo technologií jako *ATM* a *FDDI*, ale dnes jsou většinově nahrazovány ethernetem označovaným jako ***metro-ethernet***.

## Wide Area Network - WAN

WAN je komunikační síť, která pokrývá rozsáhlé území, jako je spojení zemí či kontinentů. Obecně můžeme říct, že jednotlivé LAN sítě se propojují přes WAN síť, aby se zajistila komunikace na velké vzdálenosti. Tímto způsobem pracuje internet jako nejrozsáhlejší a nejznámější WAN.

Nejvíce se dnes asi používá technologie *Frame relay* případně *ATM*.

## Virtual Local Area Network - VLAN

***Virtuální LAN*** je obdobou klasické lokální sítě s tím, že LAN závisí na *fyzickém uspořádání* a propojení, kdežto VLAN vzniká *logicky* uvnitř fyzické LAN.

# Wireless Local Area Network - WLAN

***Bezdrátová lokální síť*** je opět obdobou běžné LAN, ale jednotlivé prvky nejsou fyzicky propojeny drátem (metalikou či optikou), ale jsou propojeny bezdrátově. Využívají se rádiové vlny a určitá modulace pro přenos dat.
Výhoda bezdrátu je jasná pro mobilní zařízení. Nevýhodou je například to, že se špatně omezuje šíření signálu, a případný útočník nemusí získat přímo fyzický přístup k zásuvce, jako v případě drátových sítí.



Autor: Petr Bouška

**Související články:**

## Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

# Počítačové sítě - základní topologie

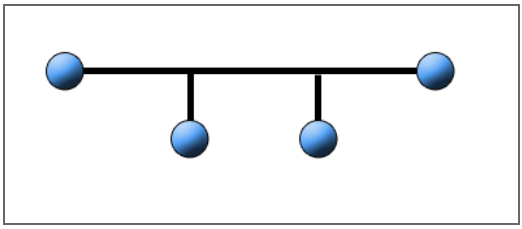
**Upraveno** 07.04.2009 18:16 | vytvořeno 30.07.2007 14:47 | Samuraj - Petr Bouška | *sítě*

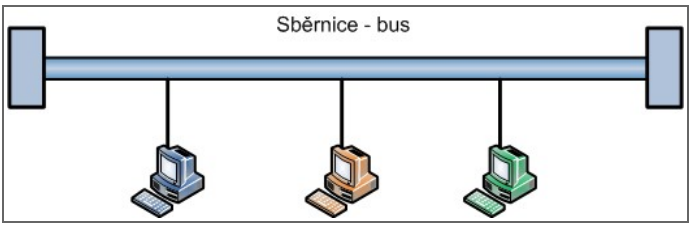
V druhé části popisu základních termínů a funkcí počítačových sítí se stručně věnuji síťovým topologiím. Síťová topologie popisuje uspořádání síťových prvků, ať už fyzické nebo logické. Základní topologie používané v lokálních sítích (LAN) jsou sběrnice, hvězda a kruh.



## Sběrnice (bus)

***Sběrnice*** byla používána v prvních dobách ethernetu a realizovala se pomocí koaxiálního kabelu a BNC konektorů, na konci musel být vždy terminátor. Všechna zařízení jsou zapojena na společnou sběrnici. V sítích se od této technologie ustoupilo a dnes se používá převážně zapojení do hvězdy.



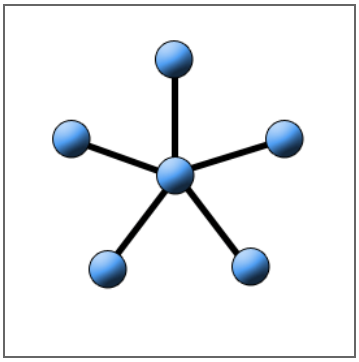


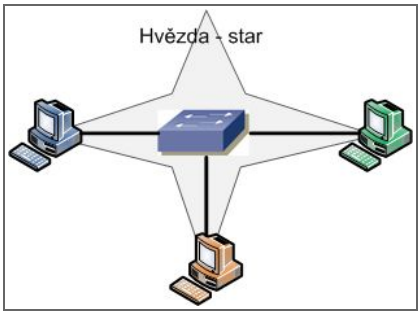

## Hvězda (star) - Hub and Spoke

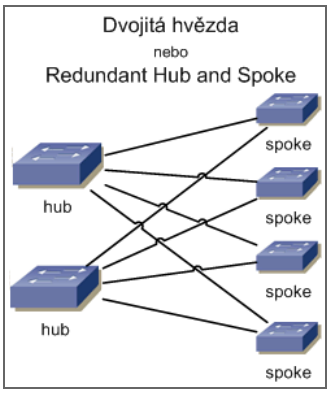
***Hvězda*** je dnes nejpoužívanější topologie pro ethernet. Je zde centrální prvek, který realizuje propojení zařízení, a do něj jsou připojena jednotlivá zařízení. Jako centrální prvek slouží hub nebo switch, ale z jiného pohledu se může jednat i o router.

Obdobná je ***Rozšířená topologie hvězda***, která vznikne, když několik samostatných *hvězd* propojíme dohromady přes centrální prvky. V praxi se dnes často používá tří vrstvá topologie (případně dvou vrstvá, kdy se jedna vrstva vynechá) využívající propojení do hvězdy. V nejvyšší vrstvě je jeden nebo dva (pro redundanci) switche, tzv. ***jádro*** (core). Druhá vrstva obsahuje několik switchů připojených do jádra a zvaných ***distribuční*** (distributed). Poslední vrstva, připojená do distribuční, je ***přístupová*** (access) a do ní se připojují stanice a servery. Některé důležité servery se mohou připojit přímo do distribuční vrstvy. Zapojení, kdy zdvojíme centrální prvek, se také označuje jako ***Dvojitá hvězda***.

Topologie hvězda se také označuje jako zapojení ***Hub and Spoke***. Zde není použit termín ***hub*** jako aktivní síťový prvek, ale je to odvozeno od označení pro kolo u vozu, kde máme střed (Hub) a paprsky/loukotě (Spoke). Tento termín se používá také u routování nebo VPN, kdy ***Hub*** je centrála/centrální prvek a ***Spoke*** jsou pobočky.

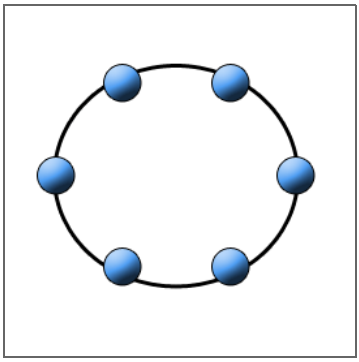


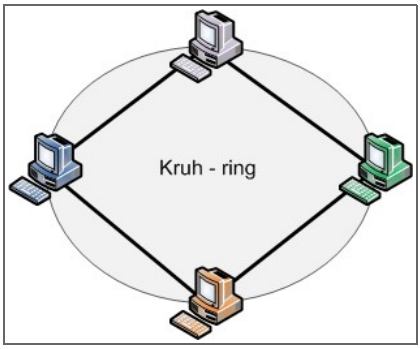



## Kruh (ring)

V ***kruhové topologii*** je každý uzel připojen ke dvěma sousedním a dohromady tvoří kruh. Standardně existuje pouze jedna cesta mezi dvěma uzly. Rozšířením je, že komunikace probíhá ve směru i proti směru hodinových ručiček. Používá se pro síťové technologie FDDI a Token Ring.

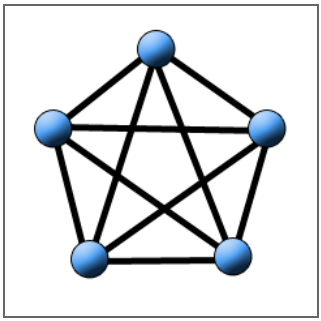





## Mřížka (mesh)

V topologii mesh jsou uzly propojeny s více sousedy. Buď se může jednat o ***Full Mesh*** (plnou mřížku), kdy je každý uzel spojený se všemi ostatními, takže může komunikovat s každým přímo a v případě výpadku nějaké linky může jednoduše nalézt cestu. Ale při více uzlech se jedná o složité a drahé zapojení. Nebo o ***Partial Mesh*** (částečnou mřížku), kdy některé uzly jsou přímo spojeny (point-to-point) s více jinými uzly.

Logická Full Mesh se používá například pro routování BGP protokolem. Dále se Mesh topologie používá v některých WiFi sítích.





Autor: **Petr Bouška**

**Související články:**

## Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

- **OSI model** [05.03.2007 08:09]
- **VLAN - Virtual Local Area Network** [02.06.2007 15:54]
- **Počítačové sítě a jejich typy** [09.07.2007 15:32]
- **Počítačové sítě - základní topologie** [30.07.2007 14:47] ..... **právě čtete**
- **Ethernet - CSMA/CD, kolizní doména, duplex** [03.08.2007 12:40]
- **TCP/IP - model, encapsulace, paket vs. rámec** [16.08.2007 16:43]
- **TCP/IP - metody vysílání dat** [02.09.2007 15:05]
- **TCP/IP - adresy, masky, subnety a výpočty** [05.09.2007 14:53]
- **TCP/IP - navázání a ukončení spojení** [13.09.2007 16:10]
- **TCP/IP a ethernet - cesta v síti, aktivní síťové prvky** [17.09.2007 10:22]
- **TCP/IP - Routing - směrování** [21.09.2007 15:34]
- **TCP/IP - nalezení MAC adresy k IP - ARP** [25.09.2007 17:35]
- **Počítačové sítě - Computer Networks** [30.09.2007 13:28]
- **DNS (Domain Name System) zaměřeno na Microsoft** [26.11.2007 14:26]
- **Diagramy a schémata počítačové sítě (Visio)** [13.01.2008 18:02]
- **Wake on LAN - lokální i vzdálený subnet** [10.08.2008 20:42]
- **TCP/IP - Internet Protocol Version 6 - IPv6** [05.03.2009 15:41]
- **TCP/IP - skupinové vysílání IP Multicast a Cisco** [10.03.2009 20:05]
- **Běžné útoky na switche, Cisco Dynamic ARP Inspection** [18.06.2009 10:15]
- **Cisco - Router Switching metody a související termíny - CAM, FIB, CEF** [28.06.2009 17:15]
- **Výstavba moderní sítě** [28.04.2010 14:41]
- **Microsoft Network Load Balancing (NLB) a Cisco switche** [09.08.2010 18:26]
- **Cisco Network Design - Enterprise Architecture** [24.09.2010 13:47]
- **VPN 1 - IPsec VPN a Cisco** [10.04.2011 19:59]

# OSI model

Pondělí, 05.03.2007 08:09 | Samuraj - Petr Bouška | *sítě*

Třetí část seriálu o počítačových sítích. OSI model je zažitý termín pro Open Systems Interconnection Basic Reference Model. Jedná se o abstraktní popis síťové komunikace a protokolů použitých pro komunikaci mezi počítači, který je dělen do sedmi vrstev. Při komunikaci se provádí zapouzdření - encapsulate na straně odesílatele a rozbalování na straně příjemce. Začíná se vrstvou 7, ta se zabalí do vrstvy 6, atd. Tento model byl vytvořen, aby se standardizovala komunikace a hardware i software různých výrobců mohl vzájemně komunikovat.



Popis ***OSI modelu*** vznikl již před řadou let a přesto, že se považuje za základ pro síťové technologie, tak nebyl nikdy přesně realizován. Obdobou OSI modelu je **TCP/IP**, u kterého můžeme říci, že vychází z OSI modelu, ale upravuje jej, aby byl více flexibilní. Podle ***OSI modelu*** je vždy možno komunikovat pouze s vrstvou nad nebo pod. A všechny vrstvy musí být v komunikaci obsaženy, což v řadě praktických úloh přináší zbytečnou zátěž (časovou i datovou). Přesto je OSI model dobrý pro výklad a teoretický popis sítí.

| | vrstva AJ | vrstva CZ | jednotka | funkce vrstvy | příklad |
|---|---|---|---|---|---|
| **Layer 7** | Application | Aplikační | Data | Síťové procesy pro aplikaci, ověření uživatelů, vše závislé na aplikaci. | Telnet, FTP |
| **Layer 6** | Presentation | Prezentační | Data | Reprezentace dat a šifrování. Řeší rozdíly v reprezentaci dat mezi aplikací a síťovým formátem - kóduje data pro přenos. | MIDI, MPEG |
| **Layer 5** | Session | Relační | Data | Spojení mezi aplikacemi, správa session. Komunikace jedné aplikace s druhou, posílání více dat po sobě. Udržuje celé spojení mezi dvěma počítači. | NetBIOS |
| **Layer 4** | Transport | Transportní | Segments (segmenty) | End-to-end spojení systémů, spolehlivost - zajišťuje kompletní přenos dat, kvalita služby. Řeší spolehlivé odeslání všech dat ze zdroje do cíle pomocí segmentace a potvrzování. | TCP, UDP |
| **Layer 3** | Network | Síťová | Packets (pakety) | Logická adresace - routování - určení cesty paketu, přenos dat z bodu do bodu, používá IP adresy. Komunikace mezi zdrojovým a cílovým zařízením pomocí IP adresy. | IP, ICMP, ARP, RIP |
| **Layer 2** | Data Link | Linková | Frames (rámce) | Fyzická adresace, MAC - media access control a LLC - logical link control, datový tok, synchronizace rámů, komunikace 1 hop, používá MAC adresy. Detekce chyb, řízení toku a přístupu na médium. Komunikace mezi dvěma zařízeními v jednom subnetu (nebo na bránu) pomocí MAC adresy. Vytváří rámce (hlavička + data + zápatí). | Ethernet, FDDI, Token Ring, PPP, SLIP |
| **Layer 1** | Physical | Fyzická | Bits (bity) | Fyzické parametry linky - média (kabely, rádio, světlo), signály a binární přenos. Řeší fyzické poslání dat (přenášeným bitům nepřiřazuje žádný význam). | 100BaseT, RS-232, 802.11g |

## Srovnání OSI vs. TCP/IP

V praktickém případě a hlavně, když používáme protokol TCP/IP, můžeme použít TCP/IP model. Hrubé srovnání s ISO OSI modelem je zde.

| TCP/IP | OSI |
|---|---|
| Aplikační | Aplikační |
| | Prezentační |
| | Relační |
| Transportní | Transportní |
| Síťová | Síťová |
| Vrstva síťového rozhraní | Linková |
| | Fyzická |

# Ethernet - CSMA/CD, kolizní doména, duplex

Pátek, 03.08.2007 12:40 | Samuraj - Petr Bouška | sítě

Čtvrtá část popisu základních termínů počítačových sítí se věnuje nejpoužívanější síťové technologii (hlavně v LAN) Ethernetu. Zároveň je zde popis souvisejících termínů kolizní doména a duplex (half a full). Pokud byste hledali třetí část, tak to je popis ISO/OSI modelu, který jsem zveřejnil již dříve.



## Ethernet

**Ethernet** je rozsáhlá rodina ***síťových technologií***. Ethernet dříve využíval pro přenos dat sdílené komunikační médium, kterým může být metalický či optický kabel. Ethernet je specifikován normou **IEEE 802.3**.

***Ethernet*** je majoritní technologií při budování lokálních sítí (LAN), ale má velké zastoupení i u větších sítí. Síťová technologie spadá v ISO OSI modelu do ***druhé (linkové) vrstvy***, ale také do ***první (fyzické) vrstvy***. Definuje řadu parametrů, co se týče kabeláže (přenosového média) a signálů, to je 1. vrstva. Ale také popisuje způsob přístupu k síti a obecný adresovací formát, to je 2. vrstva.

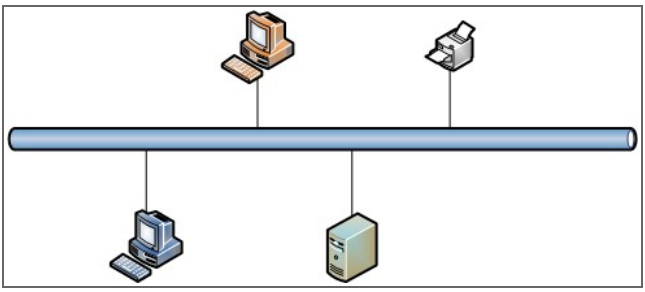
Příkladem dalších síťových technologií, které byly většinově nahrazeny *Ethernetem*, je *FDDI*, *Token Ring* či *ARCNET*. Druhou technologií, která má význačnější podíl v LAN a jejíž použití roste, je ***WiFi*** - bezdrátová síť, standardizovaná v **IEEE 802.11**.

Původně se jako přenosové médium využíval ***tlustý koaxiální kabel*** (10Base-5 či slabý koaxiální kabel 10Base-2) v zapojení sdílené sběrnice. Přenosová rychlost byla 10Mb/s. Postupně byl nahrazen ***kroucenou dvojlinkou*** (twisted pair) zapojenou do hvězdy (využíval se hub). Při rychlosti 10Mb/s se označoval pouze *ethernet* (10Base-T). Poslední médium pro Ethernet je ***optické vlákno*** (10Base-F). Postupem času rostly rychlosti na 100Mb/s označované jako ***Fast Ethernet*** (třeba 100Base-TX), 1Gb/s označované jako ***Gigabit Ethernet*** (třeba 1000Base-T) a dnes máme standard i pro ***Desetigigabitový Ethernet*** (10GBase-T). Stejně tak byly *huby* nahrazeny ***switchi***, tedy spoji bod-bod, a technologie se označuje ***přepínaný Ethernet*** (switched Ethernet), kde se již nevyužívá sdílené médium.

Pro určité typy Ethernetu je vždy definován patřičný kabel. Pro optiku můžeme mít jednovidová či mnohovidová vlákna daných parametrů. Pro metaliku se používá stíněná (STP - *Shielded Twisted Pair*) či nestíněná (UTP - *Unshielded Twisted Pair*) kroucená dvojlinka. Navíc je řazena do několika kategorií podle vlastností a možnosti použití pro určitý ethernet, máme například UTP kabel kategorie 3, 4, 5, 5e, 6, 6a či 7.

## Přístup k médiu - CSMA/CD

Když je více zařízení, připojeno ke sdílené sběrnici, tak v jednu chvíli může komunikovat pouze jedno a ostatní mohou pouze naslouchat. Problém nastane, když více zařízení začne komunikovat najednou. V té chvíli dojde ke **kolizi**. Pro vysílání (přístup k médiu) se používá ***Carrier Sense Multiple Access With Collision Detection*** (CSMA/CD) jako síťový kontrolní protokol.

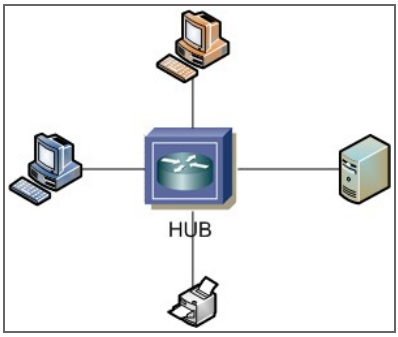


## Algoritmus CSMA/CD

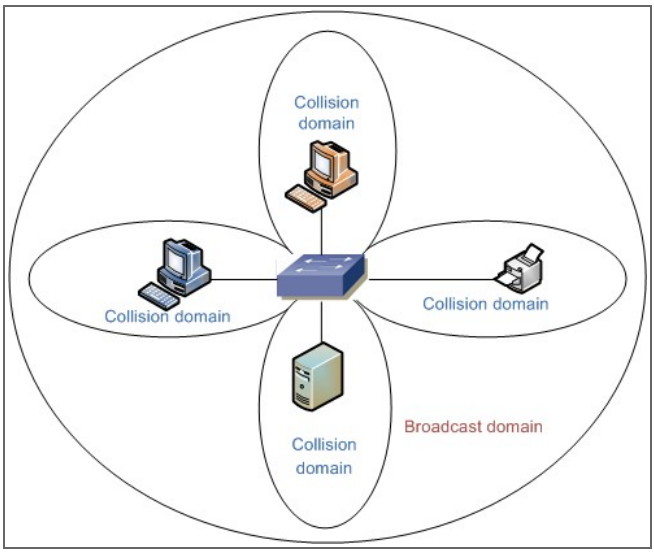
1. zařízení, které chce vysílat, poslouchá, zda je linka volná
2. pokud je volná, začne vysílat rámec
3. odesílatel poslouchá, zda nedošlo ke kolizi
4. pokud zjistí, že došlo ke kolizi, pošle jam signál
5. po ukončení jamu si stanice, které vysílaly nastaví náhodný čas po který čekají
6. po uplynutí času se začne bodem 1

## Kolizní doména

**Kolizní doména** (collision domain) je část sítě, kde může dojít ke kolizi vysílání několika stanic. Tedy skupina zařízení, která jsou připojena na sdílené médium. To je v praxi příklad starých síťových rozvodů pomocí koaxiálního kabelu. Nebo používání ***hubů*** (rozbočovač), které přijatý paket přeposílají na všechna rozhraní mimo toho odkud přišel.




Dříve se pro rozdělení sítě na více kolizních domén (a jejich zmenšení) provedlo to, že se síť rozdělila na více segmentů a ty se propojily pomocí routeru. Pak se ale objevily ***switche*** (přepínač), které z principu své funkce, vytvářejí samostatné kolizní domény na každém rozhranní. Switch totiž posílá přijatý paket pouze na jedno rozhranní (v naučeném stavu) a provádí přepínání paketů.




Pokud je do switche připojeno koncové zařízení nebo jiný switch a linka je plně duplexní, tak na této lince nemůže dojít ke kolizi. Prakticky to totiž znamená, že máme dvě komunikující zařízení na lince, a každé zařízení posílá data po vlastním páru vodičů.

## Duplex

**Half duplex** - poloviční duplex znamená, že aktivní může být komunikace pouze v jednom směru.

**Full duplex** - plný duplex je ve chvíli, kdy můžeme zároveň vysílat i přijímat. V ethernetu se využívají dva páry vodičů (většinou, ale někdy čtyři), kdy polovina je použita pouze pro vysílání a polovina pro příjem. Odpadají zde problémy a zpoždění způsobená algoritmem CSMA/CD a teoretická rychlost je dvojnásobná (u fast ethernetu je 100Mb/s použito pro vysílání a 100Mb/s pro příjem).

> ***Pozn.:*** *Huby nepodporují plně duplexní komunikaci, protože, zde může dojít ke kolizi. Pokud dvě stanice vysílají najednou, hub to přijme a odešle na ostatní porty, takže vlastně odesílá kolizi.*



**Související články:**

## Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

# TCP/IP - model, encapsulace, paket vs. rámec

Čtvrtek, 16.08.2007 16:43 | Samuraj - Petr Bouška | sítě

V pátém díle seriálu o počítačových sítích stručně sepisuji základní informace o nejpoužívanějším protokolu TCP/IP. Je zde zmíněn TCP/IP model a princip encapsulace při odesílání dat. Pokouším se popsat rozdíly mezi běžnými termíny paket (packet) a rámec (frame). A nakonec je zobrazen formát rámce.



**Internet protocol suite**, bežně označovaná jako **TCP/IP**, je sada komunikačních protokolů, kde hlavními jsou protokoly ***Transmission Control Protocol*** (TCP) a ***Internet Protocol*** (IP).

## TCP/IP model

Klasický čtyřvrstvý TCP/IP model, někdy zvaný také ***Internet Reference Model***, vychází z *OSI modelu*, ale více se blíží praxi.

| vrstva | jméno CZ | jméno EN | příklady aplikací dané vrtsvy |
|---|---|---|---|
| layer 4 | Aplikační | Application | Telnet, FTP, POP3, HTTP, DNS, SNMP |
| layer 3 | Transportní | Transport | TCP, UDP |
| layer 2 | Síťová | Network (Internetwork/Internet) | IP, ICMP, IGMP, ARP, STP |
| layer 1 | Síťové rozhranní (přístupová) | Network Interface (Access) | ovladač síťové karty (Ethernet, Frame Relay) |

Někdy se nejnižší vrstva rozděluje na dvě části. Na vrstvu ***fyzickou*** (Physical Layer) a vrstvu ***linkovou*** (Data link).

## Encapsulace při odesílání dat

Při odesílání dat se provádí ***encapsulace*** (zapouzdřování - zabalování) od nejvyšší vrstvy dolů. Probíhá to tak, že aplikace vezme data (aplikační vrstva), která chce zaslat jiné stanici a doplní je o aplikační hlavičku. Tato data zašle nižší vrstvě (transportní), která odesílaná data rozdělí na segmenty, zabalí a přidá TCP (nebo UDP) hlavičku a vytvoří ***TCP segment***. Další vrstva (síťová) doplní IP hlavičku a takto vznikne ***IP paket*** (někdy označovaný jako IP datagram). V poslední vrstvě (přístupové) se k paketu přidá ethernetová hlavička na začátek a trailer na konec, ten obsahuje ***FCS*** (Frame Check Sequence) - kontrolní součet (často se pro jeho výpočet používá CRC - cyclic redundancy check). Takto v posledním kroku vznikne ***ethernetový rámec*** (ethernet frame), který se vysílá na komunikační médium.

> ***Pozn.:*** *Výše uvedený popis se týkal protokolu IP, pokud se jedná o nějaký pomocný protokol (třeba ARP, ICMP), tak je situace obdobná.*

Když cílové zařízení přijme data, provádí se obrácený postup ***deencapsulace*** (rozbalování) od nejnižší vrstvy nahoru a cílová aplikace dostane odesílaná data.

## Paket vs. rámec

V souvislosti s daty a jejich přenosem po síti se používají dva důležité termíny, jsou to ***paket*** a ***rámec***. Tyto termíny jsou hojně používány v literatuře, na internetu, i v praxi. Bohužel v literatuře existuje několik odlišných vysvětlení a také používání těchto výrazů je různé. Navíc záleží na technologii, u které se používají (podobné termíny se používají třeba i u sériové komunikace). Uvádím zde vysvětlení těchto termínů, které se mi zdá nejsprávnější.

**Paket** (packet) v překladu znamená balíček a jedná se o formátovaný blok dat, který se přenáší v počítačové síti. O paketech se mluví v souvislosti se síťovou vrstvou. Paket obsahuje IP adresu, další atributy a data. Zabalí se do rámce a následně putuje sítí.

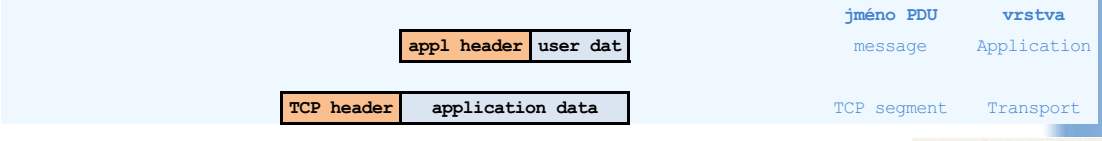
**Rámec** (frame) je to, co skutečně putuje v síti. Rámce vznikají až na fyzického vrstvě síťového rozhranní. Název naznačuje, že se spíše než o objekt jedná o časový úsek. Rámců existuje více typů, nejpoužívanější je ***Ethernet II***, který umí přepravovat TCP/IP i IPX/SPX.

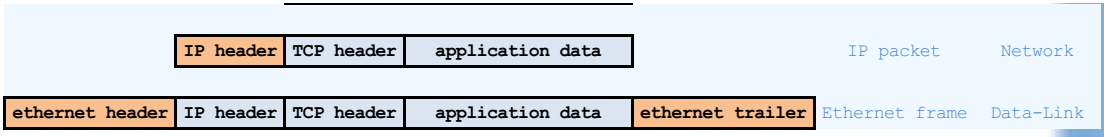
Maximální velikost dat přenášených v rámci (tedy velikost paketu) je dána pomocí ***MTU*** (Maximum transmission unit) a váže se k linkové vrstvě. Standardní velikost MTU v ethernetu je 46 - 1500 bytů. Na začátku komunikace se zařízení pomocí ICMP domluví na velikosti MTU. Pokud vyžaduje druhá strana menší MTU, může se provést ***fragmentace*** - rozdělení rámce do více menších. Fragmentaci je možno zakázat příznakem v hlavičce paketu. Pokud je fragmentace zakázána a druhá strana akceptuje pouze menší MTU, tak odpoví chybou ICMP.

V obecné terminologii se používá označení ***Protocol Data Unit*** (PDU), což je datová jednotka na libovolné vrstvě (IP paket, rámec .). Když se PDU předá na nižší vrstvu, tak se označuje jako ***Service Data Unit*** (SDU), tam se doplní o ***Protocol Control Information*** (PCI), tedy hlavičku, a vznikne nové PDU.

## Formát paketu - rámce

Základní tvar rámce je ***hlavička*** (header), ***data*** (payload) a ***zakončení*** (trailer). Data jsou složena ze zapouzdřených dat vyšších vrstev. Následující obrázek zobrazuje čtyři vrstvy TCP/IP modelu spolu s hrubým formátem dat, jak jsou zapouzdřována odshora dolů. Uvedený příklad je pro protokol TCP, ale obecný model je shodný.






## Ethernetová hlavička

**Ethernetová hlavička** obsahuje zdrojovou a cílovou MAC adresu (fyzickou adresu od hopu k hopu) a typ/délku. Typ v hlavičce určuje jaký protokol je použit na vyšší vrstvě (IP, ARP, atp.). Pro rozlišení jednotlivých rámců se na začátku vysílá speciální úvod (preamble) složený ze sekvence střídajících se jedniček a nul a SFD - oddělovač začátku rámce.

| úvod | SFD | cílová MAC adresa | zdrojová MAC adresa | typ / délka | data (payload) | CRC |
|---|---|---|---|---|---|---|
| 7B | 1B | 6B | 6B | 2B | 46-1500B | 4B |

## IP hlavička

**Ip hlavička** obsahuje zdrojovou a cílovou IP adresu (logickou adresu komunikujících stran), dále určení protokolu pro další vrstvu (TCP, UDP, ICMP, apod.), kontrolní součet hlavičky a další hodnoty. Standardní velikost IP hlavičky je 20B, ale může ještě obsahovat volitelné vlastnosti. Tvar IP hlavičky záleží na použitém protokolu, následující příklad je pro IPv4.

| bits | 0-3 | 4-7 | 8-15 | 16-18 | 19-31 |
|---|---|---|---|---|---|
| 0 | 4 | header length | Type of Service | total length (header + data) | |
| 32 | identification | | | flags | fragment offset |
| 64 | TTL | | protocol | header checksum | |
| 96 | source IP | | | | |
| 128 | destination IP | | | | |
| 160 | options (if any) | | | | |
| 160/192+ | DATA | | | | |

## TCP hlavička

**Tcp hlavička** obsahuje zdrojový a cílový port, číslo sekvence a odpovědi (pro zajištění spolehlivosti), příznaky (kam patří SYN a ACK pro navázání spojení), velikost potvrzovacího okna a další. Standardní velikost je opět 20B.

| bits | 0-3 | 4-7 | 8-15 | 16-31 |
|---|---|---|---|---|
| 0 | Source Port | | | Destination Port |
| 32 | Sequence Number | | | |
| 64 | Acknowledgment Number | | | |
| 96 | Data Offset | Reserved | Flags | Window |
| 128 | Checksum | | | Urgent Pointer |
| 160 | Options (optional) | | | |
| 160/192+ | DATA | | | |



Autor: Petr Bouška

**Související články:**

## Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

# TCP/IP - metody vysílání dat

**Upraveno** 11.08.2008 13:05 | vytvořeno 02.09.2007 15:05 | Samuraj - Petr Bouška | *sítě*

Šestá část základů počítačových sítí se věnuje metodám vysílání (nevím, zda je to nejvhodnější označení). Je zde stručně popsáno broadcastové, unicastové a multicastové vysílání v síti.

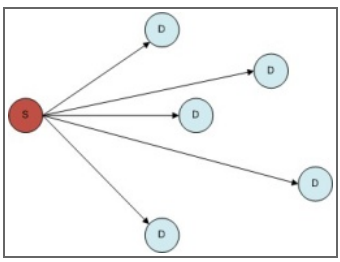


*Článek byl doplněn 11.8. 2008.*

Když chceme poslat nějaká data v počítačové síti, je důležité správné adresování zprávy a s tím je svázána metoda vysílání. Záleží na tom, zda chceme data zaslat pouze jedné stanici, několika stanicím najednou nebo všem stanicím ve stejném subnetu.

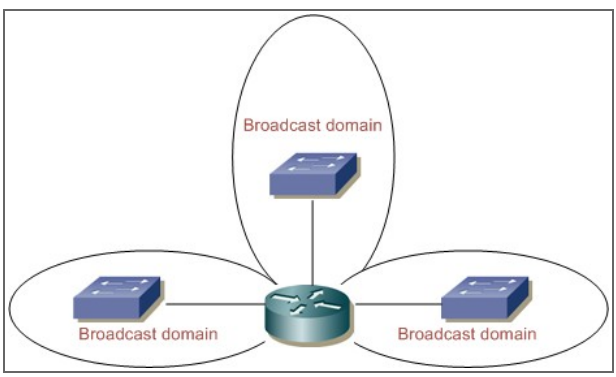
## Broadcast

**Broadcast** je jeden ze způsobů vysílání v TCP/IP síti (používá se nespojový (connectionless) protokol, jako třeba UDP). Je to vysílání jednoho všem, tedy vysílaný paket je (teoreticky) zachycen všemi zařízeními v síti, lepé řečeno v dané broadcast doméně (subnetu). Toto vysílání se používá převážně v LAN sítích (ne WAN). Broadcasty jsou dnes používány pro řadu účelů (třeba DHCP, ARP) a využívá je velké množství aplikací. Tvoří velkou část provozu v LAN síti a zatěžují aktivní síťové prvky i stanice, proto je snaha o jejich minimalizaci.




**Broadcast adresa** je adresa, na kterou se posílá komunikace v případě brodcastového vysílání. Jedná se o adresu, která má binárně samé jedničky. Na 3. vrstvě OSI modelu je to IP adresa `255.255.255.255` (občas se tak nepřesně označuje i broadcastová adresa subnetu, tedy poslední adresa v subnetu, viz dále). Na 2. vrstvě (fyzické) je broadcastová MAC adresa identicky adresa "se samými jedničkami" `FF:FF:FF:FF:FF:FF`.

**Broadcast domain** je logická část sítě, ve které mohou připojená zařízení přímo komunikovat. Mělo by se tedy jednat o jeden subnet a hranicí je router, který by broadcasty neměl přeposílat.




Pro rozdělení sítě na menší broadcast domény slouží routery nebo technologie VLAN.

**Broadcast storm** je kritický stav, když počet broadcastů v síti je tak velký, že není možno vytvářet nová spojení a stará spojení jsou rušena. Často vzniká z důvodu smyček v síti.

## Směrovaný broadcast na subnet - Subnet Directed Broadcast

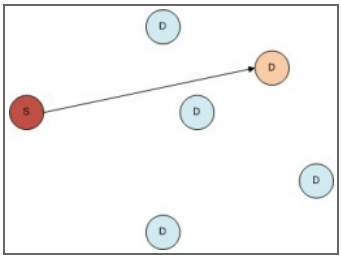
**Subnet Directed Broadcast** je speciálním případem broadcastu, který se použije ve chvíli, kdy chceme zaslat broadcast do jiného subnetu, než v kterém se nacházíme. Funguje to tak, že rámec (frame) putuje sítí jako unicast přes routery až na router, který má cílový subnet (určený broadcastovou adresou subnetu) jako přímo připojený. Tento router upraví rámec na běžný broadcast a odešle jej na odpovídající rozhranní. Protože však tato technika byla používána k útokům, tak bývá na routerech nebo i firewallech většinou zakázána. To se týká routeru, který by měl rámec převést na broadcast, přes ostatní by měl rámec projít normálně.

**Subnet broadcast address** (broadcast adresa subnetu, někdy také nazývaná directed broadcast address) je IP adresa, která se skládá z network ID a samých jedniček v host ID. Jinak řečeno, jedná se o poslední adresu subnetu (ta se nemůže přiřadit stanici). Pro jednoduchý příklad máme počítač s adresou `192.168.10.39` a maskou `255.255.255.0`, podle masky vidíme, že první tři oktety tvoří network ID a poslední oktet jsou host ID, takže broadcast adresa subnetu je `192.168.10.255`.

**Smurf Attack** je nejběžnějším případem útoku pomocí Subnet Directed Broadcast, který má zařídit ***Denial of Services*** (DoS). Útočník odesílá množství paketů `ICMP echo reply` s podvrženou zdrojovou adresou jako directed broadcast, všechny stanice v subnetu odpovídají `echo reply` a stanice, jejíž adresa je ta podvržená, je zahlcen

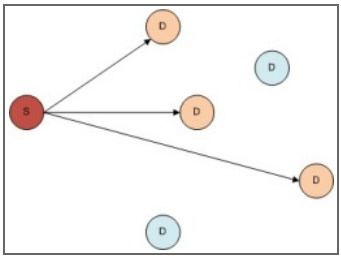
## Unicast

***Unicast*** je vysílání, kdy je paket zasílán jednomu cíli. Jedná se o běžnou komunikaci, kdy spolu komunikují dvě stanice.




## Multicast

**P**ři této komunikaci se jedna informace doručuje skupině cílů. ***Multicast*** využívá efektivní metodu doručování, aby pakety v síti putovaly pouze jednou. Principem je, že se vytvoří mutlicastová adresa z rozsahu `224.0.0.0/4` (pro LAN je určeno 224.0.0.0/24), a klienti se k této adrese zaregistrují. Pro vytváření skupina a registrování se používá protokol ***Internet Group Management Protocol*** (IGMP).






Autor: **Petr Bouška**

**Související články:**

### Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

- OSI model [05.03.2007 08:09]
- VLAN - Virtual Local Area Network [02.06.2007 15:54]
- Počítačové sítě a jejich typy [09.07.2007 15:32]
- Počítačové sítě - základní topologie [30.07.2007 14:47]
- Ethernet - CSMA/CD, kolizní doména, duplex [03.08.2007 12:40]
- TCP/IP - model, encapsulace, paket vs. rámec [16.08.2007 16:43]
- TCP/IP - metody vysílání dat [02.09.2007 15:05] ..... **právě čtete**
- TCP/IP - adresy, masky, subnety a výpočty [05.09.2007 14:53]
- TCP/IP - navázání a ukončení spojení [13.09.2007 16:10]
- TCP/IP a ethernet - cesta v síti, aktivní síťové prvky [17.09.2007 10:22]
- TCP/IP - Routing - směrování [21.09.2007 15:34]
- TCP/IP - nalezení MAC adresy k IP - ARP [25.09.2007 17:35]
- Počítačové sítě - Computer Networks [30.09.2007 13:28]
- DNS (Domain Name System) zaměřeno na Microsoft [26.11.2007 14:26]
- Diagramy a schémata počítačové sítě (Visio) [13.01.2008 18:02]
- Wake on LAN - lokální i vzdálený subnet [10.08.2008 20:42]
- TCP/IP - Internet Protocol Version 6 - IPv6 [05.03.2009 15:41]
- TCP/IP - skupinové vysílání IP Multicast a Cisco [10.03.2009 20:05]
- Běžné útoky na switche, Cisco Dynamic ARP Inspection [18.06.2009 10:15]
- Cisco - Router Switching metody a související termíny - CAM, FIB, CEF [28.06.2009 17:15]
- Výstavba moderní sítě [28.04.2010 14:41]
- Microsoft Network Load Balancing (NLB) a Cisco switche [09.08.2010 18:26]
- Cisco Network Design - Enterprise Architecture [24.09.2010 13:47]
- VPN 1 - IPsec VPN a Cisco [10.04.2011 19:59]

# TCP/IP - adresy, masky, subnety a výpočty

**Upraveno** 11.08.2008 10:00 | vytvořeno 05.09.2007 14:53 | **Samuraj - Petr Bouška** | ***sítě***

Sedmá část seriálu o počítačových sítích je již více zajímavá a přináší praktické informace. Na začátku je popis základních termínů pro sítě a podsítě, IP adresy a masky. Dále se probírají různé síťové třídy a způsoby zápisu subnetů. Druhá část se věnuje praktickým výpočtům síťových rozsahů, masek sítí, počtu hostů a subnetů.



*Pro vyzkoušení výpočtů a informací popsaných v tomto článku můžete zkusit moji webovou aplikaci **Online IP Subnet Calculator** (v AJ), která pro zadanou IP adresu a masku vypočítá a nalezne řadu různých informací. Také se na této adrese nachází obdoba tohoto článku v anglickém jazyce, kterou jsem ale napsal po delší době, takže některé věci popisuje trochu jinak (doporučuji k nahlédnutí).*

*Pokud potřebujete zjistit vaši veřejnou IP adresu, přes kterou jste připojeni do internetu, tak můžete využít jednoduchou webovou stránku **What is my public IP address?** (v AJ).*

*Článek byl doplněn 11.8. 2008.*

## Co je network a subnet (síť a podsíť)

### Počítačová síť - computer network

***Počítačová síť*** vznikne ve chvíli, kdy dva (někdy se říká minimálně tři) nebo více počítačů propojíme dohromady pomocí telekomunikačního systému za účelem sdílení zdrojů. V praxi je dnes nejrozšířenější síť založena na technologii ***ethernetu*** a používá protokol ***TCP/IP***. Reprezentantem jedné velké sítě je *internet*. Sítě se dále dělí podle řady parametrů na LAN, WAN, WLAN, MAN apod., ale pro tento popis to není důležité.

### Podsíť - subnet

V praxi není možné komunikovat v celé síti (například internetu) přímo (z řady důvodů fyzických i logických). Proto se síť dělí na ***podsítě - subnets - subnetworks***. Subnety slouží k logickému dělení sítě do menších hierarchických částí. Příkladem je, že velký ISP má určitý síťový rozsah (subnet), ten dělí na části, které přiděluje firmám a ve firmě se ještě dělí na menší části. Ke spojování jednotlivých subnetů slouží routery. Dělení sítě na subnety je důležité nejen proto, že naši síť "fyzicky" oddělíme od jiných sítí, ale také z výkonových důvodů. Řada informací se v rámci lokální sítě (subnetu) šíří pomocí ***broadcastů***, tedy vysílání všem zařízením, což je značná režie pro síť i zařízení. Proto se využívají například VLANy.

***Pozn.:*** *V praxi se často používá pojem síť - net - network, i když mluvíme o subnetu. Vychází to z termínu LAN - lokální síť, který označuje síť pokrývající malou geografickou oblast, jako třeba budovu, kancelář, apod. LAN je většinou podmnožinou větší sítě a zkráceně se používá pouze pojem síť.*

### IP adresa - IP address

**IP adresa** je logická adresa zařízení v síti IP. Skládá se ze 4 částí zvaných ***octety***, každá část je veliká 8 bitů, a zapisuje se oddělená tečkou. Adresa se většinou zapisuje v dekadické formě, ale pro výpočet je jasnější binární zápis. Teoreticky je tedy adresní rozsah od `0.0.0.0` do `255.255.255.255`. Příkladem IP adresy je třeba `68.12.5.10`.

Pro práci s adresami je třeba znát základy binární matematiky. ***Binární soustava*** má základ 2 a zapisuje se pomocí 0 a 1. Hodnota jedničky v dekadické soustavě je podle pozice (váhy), kde se nachází. Pro převod binárního čísla na dekadické, převedeme jedničky na desítkové hodnoty a tyto sečteme. Osmi bitová binární číslice může nabývat dekadických hodnot od 0 do 255, tedy celkem 256 hodnot.

| pozice | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| výpočet | $2^7$ | $2^6$ | $2^5$ | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ | |
| hodnota | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 | součet je 255 |

**Broadcast IP adresa** je adresa, na kterou se posílá komunikace v případě brodcastového vysílání. Jedná se o adresu, která má binárně samé jedničky, je to tedy IP adresa `255.255.255.255`. Tato adresa určuje všechny klienty v síti.

### Maska podsítě - Subnet mask

***Subnet mask*** nám pomáhá určit rozdělení sítě na podsítě. Určuje, která část IP adresy je síťová, a která pro hosty. Zápis je stejný jako u IP adresy, ale platné hodnoty jsou pouze ty, které mají v binárním tvaru zleva jedničky a zprava nuly (pokud se zleva na některé pozici objeví nula, dále již musí následovat pouze nuly). Jedničky v masce jsou tzv. ***network ID*** a je to část, která je pro daný subnet stále stejná. Nuly jsou tzv. ***host ID*** a tedy část, která je proměnná a určuje adresu hosta v daném subnetu. Příkladem jednoduché masky je `255.255.255.0`, ta určuje, že prvních 24 bitů adresy je network ID a posledních 8 bitů je hostovská část.

Možné hodnoty jednotlivých octetů pro masku:

| binárně | dekadicky |
|---|---|
| 00000000 | 0 |
| 10000000 | 128 |
| 11000000 | 192 |

| | |
|---|---|
| 11100000 | 224 |
| 11110000 | 240 |
| 11111000 | 248 |
| 11111100 | 252 |
| 11111110 | 254 |
| 11111111 | 255 |

> ***Pozn.:*** *Poslední octet nemůže nabývat hodnoty 254, zde totiž neexistuje ani jeden host (viz. dále). Pokud má hodnotu 255, znamená to, že se jedná o unicast, tedy jedinou adresu.*

## Zkrácený zápis masky - Classless Inter-Domain Routing (CIDR) notace

***Subnet mask*** se může zapisovat také ve zkrácené formě, které se říká **CIDR notace**. Ta se zapisuje jako IP adresa následovaná lomítkem (/) a číslem, které reprezentuje počet jedničkových bitů v masce podsítě v binární formě. Protože celkový počet bitů v masce je 32, tak počet nul je `32 - počet jedniček`. Příklad CIDR notace je `10.0.5.2/20` a tedy maska je `255.255.240.0`.

| dekadicky | 255 . | 255 . | 240 . | 0 | |
|---|---|---|---|---|---|
| binárně | 11111111 | 11111111 | 11110000 | 00000000 | |
| počet jedniček | 8 | 8 | 4 | 0 | = 20 |

## Wildcard-Mask

***Wildcard mask*** nebo také ***inversní maska*** je speciální zápis síťové masky, který používá například Cisco u Access listů. Jedná se o opak ke klasické masce, počítají se zde nuly místo jedniček. Takže například ke klasické masce `255.255.255.240` je inverzní maska `0.0.0.15`. Inverzní masku dostaneme tak, že normální masku zobrazíme binárně, provedeme inverzi a převedeme na dekadickou hodnotu. Nebo jednodušeji stačí, u každého octetu spočítat `255 - hodnota`. Tedy v našem příkladě 255-255=0, 255-240=15.

## Supernet

Občas se také používá termín ***supernet***, který označuje skupinu několika sdružených po sobě jdoucích ***subnetů***. Jedná se vlastně o technologii CIDR a používá se například pro agregaci routovacích záznamů.

***Příklad:*** Máme dva subnety `192.168.0.0/24` a `192.168.1.0/24`, z nich můžeme vytvořit jeden supernet `192.168.0.0/23`.

# Síťové třídy - classes

## Classful network - adresování s třídami

Když vznikl protokol IPv4, byla představa, že adresní prostor je ohromný a nikdy se nemůže vyčerpat. Tento adresní prostor byl rozdělen do základních pěti tříd, kdy se zařazení do patřičné třídy určovalo podle prvních bitů adresy. Při komunikace se nepoužívaly masky podsítě, protože ty byly napevno dány adresou.

| třída | určující bity | rozsah adres | maska | CIDR maska |
|---|---|---|---|---|
| class A | 0 | 0 - 127.x.x.x | 255.0.0.0 | /8 |
| class B | 10 | 128 - 191.x.x.x | 255.255.0.0 | /16 |
| class C | 110 | 192 - 223.x.x.x | 255.255.255.0 | /24 |
| class D | 1110 | 224 - 239.x.x.x | 255.255.255.255 | /32 |
| class E | 1111 | 240 - 255.x.x.x | rezervováno | |

> ***Pozn.:*** *Třída D je určena pro multicastové adresy.*

## Classless network - adresy bez tříd

Od classful network se již před dlouhou dobou ustoupilo a začalo se používat adresování CIDR, které je více flexibilní při dělení sítě na podsítě. V komunikaci používáme vždy IP adresu spolu s maskou. I když se opustili classful network, tak se v praxi běžně setkáme s označováním subnetů jako třída C apod., myslí se tím však typ masky.

U Cisco switchů a routerů se používá příkaz pro použití classless network, který je defaultně zapnutý

```
Switch#ip classless
```

CIDR používá ***Variable Length Subnet Masking*** (VLSM) pro rozdělování IP adres na subnety. Délká masky nemusí být stejná, aby se dosáhlo lepšího využití IP rozsahu. Můžeme například použít dohro

mady subnety `10.0.0.0/26` a `10.0.0.64/28`.

# Neveřejné síťové rozsahy (private subnets)

Některé síťové rozsahy mají speciální vlastnosti, tou hlavní je, že se neroutují, tzn. neprochází do dalšího subnetu. To se využívá u privátních subnetů, které neprochází do internetu. V praxi je využívá většina firem v lokální síti a do internetu přistupují přes veřejnou adresu za pomoci ***NATu***.

| síť | adresa sítě | broadcast adresa | adresy hostů |
|---|---|---|---|
| 10.0.0.0/8 | 10.0.0.0 | 10.255.255.255 | 10.0.0.1 - 10.255.255.254 |
| 192.168.0.0/16 | 192.168.0.0 | 192.168.255.255 | 192.168.0.1 - 192.168.255.254 |
| 172.16.0.0/12 | 172.16.0.0 | 172.31.255.255 | 172.16.0.1 - 172.31.255.254 |

Dalšími speciálními subnety je ***Localhost Loopback Address***, což jsou adresy, které by měli být lokální pro dané zařízení (nepřenáší se nikam do sítě). A ***Zeroconf Address***, kterou používá Microsoft a slouží k automatické konfiguraci sítě pro propojení pár počítačů.

| síť | adresa sítě | broadcast adresa | jméno |
|---|---|---|---|
| 127.0.0.0/8 | 127.0.0.0 | 127.255.255.255 | Localhost Loopback Addresses |

| 169.254.0.0 /16 | 169.254.0.0 | 169.254.255.255 | Zeroconf Address |
|---|---|---|---|

## Podsítě - Subnets a výpočty adres

**K**aždý subnet obsahuje ***základní adresu podsítě*** (base address nebo network address), což je první adresa rozsahu, dále ***adresy hostů*** a jako poslední ***broadcastovou adresu subnetu*** (subnet broadcast address), to je poslední adresa rozsahu. Podsíť je identifikována adresou sítě (teoreticky i jakoukoliv adresou hosta) a síťovou maskou, která se často zapisuje v CIDR notaci.

**N**apříklad podsíť `192.168.0.0/28` vypadá následovně

| 192.168.0.0 | adresa sítě |
|---|---|
| 192.168.0.1 | adresa hosta |
| ... | adresy hostů |
| 192.168.0.14 | adresa hosta |
| 192.168.0.15 | broadcast adresa |

**A**dresa je v síti lokální, pokud má stejné network ID. Tedy, když chceme určit, jestli dvě adresy patří do stejného subnetu, převedeme všechny hodnoty do binárního tvaru a ta část, kde jsou v masce jedničky, musí být u obou adres shodná.

**Subnet broadcast address** (broadcast adresa subnetu), poslední adresa subnetu (která se nemůže přiřadit stanici), je IP adresa, která se skládá z network ID a samých jedniček v host ID. Slouží k zaslání cíleného broadcastu na daný subnet.

### Výpočet počtu subnetů a hostů

**Z** masky sítě určíme, kolik můžeme vytvořit subnetů a kolik tyto subnety budou mít hostů. Maska nám vlastně určuje, že ta část, kde má jedničky musí být pro podsíť stejná a ta část, kde jsou nuly, se může měnit (nabývat 0 a 1). Takže hosti mohou mít jako adresu všechna čísla, která můžeme vytvořit z těchto bitů. Těchto adres můžeme vytvořit $1+2^0+2^1+2^2+.+2^{n-1}=2^n$, kde $n$ je počet bitů. Jednička na začátku je proto, že počítáme i nulovou hodnotu.

**P**ři výpočtu postupujeme následovně. Vezmeme octet masky, v kterém je přechod mezi jedničkami a nulami. Podle počtu jedniček v tomto octetu a celkového počtu nul spočítáme počet podsítí (z jedniček) a počet hostů (z nul).

> $2^{\text{počet jedniček}}$ **= počet subnetů** (nově)

**T**akto je to, alespoň dnes v praxi a podle RFC 1812. V teoretických případech však musíme vycházet ze staršího RFC 950, které nepovolovalo adresy podsítí se samými jedničkami nebo nulami. Proto musíme odečíst 2 podsítě:

> $2^{\text{počet jedniček}}$ **- 2 = počet subnetů** (postaru)

**U** Cisco zařízení se použití nového RFC zapíná pomocí (defaultně zapnuto)

```
Switch#ip subnet zero
```

**U** počtu hostů musíme vždy odečíst první a poslední adresu (base a broadcast address), které se nemohou využít jako adresa pro hosta:

> $2^{\text{počet nul}}$ **- 2 = počet hostů**

***Příklad:***

**M**áme adresu `10.0.5.2/20`

| maska binárně | 11111111 | 11111111 | 11110000 | 00000000 |
|---|---|---|---|---|
| maska dekadicky | 255 | 255 | 240 | 0 |

**P**oužijeme třetí octet, kde jsou 4 jedničky a 4 nuly (plus 8 nul ve čtvrtém octetu).

**Hostů** v každém subnetu může být $2^{12} - 2 = 4094$.

**V** rámci třetího octetu můžeme vytvořit $2^4 = 16$ **podsítí.**

### Výpočet základní adresy sítě

**P**okud máme adresu hosta a masku podsítě, můžeme jednoduše spočítat základní adresu podsítě. Pokud vezmeme binárně adresu hosta a masku sítě a provedeme bitový součin (AND), dostaneme adresu sítě. Příklad pro `10.217.123.7/20`

**IP adresa**

| dekadicky | 10 | 217 | 123 | 7 |
|---|---|---|---|---|
| binárně | 00001010 | 11011001 | 01111011 | 00000111 |

**Maska podsítě**

| dekadicky | 255 | 255 | 240 | 0 |
|---|---|---|---|---|
| binárně | 11111111 | 11111111 | 11110000 | 00000000 |

**Provedeme bitový AND a dostaneme adresu sítě**

| binárně | 00001010 | 11011001 | 01110000 | 00000000 |
|---|---|---|---|---|
| dekadicky | 10 | 217 | 112 | 0 |

### Výpočet broadcast adresy subnetu

**B**roadcast adresu subnetu, v kterém se nachází klient, spočítáme jednoduše. V IP adrese hosta změníme bity v hostovské části na 1. Matematicky řečeno vezmeme IP adresu a provedeme bitový součet (OR) s negovanou (NOT)

maskou podsítě. Pro předchozí příklad (síť 10.217.123.7/20), kde prvních 20 bitů je network ID a zbylých 12 je hostovská část, to tedy je:

**IP adresa**

| | | | | |
|---|---|---|---|---|
| dekadicky | 10 | 217 | 123 | 7 |
| binárně | 00001010 | 11011001 | 01111011 | 00000111 |

**Invertovaná (bitová negace) maska**

| | | | | |
|---|---|---|---|---|
| binárně | 00000000 | 00000000 | 00001111 | 11111111 |
| dekadicky | 0 | 0 | 15 | 255 |

**Provedeme bitové OR a máme broadcast adresu subnetu**

| | | | | |
|---|---|---|---|---|
| binárně | 00001010 | 11011001 | 01111111 | 11111111 |
| dekadicky | 10 | 217 | 127 | 255 |



Autor: Petr Bouška

**Související články:**

## Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

# VLAN - Virtual Local Area Network

Sobota, 02.06.2007 15:54 | Samuraj - Petr Bouška | sítě

Osmá část seriálu o počítačových sítích. VLAN neboli virtuální lokální síť je v dnešní době běžná technologie, která přináší řadu výhod. Myslím, že všechny střední a větší firmy technologii VLAN používá a i pro malé firmy může být zajímavá. VLANy slouží k logickému členění sítě bez vazby na členění fyzické. V článku se pokouším popsat vše potřebné k pochopení toho, co to VLAN je, jaké jsou výhody a způsoby nasazení.



*O vytváření a konfiguraci VLAN na Cisco zařízeních, stejně jako o Dynamic Trunk Protocolu a VLAN Trunking Protocolu si můžete přečíst v článku **Cisco IOS 7 - konfigurace VLAN, VTP**.*

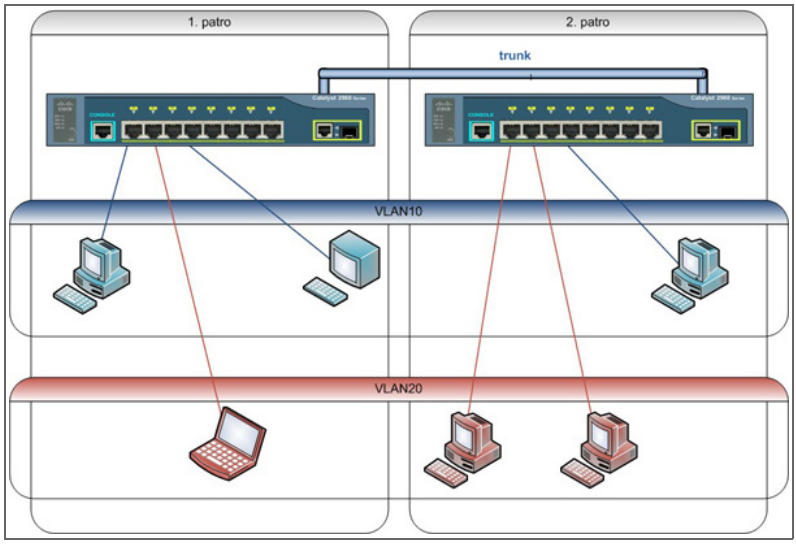
## Co je to VLAN

**Virtuální LAN** slouží k logickému rozdělení sítě nezávisle na fyzickém uspořádání. Můžeme tedy naši síť segmentovat na menší sítě uvnitř fyzické struktury původní sítě. Druhým důležitým pojmem, který bude více vysvětlen později, je **trunk**. Jako ***trunk*** označujeme port, který je zařazen do více VLAN.

Jednoduše řečeno pomocí VLAN můžeme dosáhnout stejného efektu, jako když máme skupinu zařízení připojených do jednoho (několika propojených) switche a druhou skupinu do jiného (jiných) switche. Jsou to dvě nezávislé sítě, které spolu nemohou komunikovat (jsou fyzicky odděleny). Pomocí VLAN můžeme takovéto dvě sítě vytvořit na jednom (nebo několika propojených) switchi.

V praxi samozřejmě často potřebujeme komunikaci mezi těmito sítěmi. S VLAN můžeme pracovat stejně jako s normálními sítěmi. Tedy použít mezi nimi jakýkoliv způsob routování. Často se dnes využívá L3 switch (switch, který funguje na třetí vrstvě OSI) pro inter-VLAN routing - směrování mezi VLAN.

Níže uvádím klasický obrázek, který se používá pro vysvětlení VLAN. Máme dvě patra, na každém patře je switch, switche jsou propojeny páteří s trunkem. Chceme propojit zařízení do dvou nezávislých skupin (modrá - VLAN10 a červená VLAN20). Pomocí VLAN je to takto jednoduché. Tradiční technikou bychom museli mít switche oddělené a každou skupinu (modrou a červenou) propojit do jednoho switche, což by byl problém, protože jsou na různých patrech.




Pro podrobné pochopení VLAN je třeba rozumět základům sítí a jejich segmentování (dělení na subnety - podsítě).

## Počítačová síť - WAN, LAN

***Počítačová síť*** vznikne ve chvíli, kdy dva (někdy se říká minimálně tři) nebo více počítačů propojíme dohromady pomocí telekomunikačního systému za účelem sdílení zdrojů. Sítě se dále dělí podle řady parametrů na LAN, WAN, WLAN, MAN apod. V tuto chvíli nás zajímá ***lokální počítačová síť*** - LAN (Local Area Network), která se vyznačuje tím, že počítače jsou propojeny na menším geografickém území (tedy v rámci firmy, budovy, místnosti, atp.). Pro **LAN** se nejčastěji používá technologie ***Ethernet*** s protokolem ***TCP/IP*** a pro **WAN** (Wide Area Network - propojení jednotlivých LAN) technologie ***Frame Relay***.

## Podsíť - subnet

**TCP/IP protokol** používá pro adresování zařízení ***IP adresy***. Těchto adres je určitý rozsah, který se pro praktické použití (směrování, přidělování adres organizacím, broadcasty) dělí na menší hierarchické části, kterým se říká **subnety** (podsítě).

Zařízení mohou přímo komunikovat pouze s dalšími zařízeními, která jsou ve stejném subnetu. Se zařízeními z jiných subnetů komunikují typicky přes jednu adresu - gateway (bránu), která provádí routování.

## Oddělení sítí

Jak jsem uvedl výše, pokud použijeme různé subnety, tak spolu zařízení nemohou komunikovat. Není to však úplně pravda, rozhodně nedojde k oddělení těchto zařízení. Pokud jsou totiž připojena na stejné médium, propojena do

stejného hubu (pracuje na 1. vrstvě OSI) nebo switche (pracuje na 2. vrstvě OSI). Tak komunikace dorazí z jednoho zařízení na druhé, i když jsou v jiném subnetu. Zařízení však bude tuto komunikaci ignorovat. Je to proto, že hub (posílá všude) ani switch (používá MAC adresy) se nedívá na IP adresy procházející komunikace. Proto se dá tato komunikace zachytávat a odposlouchávat. Pokud tedy chceme mít oddělené sítě, tak musíme použít oddělené switche.

Kdežto použitím VLAN dojde k tomu, že komunikace se posílá pouze na porty, které jsou zařazeny do stejné VLANy. Záleží tedy sice na softwaru switche, ale dá se říct, že se jedná o fyzické oddělení. Existují nějaké metody útoky na VLAN (proniknutí do jiné VLAN), ale při dobře nastavené síti by mělo být vše bezpečné.

### Subnety a VLANy

Z výše uvedeného také plyne to, že pro různé VLANy bychom měli používat různé subnety. Pokud chceme mezi těmito VLANami routovat, tak je to nutné, stejně jako v případě, kdy chceme využít některé speciální funkce na switchi.

## Proč vznikly VLANy

Technologie VLAN začala vznikat kolem roku 1995, ale zprvu se jednalo o různá proprietární řešení. V praxi se však více rozšířili až před několika lety a to hlavně ve středních a velkých firmách, přestože již delší dobu existuje standard.

Hlavní důvody proč vznikly VLAN byly asi tyto:

- ***seskupování uživatelů*** v síti podle skupin či oddělení nebo podle služeb místo podle fyzického umístění a oddělení komunikace mezi těmito skupinami
- ***snížení broadcastů*** v síti, které začaly být problémem již před několika lety
- ***zmenšení kolizních domén*** v době, kdy se nepoužívaly switche, ale třeba huby

Idea pro logické seskupování uživatelů, která se uvádí v řadě materiálů, a tedy vytváření VLAN je

- ***podle organizační struktury*** - pokud je většina komunikace v rámci oddělení, kde jsou vlastní tiskárny, file servery, atd. a mezi jednotlivými odděleními není komunikace, pouze pár služeb (mail) je společných pro všechny
- ***podle služeb*** - do VLAN se seskupují pracovníci, kteří využívají stejné služby (účetnictví, DB, atd.)

Původní důvody již dnes často nejsou aktuální nebo se z praxe změnily názory. Jak se používají VLANy dnes, jinak řečeno, jaký je jejich hlavní přínos, uvádím v další kapitole.

## Jaké jsou praktické výhody VLAN

- ***snížení broadcastů*** - hlavní výhodou VLAN je vytvoření více, ale menších, broadcastových domén. Tedy zlepšení výkonu sítě snížením provozu (traffic).
- ***zjednodušená správa*** - k přesunu zařízení do jiné sítě stačí překonfigurovat zařazení do VLANy, tedy správce konfiguruje SW (zařazení do VLAN) a ne HW (fyzické přepojení)
- ***zvýšení zabezpečení*** - oddělení komunikace do speciální VLANy, kam není jiný přístup. Toho se dá samozřejmě dosáhnout použitím samostatných switchů, ale často se toto uvádí jako bonus VLAN.
- ***oddělení speciálního provozu*** - dnes se používá řada provozu, který nemusí být propojen do celé sítě, ale přesto jej potřebujeme dostat na různá místa, navíc nechceme, aby nám ovlivňoval běžný provoz. Příkladem je například IP telefonie, komunikace mezi AP v centrálně řízeném prostředí, management (zabezpečení správcovského přístupu k zařízením). Například pro IP telefonii, kde je použití VLAN naprosto běžné, nám stačí jediná zásuvka, kam přivedeme VLAN pro telefonii i VLAN s přístupem do sítě a v telefonu se komunikace rozdělí. Navíc VLANy můžeme použít spolu s QoSem pro zaručení kvality komunikace (obsazení pásma).
- ***snížení HW*** - samozřejmě se nám nesnižuje potřebný počet portů (až na speciální případy jako IP telefonie), ale tím, že mohou být různé podsítě na stejném switchi, jej můžeme lépe využít (například pro propojení tří zařízení nepotřebujeme speciální switch, který má minimálně 8 portů).

## Jak se zařazuje komunikace do VLAN

Přiřazení do VLANy se nastavuje typicky na **switchi** (pouze v některých speciálních případech přichází označená komunikace přes trunk z jiného zařízení). Na switchích, které podporují VLANy, vždy existuje alespoň jedna VLAN. Jedná se o defaultní ***VLAN číslo 1***, kterou není možno smazat či vypnout. Pokud nenastavíme jinak, tak jsou všechny porty (tedy veškerá komunikace) zařazeny do VLAN 1.

Pro zařazení komunikace do VLANy existují čtyři základní metody, ale v praxi je nejvíce využívána možnost první - zařazení dle portu.

### 1. podle portu

***Port switche je ručně*** a napevno zařazen (nakonfigurován) do určité ***VLANy***. Veškerá komunikace, která přichází přes tento port, spadá do zadané VLANy. To znamená, že pokud do portu připojíme další switch, tak všechny zařízení připojená k němu budou v jedné VLANě. Jedná se o nejrychlejší a nejpoužívanější řešení. Není třeba nic vyhodnocovat pro zařazení do VLAN. Definice zařazení do VLAN je lokální na každém switchi. Jednoduše se spravuje a je přehledné.

### 2. podle MAC adresy

Rámce(port) se zařadí do VLANy podle ***zdrojové MAC adresy***. Musíme tedy spravovat tabulku se seznamem MAC adres pro každé zařízení spolu s VLANou. Výhodou je, že se jedná o dynamické zařazení, takže pokud přepojíme zařízení do jiného portu, automaticky se zařadí do správné VLANy. Switch musí vyhledávat v tabulce MAC adres.

Jsou zde dvě možnosti, jak tato metoda může fungovat. Buď se podle MAC adresy prvního rámce nastaví zařazení portu do VLANy a toto nastavení zůstane, dokud se port nevypne. Nebo se každý rámec zařazuje samostatně do VLANy podle MAC adresy. Toto řešení je velmi náročné na výkon.

Cisco má řešení zvané ***VLAN Membership Policy Server*** (VMPS), pro které je třeba speciální server, který spravuje tabulky MAC adres. Navíc se při této metodě zařazuje port do VLANy, takže pokud je do něj připojeno více zařízení (max. 20), musí být všechny ve stejné VLAN.

### 3. podle protokolu = podle informace z 3. vrstvy

Tato metoda určuje zařazení podle protokolu přenášeného paketu. Například oddělíme IP provoz od AppleTalk. Nebo zařazujeme podle IP adresy či rozsahu. V praxi není příliš rozšířené. Zařízení musí mít napevno definovanou IP adresu a switch se musí dívat do třetí vrstvy (normálně funguje na druhé), znamená to zpomalení.

### 4. podle autentizace

Ověří se uživatel nebo zařízení pomocí protokolu **IEEE 802.1x** a podle informací se automaticky umístí do VLANy. Je to primárně bezpečnostní metoda, které řídí přístup do sítě (NAC), ale po rozšíření slouží i pro VLANy. Je to zajímavá metoda proto, že je velmi univerzální. Nezáleží ani na fyzickém zařízení ani na místě zapojení.

RADIUS server, který ověřuje identitu uživatele, obsahuje také mapovaní uživatelů na VLANy a tuto informaci zašle po úspěšné autentizaci. U této metody je možné nastavení, že v případě, kdy není uživatel autentizován, tak je zařazen do speciální hostovské VLANy.

U Cisco switchů může být port single-host, kdy je možno připojit pouze jedno zařízení nebo multiple-host, kdy sice může být do portu připojeno více zařízení, ale ve chvíli, kdy se první autentizuje, tak je port autentizovaný (a zařazený do VLANy) a komunikovat mohou všechna zařízení.

## Jak funguje komunikace v rámci VLAN

V praxi máme dvě situace, kdy se při komunikaci řeší příslušnost k VLANě. Je to při komunikaci v rámci jednoho switche nebo při komunikaci mezi několika switchi.

### VLANy na jednom switchi

Při komunikaci ve VLANách v rámci ***jednoho switche*** je to jednoduché. Switch si v operační paměti udržuje informace, do které VLANy patří daná komunikace (port), a v rámci switche povoluje pouze správné směrování. V tomto případě máme jednotlivé porty zařazeny do jedné VLANy a to buď staticky, nebo dynamicky, jak bylo řečeno výše (možnosti 2,3,4). Cisco těmto portům říká ***access port*** (přístupový port).

### VLANY mezi více switchi

Složitější situace nastává, když chceme, aby se informace o ***zařazení do VLANy*** neztratila při přechodu na jiný switch, tedy abychom v celé naší síti mohli využít stejné VLANy a nezáleželo, do kterého switche je zařízení připojeno. Navíc chceme, aby tato metoda fungovala i mezi switchi různých výrobců. To byl ze začátku problém a používali se různé metody. Například, když zařazujeme komunikaci podle MAC adresy, tak můžeme tabulku přiřazení mít na všech switchích. Cisco vytvořilo svoji metodu ISL, která zapouzdřuje celý rámec, ale funguje pouze na Cisco zařízeních. Také můžeme propojit dva *access porty* na dvou switchích, zařadit je do stejné VLANy a přeneseme potřebné informace. To je ale velmi nepraktické.

Naštěstí vznikl standard **IEEE 802.1q**, který využívá značkování rámců. Označuje se komunikace jen ve chvíli, kdy je to třeba. Takže dokud probíhá v rámci jednoho switche a připojených zařízení, tak se nic nepřidává. Teprve, když chceme poslat komunikaci dalšímu switchi (či podobnému zařízení), tak ji označíme. Odchozí komunikace se taguje na portu, kterému se říká ***trunk port***. Tento port přenáší více (vybraných) VLAN a aby je mohl odlišit, tak je označuje. Spoji dvou trunk portů se říká **trunk** nebo ***trunk link***.

### IEEE 802.1q tagging

Protokolu **IEEE 802.1q** se říká také ***trunking protokol*** nebo ***dot1q tagging***. Jedná se o standardizovanou metodu, kterou podporují všechny moderní switche s podporou VLAN. Funguje na principu tzv. tagování. Vezmeme originální rámec, jeho hlavičku rozšíříme o 4B informací, z nichž první je značka, že se jedná o protokol 802.1q (hodnota 0x8100). Dále následuje priorita dle protokolu 802.1p, příznak, zda je MAC adresa v kanonickém tvaru a poslední je číslo VLANy.

Protože se změnila data, je třeba přepočítat kontrolní součet na konci rámce.

**Originální rámec**

| 6B | 6B | 2B | 64 až 1500B | 4B |
|---|---|---|---|---|
| cílová adresa (DA) | zdrojová adresa (SA) | typ nebo délka | data | kontrolní součet (FCS) |

**Upravený rámec pomocí 802.1q**

| 6B | 6B | 4B | 2B | 64 až 1500B | 4B |
|---|---|---|---|---|---|
| cílová adresa (DA) | zdrojová adresa (SA) | 802.1q tag | typ nebo délka | data | kontrolní součet (FCS) |

**Tvar 802.1q tagu**

| 2B | 3b | 1b | 12b |
|---|---|---|---|
| 0x8100 | priorita (802.1p) | Canonical Format Indicator (CFI) | VLAN ID |
| Tag Protocol ID (TPID) 2B | | Tag Control Information (TCI) 2B | |

**Native VLAN** je termín spojený s protokolem 802.1q. Nastavuje se na ***trunk portu***, u Cisco prvků musíme nativní VLANu vždy nastavit a to shodně na obou stranách trunku. Provoz, který je zařazen do native VLAN se při přenosu netaguje (zůstává nezměněn) a příchozí provoz, který není tagovaný se zařazuje do native VLAN. Často se jako native VLAN nastavuje management VLAN. Důsledkem také je, že pokud se na trunk port dostane nějaký rámec, který nemá tag, tak je zařazen do nativní VLANy. Jinak řečeno, pokud do portu, který je nakonfigurován jako trunk, připojíme normální stanici (která nepodporuje trunk), tak bude komunikovat v této VLANě. V praxi můžeme využít tuto vlastnost třeba u zapojení, kdy je připojen IP telefon a za ním PC. Native VLAN nastavíme stejně jako VLAN pro PC, potom pokud se odpojí telefon a PC se připojí přímo, tak jeho komunikace stále funguje.

### Cisco ISL encapsulation

**ISL** znamená ***Inter-Switch Link***, tedy spoj mezi switchi. Cisco vytvořilo svůj protokol ještě v době, kdy neexistoval standard. ISL má výhodu, že funguje nejen pro protokol IP, ale i pro jiné. Stejně jako 802.1q podporuje prioritizaci.

Bohužel se jedná o proprietární metodu, kterou používá pouze Cisco a v dnešní době ji nalezneme pouze na switchích vyšší řady (třeba Catalyst 3750). Princip funkce je takový, že se celý rámec zabalí (encapsulate) do nové hlavičky a kontrolního součtu. Z toho plyne nevýhoda, že se více zvětšuje komunikace, každý rámec je o 30B větší.

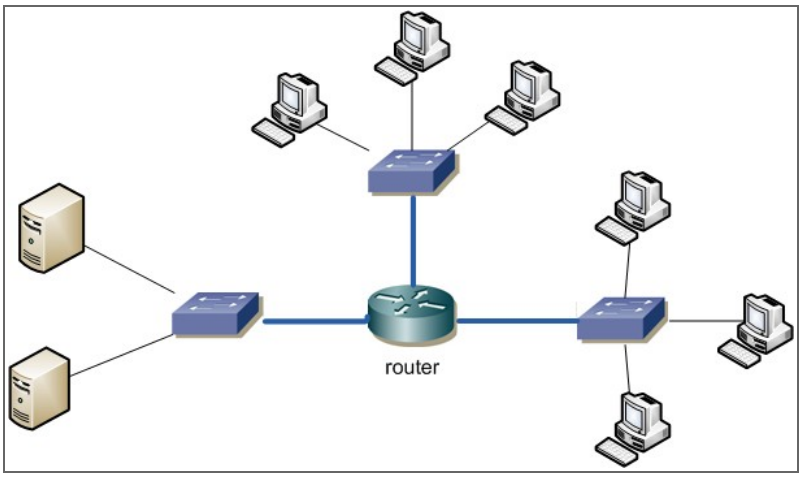
| 26B | | 4B |
|---|---|---|
| ISL header | encapsulation frame (originální rámec) | kontrolní součet (FCS) |

### Další využití trunků

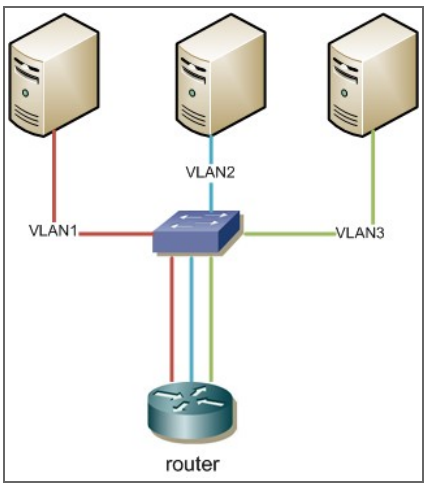
V dnešní době nemusíme používat ***trunk*** pouze pro propojení switchů, ale velice účinně jej využít i pro ***připojení serverů***, které potřebujeme mít připojené do více sítí. Operační systém ***Linux*** podporuje protokol ***802.1q*** již v jádře, pro ***Windows*** potřebujeme ovladač pro síťovou kartu s podporou VLAN (například Intel). Na síťové kartě pak nakonfigurujeme VLANy, které zde přichází, a pro každou se vytvoří virtuální síťové spojení, které můžeme používat jako běžnou síťovku. Ušetříme tím fyzické síťové karty na serveru, i když se snižuje propustnost. Použít to můžeme třeba na firewallu nebo hostovském serveru pro virtuály. Na druhou stranu je třeba se vždy zamyslet, zda je toto to správné řešení a neexistuje něco jednodušššího (například, abychom DHCP server nepřipojovali do každé routované podsítě, když můžeme využít DHCP relay agenta).

## Routing mezi VLANy

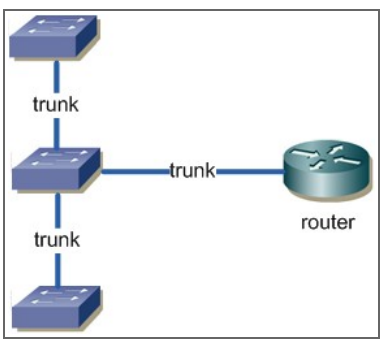
Tradiční routing vypadá tak, že máme několik oddělených sítí a chceme mezi nimi umožnit nějakou komunikaci. Následující obrázek představuje tři samostatné switche propojené pomocí routeru. Může se jednat například o firemní servery a dvě oddělení, které k nim chceme připojit, ale nechceme, aby mohla komunikovat mezi sebou.

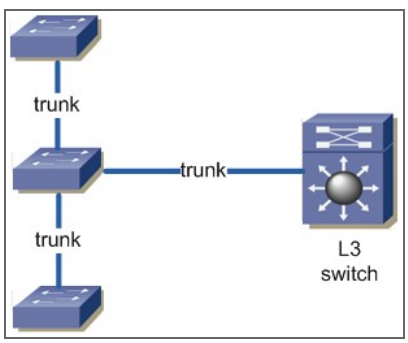



Pokud využíváme VLANy, můžeme se k nim chovat stejně jako k normálním podsítím. Na switchi můžeme jednotlivé VLANy, které chceme routovat, vyvést do samostatných access portů a ty připojit k routeru.




To je však zbytečné plýtvání a výhodnější je použít mezi routrem a switchem trunk. Také můžeme místo klasického routeru využít L3 switch, který je rychlejší.




U Cisco L3 switchů provedeme inter-VLAN routing jednoduše tak, že zapneme routování a na ty VLANy, které mají mezi sebou routovat, nastavíme IP adresu.



Autor: Petr Bouška

**Související články:**

## Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

# TCP/IP - navázání a ukončení spojení

Čtvrtek, 13.09.2007 16:10 | Samuraj - Petr Bouška | sítě

Devátá část seriálu o počítačových sítích je opět stručná a krátká. Je zde krátce popsán způsob, jakým TCP protokol navazuje a ukončuje spojení.

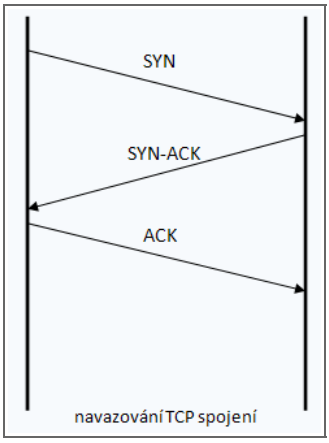


## Navázání spojení v TCP

Aby se mohla vysílat data pomocí TCP protokolu, je nejprve třeba vytvořit spojení. Pro navázání spojení se používá ***třícestný handshake*** (potřesení ruky). V průběhu navazování spojení se obě strany dohodnou na čísle sekvence (sequence number). Číslo sekvence a odpovědi (sequence, acknowledgement number) jsou 32bitové hodnoty uváděné v TCP hlavičce. Pro navázání spojení se posílá TCP segment, který má nastaveny příznaky (flags) v TCP hlavičce. Jedná se o 8 bitových hodnot `CWR` (Congestion Window Reduced), `ECE` (ECN-Echo), `URG` (Urgent), `ACK` (Acknowledgement), `PSH` (Push), `RST` (Reset), `SYN` (Synchronize), `FIN`.

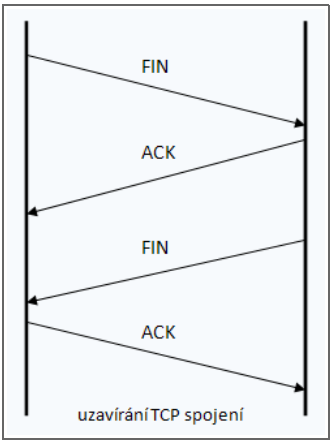
Navázání spojení probíhá ve třech krocích:

1. klient pošle `SYN` packet s uvedeným číslem sekvence (x), číslo odpovědi 0
2. druhá strana si uloží číslo sekvence (x) a odpoví `SYN-ACK`, jako číslo sekvence nastaví svoje číslo (y) a do čísla odpovědi vloží (x+1) - další očekávanou hodnotu
3. klient odpoví `ACK`, číslo sekvence (x+1), číslo odpovědi (y+1)




## Ukončení spojení v TCP

Principy při ukončení spojení jsou podobné jako při jeho navazování. Nejčastěji se používá ***čtyřcestný handshake***, kdy každá strana samostatně uzavře spojení. Zde se používá sekvence `FIN` s odpovědí `ACK`.






Autor: Petr Bouška

**Související články:**

## Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

- OSI model [05.03.2007 08:09]
- VLAN - Virtual Local Area Network [02.06.2007 15:54]
- Počítačové sítě a jejich typy [09.07.2007 15:32]

# TCP/IP a ethernet - cesta v síti, aktivní síťové prvky

Pondělí, 17.09.2007 10:22 | Samuraj - Petr Bouška | sítě

V desáté části série o počítačových sítích se věnuji způsobům, jakými se doručují (a adresují) data (rámce) na síti. Beru v potaz ethernet (a tedy rámec a fyzické adresování) a TCP/IP (paket a logické adresování). V praxi se jedná buď o přímé doručení, nebo o doručení nepřímé. Vše také záleží na použitých aktivních (pasivních) prvcích, takže popisuji tři běžné příklady (sdílené médium, switch, router). V druhé části článku je stručný popis aktivních síťových prvků a principů jejich funkce.



## Adresy v rámci při průchodu sítí

### Stanice na sdíleném médiu

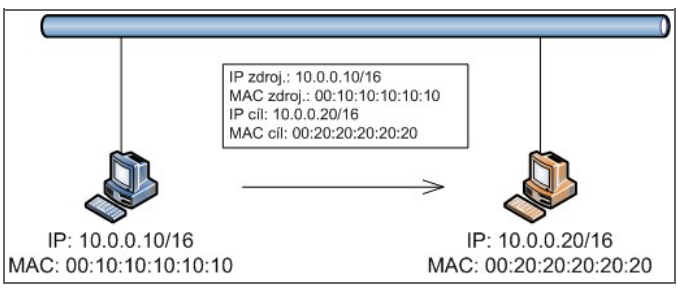
**M**áme dvě stanice připojené na sdíleném médiu, což může být například stará verze ***ethernetu*** v *topologii sběrnice* (bus) nebo běžně používaná *topologie hvězda* (star), když se jako centrální prvek použije hub. Obě stanice mají nastavenu IP adresu a určenu síť (zadánu masku subnetu) a síťové karty mají MAC adresu. Tento způsob se označuje jako ***sdílený ethernet***.

**V** tomto případě jsou obě stanice ve stejné síti (stejném subnetu), takže dojde k ***přímému doručení***.

**P**okud chce ***modrá stanice*** poslat data ***červené stanici***, tak nejprve vytvoří *ethernetový rámec*:

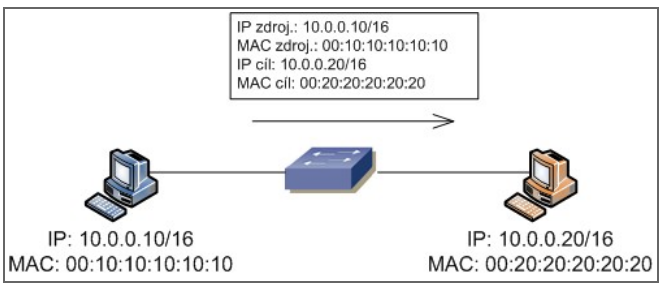
1. známe IP adresu cílové stanice
2. do hlavičky IP paketu se vloží naše IP adresa (zdrojová) a cílová IP adresa
3. nyní potřebujeme MAC adresu cílové stanice, buď ji máme v ARP tabulce, nebo provedeme ARP dotaz
4. do ethernetového rámce vložíme naši MAC adresu (zdrojovou) a cílovou MAC adresu

**N**ásledně se odešlou tato data do sítě (využije se CSMA/CD). Na sdíleném médiu projde rámec celou síť a každá stanice naslouchá, zda neobsahuje její MAC adresu. Pokud ano, tak rámec přijme.




### Propojení pomocí switche

**V** případě, kdy je ethernet propojen do topologie hvězda za pomoci switche, tak se ve většině případů dostane komunikace pouze ke stanicím, kterým je určena (pomocí CAM tabulky, tedy jaká MAC adresa komunikuje z kterého portu, pokud nemá záznam, tak odešle na všechny porty mimo příchozího). Princip komunikace je však stejný jako u sdíleného média a switch je pro komunikující stanice transparentní. Tento, dnes běžný, způsob se označuje jako ***přepínaný ethernet***.

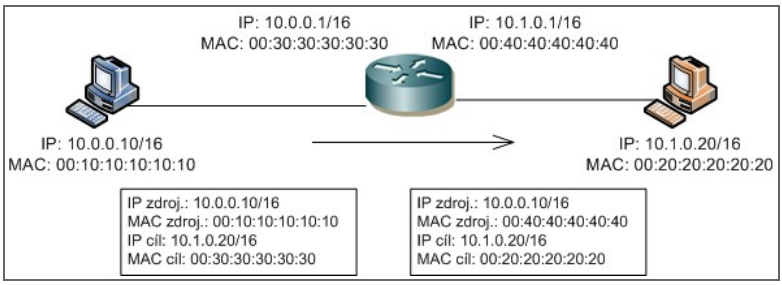


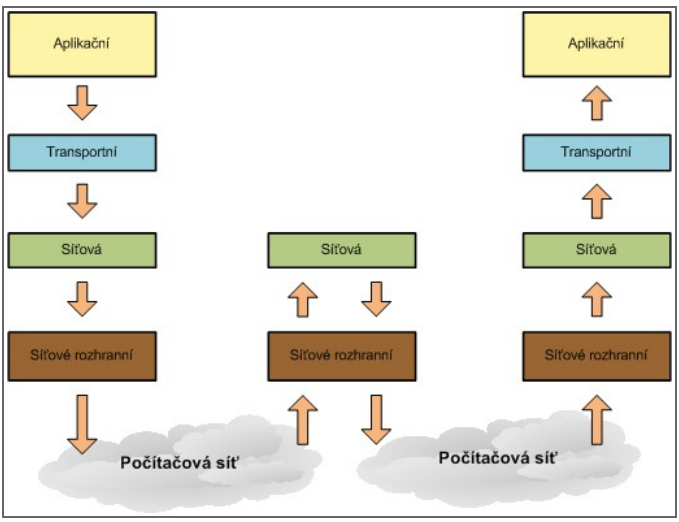

### Propojení pomocí routeru

**V** tomto případě je každá stanice v jiném subnetu, takže musí dojít k ***nepřímému doručení***. Router již není transparentní síťový prvek, ale ostatní síťová zařízení jej musí adresovat. Následující popis je obecný způsob komunikace:

1. na síťové vrstvě se vytvoří hlavička IP paketu, která obsahuje IP adresu zdrojové a cílové stanice
2. zdrojová stanice testuje, zda je cílová IP adresa ve stejném subnetu, tedy zda je pro ni lokální
3. dále pokračuje vrstva síťového rozhraní vytvářením hlavičky ethernetového rámce, vloží svoji zdrojovou MAC adresu, cílová MAC adresa se přiřadí k IP adrese z ARP keše (dotazu) a určí se podle:
   a. při lokální komunikaci adresy cílové stanice
   b. pokud není lokální, tak se podívá do routovací tabulky a použije adresu patřičného routeru (další hop, často se použije gateway)
4. rámec se odešle do sítě
5. data přijdou na router, ten podle MAC adresy pozná, že jsou určena jemu
6. zkontroluje paket a sníží TTL (doba života v síti - počet hopů)
7. znovu se provádí kontrola, zda je IP adresa lokální na jednom z interfaců a buď se odešle dalšímu routeru

nebo přímo cílové stanici
8. cílová stanice přijme rámec podle MAC adresy
9. ověří jej, zkontroluje IP adresu a postoupí jej skrze vrstvy nahoru




## Aktivní síťové prvky

***Počítačová síť*** se skládá z aktivních a pasivních síťových prvků. Mezi ***pasivní síťové prvky*** patří kabeláž a konektory. Mezi ***aktivní síťové prvky*** patří síťová karta (NIC), switch, router, firewall, apod.

### Repeater (opakovač)

- pracuje na první vrstvě OSI modelu (Layer 1)
- většinou má pouze dva porty
- slouží k prodloužení dosahu signálu, hlavně u sběrnicové topologie
- dnes se nepoužívá

#### Princip funkce

- data, která přijme na jednom portu, okamžitě přeposílá na druhý port

### Hub (rozbočovač)

- označuje se také jako multipoint repeater
- pracuje na první vrstvě OSI modelu (Layer 1)
- většinou má 4 - 24 portů
- základní prvek pro hvězdicovou topologii
- dnes se nepoužívá

#### Princip funkce

- data, která přijme na jednom portu, okamžitě přeposílá na všechny porty mimo portu odkud je přijal

### Bridge (most)

- pracuje na druhé vrstvě OSI modelu (Layer 2) - rozhoduje podle MAC adresy
- většinou má pouze dva porty
- slouží k propojení/oddělení segmentů
- snižuje velikost kolizní domény
- broadcasty a multicasty se posílají všude
- dnes se nepoužívá

#### Princip funkce

- u přicházejících rámců čte zdrojovou MAC adresu a vytváří v paměti tabulku MAC adres a portů, odkud pochází
- pokud nemá pro cílovou MAC adresu záznam, tak rámec odešle na všechny porty
- pokud záznam existuje a pokud leží příjemce ve stejném segmentu jako odesílatel, tak se data neposílají do ostatních segmentů

### Switch (přepínač)

- označuje se také jako multipoint bridge
- pracuje na druhé vrstvě OSI modelu (Layer 2) - rozhoduje podle MAC adresy
- většinou má 4 - 48 portů
- slouží k propojení/oddělení segmentů
- snižuje velikost kolizní domény
- broadcasty se posílají všude
- pracuje rychle
- základní prvek pro hvězdicovou topologii
- neupravuje rámec

**Princip funkce**

- u přicházejících rámců čte zdrojovou MAC adresu a vytváří v paměti tabulku MAC adres a portů, odkud pochází, tabulka se označuje jako CAM (Content Addressable Memory) tabulka
- pokud nemá pro cílovou MAC adresu záznam, tak rámec odešle na všechny porty mimo příchozího
- pokud má v tabulce cílovou MAC adresu, tak rámec pošle pouze na daný port

## Módy switche

Kvůli hledání kompromisu mezi zpožděním a spolehlivostí existuje několik metod.

- ***Cut-Through*** - rychlé, ale bez kontroly chyb, přeposílá rámce okamžitě, kdy zná cílovou MAC adresu
- ***Store-and-Forward*** - nejprve se celý rámec přijme, ověří se FCS (CRC) a pak teprve posílá nebo zahodí
- ***Fragment-Free*** (Modified Cut-Through) - kompromis, nejprve načte prvních 64 bytů (včetně hlavičky) a pak přeposílá

## Router (směrovač)

- pracuje na třetí vrstvě OSI modelu (Layer 3) - rozhoduje podle IP adresy
- hraniční router se občas označuje jako gateway (brána)
- slouží pro spojování sítí
- nabízí služby uvnitř LAN (směrování ze zdroje do cíle, segmentování sítě, ARP) a propojení do WAN (přes serial, ISDN, DSL, optiku)
- broadcasty se standardně nepřeposílají - snižuje velikost broadcast domény
- je pomalejší než switch, často je dnes nahrazován Layer 3 switchem (MultiLayer Switch)
- vytváří novou hlavičku a ukončení (CRC) rámce

**Princip funkce**

- v paměti si sestavuje routovací tabulku podle sítí, kam má přímo připojené interfacy, podle statických hodnot a podle informací od ostatních routerů (záleží na použitém protokolu)
- u příchozích paketů se dívá na cílovou IP adresu a podle routovací tabulky určuje cestu k cíli (odesílá data na daný port)
- při odesílání dat modifikuje hlavičku rámce, jako zdrojovou MAC adresu vkládá svojí a jako cílovou buď další router nebo stanici
- pokud cílová IP adresa patří do některého přímo připojeného subnetu, tak odesílá přímo této stanici, přitom se koukne do ARP tabulky, zda má pro danou IP adresu MAC adresu, pokud ne, tak odešle ARP dotaz (kdo má tuto IP?), pokud nedostane odpověď, tak rámec zahodí, pokud ano, tak doplní ARP tabulku a rámec odešle



Autor: Petr Bouška

**Související články:**

## Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

# TCP/IP - Routing - směrování

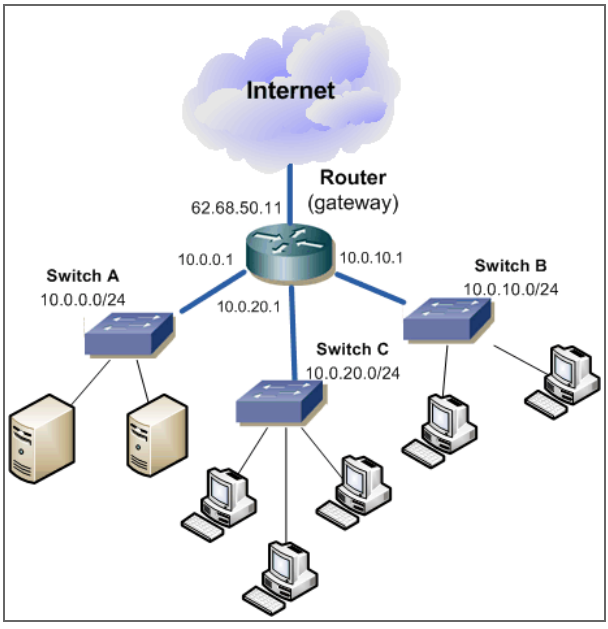
Pátek, 21.09.2007 15:34 | Samuraj - Petr Bouška | sítě

V jedenácté části seriálu o počítačových sítích se věnuji routingu, tedy směrování v sítích. Je zde stručný popis, vysvětlení termínů a pak jsou velmi stručně popsány některé běžnější routovací metody (RIP, IGRP, EIGRP a OSPF) včetně dělení těchto metod. U metod je ukázka základní konfigurace na Ciscu. Článek není zdaleka vyčerpávající a popis je často bodový. V závěru je zmíněno routování na Windows.



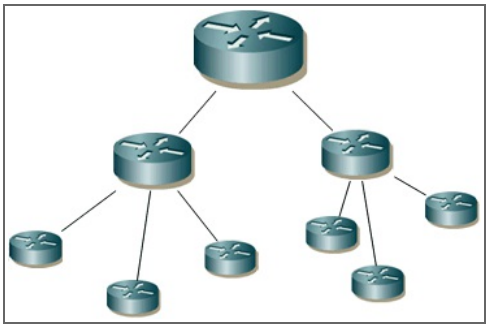
**Routing**, česky řečeno ***směrování***, ale častěji se používá slovo ***routování*** (alespoň v mém okolí). Jedná se o techniku, která slouží k propojení jednotlivých sítí (přesněji subnetů). Původním zařízením, určeným pro routování byl **router**, ale v dnešní době se velmi využívají ***L3 switche***, ***firewally*** nebo pouze ***servery***/počítače. Router přeposílá komunikaci z jedné sítě do jiné.

**N**ásledující obrázek ukazuje jednoduchý příklad sítě, kde máme subnety A, B a C. Tyto subnety jsou propojeny přes router mezi sebou a také do internetu. Takže pokud chce například stanice ze subnetu B komunikovat se serverem v subnetu A, tak pošle data na router a ten zařídí doručení do subnetu A. Pokud by stanice chtěla komunikovat do internetu, tak router naopak zašle data na jiný interface.




**D**ělení sítě na subnety je hierarchické a ve všech propojích musí být router. Při komunikaci pak postupujeme směrem nahoru ve stromu na nejbližší vrstvu, která propojuje dané subnety, a pak opět dolů. Délka cesty se počítá podle počtu **hopů** (skok), což je každý přechod ze zařízení na zařízení, je to tedy počet routrů v cestě + 1. Přímé spojení dvou počítačů je dlouhé 1 hop. Používá se také termín ***next hop*** (další skok) a označuje se jím adresa dalšího routeru v cestě.

**N**a dalším obrázku jsem se pokusil zachytit tuto situaci. Je zde malý (a pouze schematický) výřez větší sítě (či internetu). Na listech stromu jsou malé routery, ke kterým jsou připojeny switche a počítače. Tyto routery se sdružují do dalších (větších) a tak dále na několika úrovních. Samozřejmě v praxi jsou větší routery vždy redundantně, aby byla síť odolná vůči výpadku či kvůli vyvažování zátěže.



## Důležité pojmy pro routování

**Router**
zařízení, které provádí routování

**Routing**
routování, přeposílání (forwarding) dat mezi sítěmi

**Route**

cesta, která se použije, zapsaná v routovací tabulce

**Routing table**
routovací tabulka obsahuje záznamy o jednotlivých cestách

**Routing protocol**
routovací protokol slouží ke směrování routovaného protokolu, určuje nejlepší cestu k cíli a posílá routovací informace dalším routrům

**Routed protocol**
routovaný protokol je IP, IPX nebo Apple Talk

Ještě jeden občas používaný termín, který je dobré znát.

**Router on stick**
je router, který je připojený do switche pomocí jednoho trunk portu - tzn. máme pouze jeden router a pouze jednu linku, což přináší velkou zátěž na router i linku a problémy při výpadku

## Dělení routovacích protokolů

Do routovací tabulky se vytváří několik typů záznamů cest (route), záleží na tom, jakým způsobem vznikly. Pakety jsou podle toho směrovány jedním ze základních způsobů routování:

- **statické routování** - ručně zadané cesty (záznamy v routovací tabulce), bezpečné a dobré, ale nereflektuje změny v topologii sítě
- **dynamické routování** - síť se automaticky přizpůsobuje změnám v topologii a dopravě, automaticky se vypočítávají cesty pomocí routovacího protokolu
- **defaultní routování** - pokud neexistuje jiná cesta, tak se použije defaultní

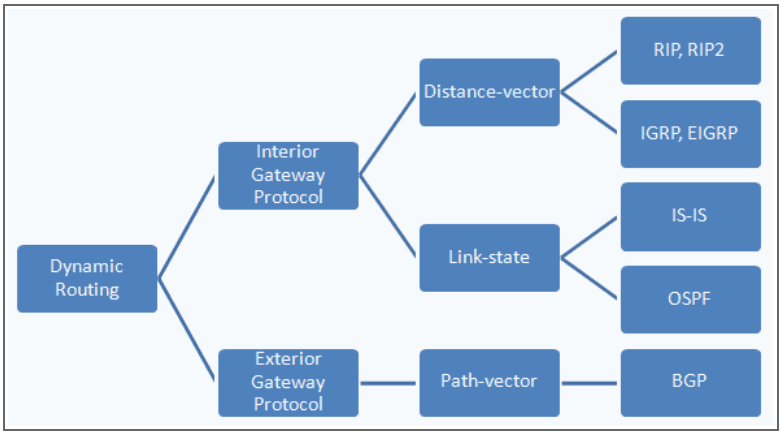
Dynamické routovací protokoly jsou dvou základních typů

- **distance-vector routing protocol** - routery udržují routovací tabulku s informací o (vektoru) vzdálenosti do dané sítě, periodicky ***routovací tabulku*** zasílají sousedům, ti si upraví svoji tabulku a tu opět odešlou dál, pro výpočet nejlepší cesty se používá jedna (počet hopů u RIP) nebo více metrik (propustnost linky a zpoždění u IGRP). Upraveným typem distance-vector protokolu je ***path-vector protocol***.
- **link-state routing protocol** - routery udržují komplexní databázi síťové topologie (vytvořenou pomocí LSA), vyměňují si ***link-state advertisements*** (LSA), LSA jsou vyvolány nějakou událostí v síti, do svého okolí také odesílá Hello pakety, kde zasílá informace o sobě, rychle reaguje na změny topologie, ale spotřebovává více pásma a zdrojů na routeru, metrika je komplexní, nejlepší cesta se počítá pomocí Dijkstrova algoritmu ***shortest path first*** (SPF)

> ***Pozn.:*** *Ještě je zde jeden speciální typ, který vychází z distance-vector protokolu a přidává některé vlastnosti link-state protokolu, označuje se jako* ***hybrid routing protokol*** *nebo* ***advanced distance-vektor protokol***. *Jeho jediným zástupcem je EIGRP.*

Dále dělíme dynamické protokoly podle toho, zda jsou určeny pro nasazení uvnitř lokální sítě (přesněji řečeno uvnitř autonomního systému (AS), který může obsahovat několik LAN) nebo fungují napříč sítěmi (spojují AS dohromady)

- **interior gateway protocol - IGP** - routuje uvnitř Autonomous System (AS)
- **exterior gateway protocol - EGP** - routuje mezi AS




## Obecné termíny

**Variable Length Subnet Masking (VLSM)**
používá se v Classless Inter-Domain Routing (CIDR). V tomto případě můžeme v subnetu použít různé velikosti masky. Můžeme například použít dohormady subnety 10.0.0.0/26 a 10.0.0.64/28.

**Autonomní systém - AS (Autonomous System)**
je skupina IP sítí a routrů, které jsou pod správou jedné (nebo více) jednotek.

**Administrativní vzdálenost - AD (Administrative Distance)**
je vlastnost používaná na routrech k určení nejlepší cesty mezi více routovacími protokoly. Definuje spolehlivost protokolu a prioritizuje lepší nižším číslem. Jinak řečeno na routeru může běžet více routovacích protokolů a podle AD se rozhoduje, který se použije. Na Cisco routrech můžeme měnit defaultní hodnoty.

| protokol | Administrativní vzdálenost |
|---|---|
| přímo připojený interface | 0 |
| statická routa | 1 |
| EIGRP | 90 |
| IGRP | 100 |
| OSPF | 110 |
| RIP | 120 |
| EGP | 140 |

**Split horizon**
metoda, která slouží k zamezení vzniku smyček v Distance Vector Routing protokolech. Používá se u RIP,

IGRP a EIGRP. Funguje tak, že zakazuje posílání routovací cesty zpět na rozhranní, z kterého se naučila.

**Hold-down timer**
je metoda, kterou používají routovací protokoly, aby zabránily zbytečnému nebo předčasnému rozesílání routovacích cest v nestabilním prostředí (ve chvíli, kdy dochází k časté změně stavu). Router čeká určitý čas než je síť stabilní.

## Jednotlivé routovací metody

### RIP - Routing Information Protocol

- jednoduchý pro konfiguraci a funguje všude
- pro malé a střední sítě
- RIP 1 nepodporuje VLSM
- plýtvá pásmem (velká režijní komunikace)
- pomalá konvergence (rozšiřování)
- hloupá metrika - počet hopů
- posílá celou routovací tabulku svým sousedům
- maximálně 15 hopů

Definují se pouze sítě, které jsou přímo na tomto routeru, ostatní se naučí pomocí updatů. Nastavení na Ciscu pomocí

```
SWITCH(config)#router rip
SWITCH(config-router)#network 132.43.54.0
SWITCH(config-router)#network 145.65.76.0
```

### IGRP - Interior Gateway Routing Protocol

- proprietární Cisco protokol
- nepodporuje VLSM
- jako metriku používá cenu, záleží na pásmu a zpoždění
- maximální počet hopů 255

Nastavení na Cisco routu (33 je číslo AS)

```
SWITCH(config)#router igrp 33
SWITCH(config-router)#network 134.43.54.0
SWITCH(config-router)#network 143.56.76.0
```

### EIGRP - Enhanced Interior Gateway Routing Protocol

- proprietární Cisco protokol
- rychlá konvergence
- redukuje spotřebu pásma pro routovací updaty
- podporuje různé protokoly (apple talk, IPX, IP) a VLSM
- metrika - pásmo, zpoždění (případně i zátěž, spolehlivost)
- routovací update se vyměňují pouze mezi routry ve stejném AS
- maximální počet hopů 255
- bez smyček

Používá *tabulku sousedů* (informace o přímých sousedech), *tabulku topologie* (všechny routovací záznamy, které se naučil) a *routovací tabulku* (nejlepší routy z tabulky topologie).

Nastavení na Cisco routeru (33 je číslo AS)

```
SWITCH(config)#router eigrp 33
SWITCH(config-router)#network 172.16.0.0
SWITCH(config-router)#network 10.0.0.0
```

### OSPF - Open Shortest Path First

- hierarchický systém - jedna nebo více oblastí je spojena k páteřní oblasti (oblast 0)
- routry posílají linkstate (pásmo a stav interfacu) informace všem sousedním routrům v oblasti
- routry vytvářejí databázi topologie, což je model celé oblasti
- z databáze se pomocí Dijkstry vypočítá nejkratší cesta a zapíše do routovací tabulky
- neomezený počet hopů
- určeno pro rozsáhlé sítě heterogenní sítě
- podporuje VLSM

Nastavení na Cisco routeru (1 je ID číslo procesu, pouze lokálně významné)

```
SWITCH(config)#router ospf 1
SWITCH(config-router)#network 132.43.56.0 0.0.0.255 area 0
SWITCH(config-router)#network 145.54.34.6 0.0.63.255 area 0
```

### Static routing

- používá se například mezi ISP a firmou, není zde třeba, aby běhal složitý routovací protokol
- pro každou síť je vložen záznam do routovací tabulky
- žádná zátěž
- pouze pro malé sítě

Nastavení na Cisco routru

```
ip route [destination_network] [mask] [next_hop or exit_interface] [administrative_distance] [permanent]
SWITCH(config)#ip route 192.168.50.0 255.255.255.0 192.168.1.1
```

***next_hop*** je IP adresa dalšího routeru v cestě, přesněji, je to adresa interfacu sousedního routeru, který sousedí s tímto routerem,
***exit_interface*** je jméno lokálního výstupního interfacu (třeba s0), přes který vede cesta k cílové síti.

### Default routing

- může se použít pouze na kraji sítě, kde je jeden port, který vede mimo síť
- pokud není definována jiná cesta k dosažení cíle, tak se použije defaultní routa, což je gateway

Nastavení na Cisco routru

```
ip route 0.0.0.0 0.0.0.0 [next_hop or exit_interface]
SWITCH(config)#ip route 0.0.0.0 0.0.0.0 62.102.58.12
```

## Routování a Windows

I pracovní stanice, kde je například OS Windows XP, musí provádět rozhodnutí o tom, kam směrovat síťová data. Vychází se z toho, jak jsou nastaveny síťové karty v počítači. Podle tohoto nastavení se dynamicky upravuje ***routovací tabulka***. Ale údaje v tabulce můžeme upravovat i ručně. Podle routovací tabulky se Windows rozhodují, přes kterou síťovou kartu zaslat určitá data.

***Základní routy*** jsou pro *gateway*, tedy to co nevím kam jinam poslat, označuje se jako ***default route***. Pak jsou routy pro *hosta* a *loopback* (127.0.0.0/8), které vedou na *MS TCP Loopback Interface*. Dále je routa pro *multicast* (224.0.0.0/4). V neposlední řadě je zde routa pro *lokální subnet*.

Jednotlivé routy mají ***metriku*** (1 až 9999), která určuje jejich prioritu. Čím nižší hodnota, tím větší priorita. Routa se vybírá nejprve tak, aby nejblíže odpovídala cílové adrese. Pokud je třeba, tak je druhým kritériem metrika.

Jak jsem psal, tak se routovací tabulky vytváří a upravuje dynamicky podle interfaců. Můžeme manuálně mazat, přidávat či upravovat záznamy, ale tyto změny se při restartu počítače ztratí. Pokud chceme vytvořit trvalou statickou routu, tak ji musíme vytvořit jako ***persistant***. Tyto routy jsou pak uloženy v registech `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\PersistentRoutes`.

Pro manipulaci s routovací tabulkou ve Windows XP (ale dalších MS OS) slouží příkaz `route`. Pár příkladů použití:

```
route [-f] [-p] [Command [Destination] [mask Netmask] [Gateway] [metric Metric]] [if Interface]]
route print                                          // vypíše routovací tabulku
route add 0.0.0.0 mask 0.0.0.0 192.168.12.1          // vytvoří dočasnou defaultní routu, tedy gateway
route -p add 10.41.0.0 mask 255.255.0.0 10.27.0.1 metric 7  // vytvoří trvalou (persistant) routu pro 10.41.0.0/16
route delete 10.41.0.0 mask 255.255.0.0              // smaže záznam pro 10.41.0.0/16
```

## Windows jako router

***Routování***, o kterém jsem tu psal, se týká pouze rozhodnutí, kam směrovat provoz ***generovaný na tomto počítači***. Pokud bychom chtěli použít počítač jako ***router***, tedy aby příchozí komunikaci posílal dál, tak musíme tuto funkci zapnout v registrech.

V registrech `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters` musíme změnit hodnotu položky `IPEnableRouter` na `1`.

> ***Pozn.:*** *Chtěl bych upozornit, že zapnutí této funkce je třeba dobře rozvážit, protože se může jednat o bezpečnostní riziko. Pomocí této funkce může dojít k propojení několika sítí (což asi v tomto případě chceme), takže ji nezapínat bezdůvodně.*



Autor: Petr Bouška

**Související články:**

## Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

- OSI model [05.03.2007 08:09]
- VLAN - Virtual Local Area Network [02.06.2007 15:54]
- Počítačové sítě a jejich typy [09.07.2007 15:32]
- Počítačové sítě - základní topologie [30.07.2007 14:47]
- Ethernet - CSMA/CD, kolizní doména, duplex [03.08.2007 12:40]
- TCP/IP - model, encapsulace, paket vs. rámec [16.08.2007 16:43]
- TCP/IP - metody vysílání dat [02.09.2007 15:05]
- TCP/IP - adresy, masky, subnety a výpočty [05.09.2007 14:53]
- TCP/IP - navázání a ukončení spojení [13.09.2007 16:10]
- TCP/IP a ethernet - cesta v síti, aktivní síťové prvky [17.09.2007 10:22]
- TCP/IP - Routing - směrování [21.09.2007 15:34] ..... **právě čtete**
- TCP/IP - nalezení MAC adresy k IP - ARP [25.09.2007 17:35]
- Počítačové sítě - Computer Networks [30.09.2007 13:28]
- DNS (Domain Name System) zaměřeno na Microsoft [26.11.2007 14:26]
- Diagramy a schémata počítačové sítě (Visio) [13.01.2008 18:02]
- Wake on LAN - lokální i vzdálený subnet [10.08.2008 20:42]
- TCP/IP - Internet Protocol Version 6 - IPv6 [05.03.2009 15:41]
- TCP/IP - skupinové vysílání IP Multicast a Cisco [10.03.2009 20:05]
- Běžné útoky na switche, Cisco Dynamic ARP Inspection [18.06.2009 10:15]
- Cisco - Router Switching metody a související termíny - CAM, FIB, CEF [28.06.2009 17:15]
- Výstavba moderní sítě [28.04.2010 14:41]
- Microsoft Network Load Balancing (NLB) a Cisco switche [09.08.2010 18:26]
- Cisco Network Design - Enterprise Architecture [24.09.2010 13:47]
- VPN 1 - IPsec VPN a Cisco [10.04.2011 19:59]

## Routing - směrování

Směrování paketů mezi jednotlivými počítačovými sítěmi (LAN) se provádí technikou zvanou routing. Používají se k tomu různé routovací protokoly. Routing je jedna ze základních částí komunikace v internetu.

- TCP/IP - Routing - směrování [21.09.2007 15:34] ..... **právě čtete**
- Cisco IOS 18 - inter-VLAN routing a ACL - směrování mezi VLANy [24.12.2008 11:50]
- Cisco Routing 1 - obecné vlastnosti směrovacích protokolů [20.03.2009 14:43]
- Cisco Routing 2 - EIGRP - Enhanced Interior Gateway Routing Protocol [29.03.2009 19:04]
- Cisco Routing 3 - OSPF - Open Shortest Path First [03.04.2009 10:54]
- Cisco Routing 4 - IS-IS - Intermediate System to Intermediate System [09.04.2009 08:48]
- Cisco Routing 5 - BGP - Border Gateway Protocol [18.04.2009 13:50]
- Cisco Routing 6 - srovnání routovacích protokolů [28.04.2009 16:27]

# TCP/IP - nalezení MAC adresy k IP - ARP

Úterý, 25.09.2007 17:35 | Samuraj - Petr Bouška | sítě

Zatím poslední dvanáctá část seriálu o počítačových sítích se stručně věnuje protokolu ARP. Jedná se o důležitý protokol, který slouží k nalezení MAC adresy stanice, která má určitou IP adresu. Protože MAC adresy potřebujeme při každé komunikaci, tak se ARP využívá velice často.



***Počítačové sítě*** jsou často založeny na technologii *ethernet* a používají protokol *TCP/IP*. Ve chvíli, kdy chtějí spolu dvě zařízení komunikovat, tak potřebují znát ***MAC adresu*** a ***IP adresu***. Svoje údaje samozřejmě stanice zná (má je nastaveny). Cílová stanice se běžně adresuje pomocí ***doménového jména*** (DN - domain name či FQDN - full qualified domain name), které lze (a musíme) převést na ***IP adresu***.

Princip komunikace mezi stanicemi v síti jsem pospal v článku **TCP/IP a ethernet - cesta v síti, aktivní síťové prvky**. Zde vidíme, že vždy potřebujeme MAC adresu buď cílové stanice nebo routeru/gatewaye na cestě (další hop). Takže potřebujeme najít MAC adresu nějaké stanice, u které známe její IP adresu.

Standardizovaný protokol, který hledá MAC adresu k zadané IP adrese, se jmenuje **Address Resolution Protocol** (ARP) a je definován v ***RFC 826***. Jedná se o protokol, který běží na *síťové vrstvě TCP/IP modelu* (2. vrstva), tedy na stejné úrovni jako IP protokol.

> ***Pozn.:*** *Protokol ARP je sice primárně určen pro ethernet, ale pracuje i v dalších technologiích jako Token Ring, IEEE802.11, FDDI či ATM.*

## Princip funkce ARP

> ***Pozn.:*** *Jako zdrojovou stanici označuji ten stroj, který hledá MAC adresu podle IP adresy. Cílová stanice má nastavenu hledanou IP adresu.*

- zdrojová stanice sestaví ***ARP žádost*** (request) a odešle ji jako ***broadcast***
- všechny stanice na lokálním segmentu přijmou žádost, a pokud nemají tuto IP, tak ji ignorují
- cílová stanice sestaví ***ARP odpověď*** (response) a odešle ji jako ***unicast*** zdrojové stanici

> ***Pozn.:*** *Jako broadcastová adrese se používá* `ff:ff:ff:ff:ff:ff`

## Struktura ARP paketu

Formát žádosti i odpovědi je obdobný. Rozdíl je v typu u rámce, požadavek má v ethernetové hlavičce nastaven typ na 0x0806 a odpověď na 0x0835. Dále se liší vyplněné hodnoty, operation code je 1 pro žádost a 2 pro odpověď. V žádosti je cílová hardwarová adresa nastavena na nuly a v odpovědi je nastavena na správnou hodnotu.

> ***Pozn.:*** *V obecném případě je MAC adresa hardwarová adresa a IP adresa je protokol adresa. Já zde popisuji použití pro IP a MAC adresy, ale všude je možno zaměnit dle technologie.*

| bits | 0 - 7 | 8 - 15 | 16 - 31 |
|---|---|---|---|
| 0 | Hardware type (2B) | | Protocol type (2B) |
| 32 | Hardware size (1B) | Protocol size (1B) | Operation code (2B) |
| 64 | Sender MAC address (6B) | | |
| 96 | Sender MAC address (cont.) | | Sender IP address (4B) |
| 128 | Sender IP address (cont.) | | Target MAC address (6B) |
| 160 | Target MAC address (cont.) | | |
| 192 | Target IP address (4B) | | |

## ARP oznámení (announcement)

Speciálním typem ARP paketu je ***ARP oznámení*** (announcement), které neslouží ke zjištění něčí MAC adresy, ale oznámení mojí MAC adresy ostatním stanicím na síti. Paket je většinou podobný ARP dotazu, má vyplněny IP a MAC adresu odesílatele a jako IP adresu cíle má svoji IP. Používá se například, když se změní IP adresa stanice, rozesláním oznámení se aktualizuje ARP cache ostatních stanic.

## ARP cache

Aby se při probíhající komunikaci nemusely stále posílat ARP dotazy, tak se na zařízeních používá ***ARP cache***, kde se po určitou dobu (v řádu minut) uchovávají ***kombinace IP adresy*** a ***MAC adresy***.

V operačním systému ***Windows*** nemáme nástroj pro vytvoření ARP dotazu, ale stačí použít například příkaz ping na patřičnou IP adresu. ARP se používá automaticky pokaždé, když je třeba zaslat rámec (data do sítě). Pro prohlížení ***ARP cache*** můžeme použít řádkový příkaz `arp -a`.

## Inverse ARP

Opačným protokolem k ***ARP*** je ***Inverse Address Resolution Protocol*** (Inverse ARP nebo InARP), který k zadané MAC adrese hledá IP adresu. InARP se používal u Frame Relay a ATM sítí, dnes se s ním asi nesetkáme.

## Reverse ARP

Protokol ***Reverse ARP*** (RARP), podobně jako InARP, slouží k nalezení IP adresy k MAC adrese. RARP ovšem hledá IP adresu pro sebe sama. Tato metoda se již dnes nepoužívá, protože byla nahrazena nejprve pomocí ***BOOTP*** (Bootstra

protocol) a dnes ***DHCP*** (Dynamic Host Configuration Protocol). Pro ***RARP*** musely být na serveru manuálně nastaveny přiřazení MAC adresy a IP adresy, navíc to není protokol nad IP, ale samostatně na stejné úrovni. Je definován v RFC 903.



Autor: **Petr Bouška**

**Související články:**

## Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

- **OSI model** [05.03.2007 08:09]
- **VLAN - Virtual Local Area Network** [02.06.2007 15:54]
- **Počítačové sítě a jejich typy** [09.07.2007 15:32]
- **Počítačové sítě - základní topologie** [30.07.2007 14:47]
- **Ethernet - CSMA/CD, kolizní doména, duplex** [03.08.2007 12:40]
- **TCP/IP - model, encapsulace, paket vs. rámec** [16.08.2007 16:43]
- **TCP/IP - metody vysílání dat** [02.09.2007 15:05]
- **TCP/IP - adresy, masky, subnety a výpočty** [05.09.2007 14:53]
- **TCP/IP - navázání a ukončení spojení** [13.09.2007 16:10]
- **TCP/IP a ethernet - cesta v síti, aktivní síťové prvky** [17.09.2007 10:22]
- **TCP/IP - Routing - směrování** [21.09.2007 15:34]
- **TCP/IP - nalezení MAC adresy k IP - ARP** [25.09.2007 17:35] ..... **právě čtete**
- **Počítačové sítě - Computer Networks** [30.09.2007 13:28]
- **DNS (Domain Name System) zaměřeno na Microsoft** [26.11.2007 14:26]
- **Diagramy a schémata počítačové sítě (Visio)** [13.01.2008 18:02]
- **Wake on LAN - lokální i vzdálený subnet** [10.08.2008 20:42]
- **TCP/IP - Internet Protocol Version 6 - IPv6** [05.03.2009 15:41]
- **TCP/IP - skupinové vysílání IP Multicast a Cisco** [10.03.2009 20:05]
- **Běžné útoky na switche, Cisco Dynamic ARP Inspection** [18.06.2009 10:15]
- **Cisco - Router Switching metody a související termíny - CAM, FIB, CEF** [28.06.2009 17:15]
- **Výstavba moderní sítě** [28.04.2010 14:41]
- **Microsoft Network Load Balancing (NLB) a Cisco switche** [09.08.2010 18:26]
- **Cisco Network Design - Enterprise Architecture** [24.09.2010 13:47]
- **VPN 1 - IPsec VPN a Cisco** [10.04.2011 19:59]

# Počítačové sítě - Computer Networks

Neděle, 30.09.2007 13:28 | Samuraj - Petr Bouška | sítě

Tento článek uzavírá sérii o počítačových sítích. Některé díly seriálu se věnovaly věcem jednoduchým a obecně známým, jiné něčemu složitějšímu. Nesnažil jsem se jít v popisu do úplných detailů, ale naopak přinést globální pohled na danou problematiku, a pokud někdo potřebuje detailnější popis, tak již jednoduše najde podle klíčových slov potřebné materiály na internetu. Tento článek shrnuje odkazy na předchozí díly a jednoduše popisuje vazby a návaznosti jednotlivých kapitol.



Mým cílem bylo věnovat se věcem praktickým, s kterými se člověk denně setkává. Na rozdíl od způsobu, jakým se sítě probírají na vysoké škole. Ale přesto vyjít od úplných základů. Od obecného začátku jsem se dále věnoval primárně lokálním sítím (LAN).

- Počítačové sítě a jejich typy. Pokud chceme mluvit o ***počítačových sítích***, tak si musíme určit, co tento termín znamená a jak můžeme počítačové sítě rozdělovat.
- Počítačové sítě - základní topologie. Při vytváření počítačové sítě, se dle technologie, zapojují prvky různým způsobem a používáme tedy určitou ***síťovou topologii***.
- OSI model. Pro popis komunikace v počítačové síti se používá ***model ISO OSI***, který využívá systému vrstev, kdy spolu mohou komunikovat pouze sousední vrstvy.
- Ethernet - CSMA/CD, kolizní doména, duplex. V sítích typu LAN (ale často i ve větších) se dnes převážně využívá technologie ***Ethernet*** (oproti dalším jako Token Ring a FDDI). Základní popis Ethernetu a principu jakým přistupuje k fyzickému médiu, tedy jak komunikuje, popisuje tento článek.
- TCP/IP - model, encapsulace, paket vs. rámec. V Ethernetových sítích je majoritní protokol ***TCP/IP***. Tento protokol se popisuje pomocí *TCP/IP modelu* a při odesílání se provádí tzv. *encapsulace* dat jednotlivých vrstev. Úvod do TCP/IP spolu s modelem, popisem encapsulace a formátem rámce (což odkazuje na Ethernet) a paketu, je uveden v tomto článku.
- TCP/IP - metody vysílání dat. Pokud chceme zaslat data pomocí TCP/IP, musíme nejprve zvolit správnou ***metodu vysílání***, podle toho, zda adresujeme jednoho nebo více příjemců.
- TCP/IP - adresy, masky, subnety a výpočty. Protokol TCP/IP používá pro adresování ***IP adresy*** a ***masky podsítě***, pomocí kterých se síť rozděluje do logických bloků - ***subnetů***. Popis těchto termínů, spolu s metodami výpočtu, naleznete zde.
- VLAN - Virtual Local Area Network. Dalším význačným pomocníkem, při logickém členění sítě na menší části, jsou ***virtuální lokální sítě - VLANy***.
- TCP/IP - navázání a ukončení spojení. Komunikace v TCP/IP začíná vždy ***navázáním spojení*** a ukončuje se předepsaným způsobem.
- TCP/IP a ethernet - cesta v síti, aktivní síťové prvky. Pokud chci zaslat data jiné stanici, musím ji správně adresovat, tedy použít správnou *fyzickou adresu* (MAC - vychází z ethernetu) a *logickou adresu* (IP z protokolu TCP/IP). O většinu věcí se automaticky postará komunikační protokol, ale je dobré vědět, jak to funguje a jakým způsobem data putují po síti. To je závislé také na tom, jaké aktivní síťové prvky jsou v cestě.
- TCP/IP - Routing - směrování. Pokud se data přenášejí mezi různými subnety (sítěmi), tak musí projít přes router nebo jiné zařízení, které provádí ***routování***. *Routovacích protokolů* existuje celá řada.
- TCP/IP - nalezení MAC adresy k IP - ARP. Pro komunikaci v síti jsou důležité některé další protokoly. Je to například ***ARP*** pro nalezení MAC adresy k IP adrese. ***DHCP*** pro dynamické přiřazování IP adres stanicím. A ***DNS*** pro překlad IP adres na doménová jména a naopak. Stručné vysvětlení principu ***protokolu ARP*** je zde.
- DNS (Domain Name System) zaměřeno na Microsoft. Tento článek trochu vybočuje z konceptu série, ale svým obsahem do ní zapadá. Pro funkci internetu je nezbytná komponenta z TCP/IP, která překládá snadno zapamatovatelná doménová jména na IP adresy.



Autor: Petr Bouška

**Související články:**

## Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

- OSI model [05.03.2007 08:09]
- VLAN - Virtual Local Area Network [02.06.2007 15:54]
- Počítačové sítě a jejich typy [09.07.2007 15:32]
- Počítačové sítě - základní topologie [30.07.2007 14:47]
- Ethernet - CSMA/CD, kolizní doména, duplex [03.08.2007 12:40]
- TCP/IP - model, encapsulace, paket vs. rámec [16.08.2007 16:43]
- TCP/IP - metody vysílání dat [02.09.2007 15:05]
- TCP/IP - adresy, masky, subnety a výpočty [05.09.2007 14:53]
- TCP/IP - navázání a ukončení spojení [13.09.2007 16:10]
- TCP/IP a ethernet - cesta v síti, aktivní síťové prvky [17.09.2007 10:22]
- TCP/IP - Routing - směrování [21.09.2007 15:34]
- TCP/IP - nalezení MAC adresy k IP - ARP [25.09.2007 17:35]
- Počítačové sítě - Computer Networks [30.09.2007 13:28] ..... **právě čtete**
- DNS (Domain Name System) zaměřeno na Microsoft [26.11.2007 14:26]
- Diagramy a schémata počítačové sítě (Visio) [13.01.2008 18:02]
- Wake on LAN - lokální i vzdálený subnet [10.08.2008 20:42]
- TCP/IP - Internet Protocol Version 6 - IPv6 [05.03.2009 15:41]
- TCP/IP - skupinové vysílání IP Multicast a Cisco [10.03.2009 20:05]
- Běžné útoky na switche, Cisco Dynamic ARP Inspection [18.06.2009 10:15]
- Cisco - Router Switching metody a související termíny - CAM, FIB, CEF [28.06.2009 17:15]
- Výstavba moderní sítě [28.04.2010 14:41]
- Microsoft Network Load Balancing (NLB) a Cisco switche [09.08.2010 18:26]
- Cisco Network Design - Enterprise Architecture [24.09.2010 13:47]
- VPN 1 - IPsec VPN a Cisco [10.04.2011 19:59]

# DNS (Domain Name System) zaměřeno na Microsoft

Pondělí, 26.11.2007 14:26 | Samuraj - Petr Bouška | Microsoft admin

Tento článek tematicky navazuje na sérii o základech počítačových sítí, ale je psán jinou formou. K jeho napsání mne inspirovaly certifikační testy firmy Microsoft, které velmi často obsahují otázky zaměřené na DNS a to hlavně na použití různých typů zón. Takže v první části je určitý teoreticko-praktický popis funkce DNS, který rozhodně není vyčerpávající. A v druhé části se řeší různé použití DNS v podání Microsoftu. Jedná se i o věci, které jsem v praxi nikdy nepoužil, ale jsou potřebné do testů.



## Funkce DNS

**Domain Name System**, známější pod svou zkratkou DNS, je internetový standard zahrnutý v TCP/IP. Slouží k překladu ***jmen objektů*** na ***IP adresy*** či jiné zdrojové záznamy (resource records). Jména objektů se označují jako ***doménová jména*** (domain name) a nejčastěji se jedná o jména hostitelů (hostname), jsou to alfanumerické řetězce, které jsou lépe zapamatovatelné než IP adresy. Příkladem doménového jména je `www.samuraj-cz.com` a k němu náleží IP adresa `193.86.238.17`.

**DNS** nabízí i obrácenou funkci a to je překlad ***IP adres*** na ***jména objektů***. K tomu se využívají tzv. ***PTR záznamy***. Záznamy v DNS dnes existují nejen pro *hostname*, ale také pro řadu služeb. Nejpoužívanějším příkladem je ***MX záznam*** pro poštovní server. Díky tomu nemusíme znát ani jméno serveru ani jeho IP adresu, ale pouze doménu, pro kterou chceme poštovní server nalézt. To využívají ostatní poštovní servery, když chtějí doručit email (z emailové adresy zjistí doménu a k ní naleznou MX záznam, tedy cíl komunikace).

**V**ýhodou používání ***internetových jmen*** je lepší zapamatovatelnost a také to, že je možno změnit fyzické umístění počítače a jeho IP adresu a přitom používat stále stejné jméno. Přitom pro komunikaci pomocí TCP/IP se musí používat ***IP adresy***. Funkci internetu bychom si asi nedokázali bez DNS představit. Microsoft na DNS postavil i funkci svého firemního prostředí - domény a využívá jej například pro lokalizaci řady doménových služeb.

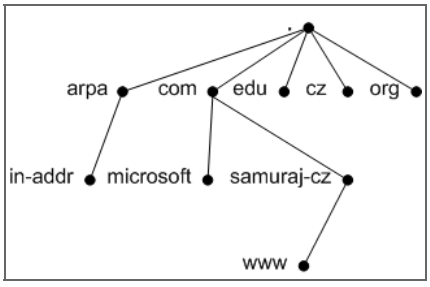
***Protokol DNS*** využívá pro komunikaci porty `TCP 53` a `UDP 53`. Definován je v RFC 1035 (a různé funkce v řadě dalších).

## Tvorba doménových jmen

***Jmenný prostor*** v internetu je rozdělen na ***domény*** (domain). Zodpovědnost za zprávu jmen uvnitř každé domény je delegována, typicky na systémy uvnitř této domény. Tomu odpovídá i hierarchická organizace serverů a způsob tvoření doménových jmen. Podle delegace oprávnění se ještě mluví o ***zónách***. Většinou je zóna rovna jedné doméně, ale může zahrnovat i několik domén, které jsou spravovány jednou autoritou.

**P**odobně jako jsou rozděleny rozsahy IP adres na *sítě* a *podsítě*, tak se dělí jmenné názvy na *domény* a *subdomény*. Ale přesto zde nemusí být pevná vazba mezi rozsahy IP adres a doménových jmen. Například dvě různá jména domén mohou odkazovat na stejné adresy.

***Obor doménových jmen*** DNS je tvořen ***stromem*** (hierarchická struktura). Každý uzel stromu obsahuje informace o doméně (kterou spravuje, tedy různé záznamy v dané doméně) a odkazy na subdomény. Kořenem stromu je ***kořenová doména***, která se zapisuje jako *tečka* (.). Pod ní následují ***domény nejvyšší úrovně*** (TLD - Top Level Domain, například com a cz). Dále ***domény druhé úrovně*** (Second Level Domain, třeba microsoft.com) a případně další subdomain.



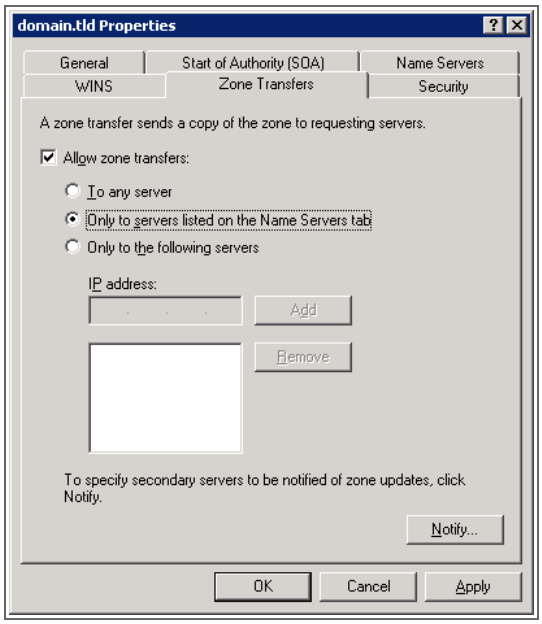

**K**dyž skládáme ***doménové jméno***, tak používáme domény od nejnižší úrovně a zapisujeme je zleva doprava oddělené tečkou. Tedy k obrázku např. `www.samuraj-cz.com`. Obrázek není úplně přesný, protože `www` není podřízená doména, ale záznam v doméně `samuraj-cz.com`.

**P**ro ***reverzní překlad*** IP adres na jména objektů se využívá pseudo domény `IN-ADDR.ARPA`. Z důvodu řazení podle významnosti se používá reverzní tvar IP adresy, např. pro IP `192.168.0.1` je DNS záznam `1.0.168.192. IN-ADDR.ARPA`.

## Zónové soubory

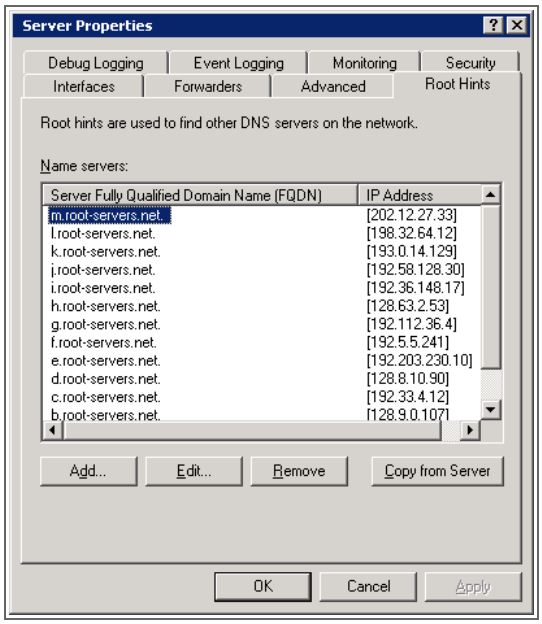
**J**ak jsem zmínil, strom doménových jmen se dělí na ***zóny***, tedy oblasti spravované jedním správcem (organizací). Zóna obsahuje jednu (nejčastěji) nebo více domén. V zóně jsou také uvedeny autoritativní informace o spravovaných doménách. Tyto informace poskytuje autoritativní DNS server, tedy server, který je považovaný za důvěryhodný pro zónu.

**O**bsah zóny, jednotlivé zdrojové záznamy, je uložen v ***zónovém souboru*** (zone file). To je většinou textový soubor. Mezi některými DNS servery může docházet k replikaci záznamů (například mezi primárním a sekundárním NS), tomuto procesu se říká ***zone transfer***.




## Nalezení záznamu v DNS

Standardně DNS server provádí rekurzivní (opakované) dotazy. Nejprve potřebuje vědět, kde začít hledat jména v první vrstvě oboru názvů DNS. Tyto informace jsou obsaženy v tzv. **root hints**, což je seznam úvodních záznamů, které použije DNS služba, aby našla servery, které jsou autoritativní pro kořen stromu oboru názvů DNS domény (DNS domain namespace tree). Standardně *root hints* obsahují odkazy na ***13 root serverů***, které se nachází po celém světě a zajišťují technickou infrastrukturu internetu.

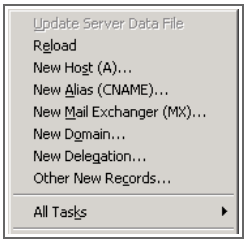



Vezmeme adresu, kterou chceme přeložit (např. www.samuraj-cz.com) a rozdělíme ji na jednotlivé domény postupně zprava. Kořenovému serveru pošleme dotaz na adresu DNS serveru nejvyšší úrovně (tedy pro com). Dostaneme adresu TLD NS a jeho se zeptáme na adresu autoritativního serveru pro doménu druhé úrovně (samuraj-cz.com). A tak můžeme pokračovat dále. Na nejnižší úrovni se zeptáme již na IP adresu záznamu (www.samuraj-cz.com).

## Typy záznamů v DNS

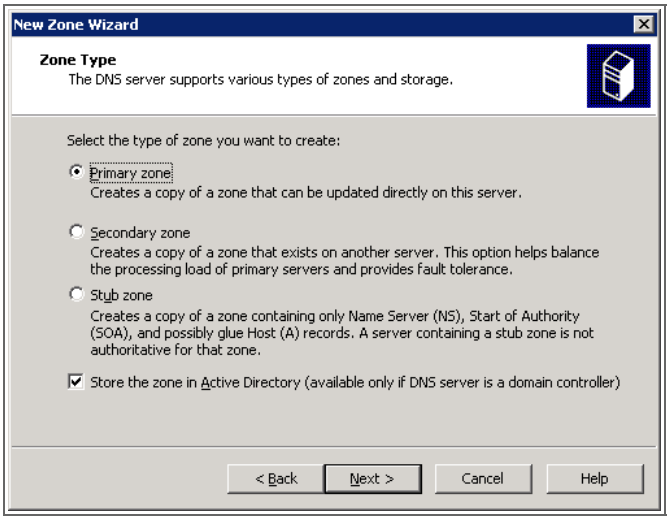
DNS podporuje řadu různých typů záznamů, podle typu záznamu uchovává různé parametry. Obecné parametry pro všechny typy záznamů jsou *jméno*, *třída* (pouze IN jako internet), *TTL* (čas jak dlouho může být záznam uložen v keši), *typ záznamu*, *data záznamu*. Zde uvádím několik nejdůležitějších typů.

- **host – address (A)** – běžný záznam, obsahuje adresu počítače
- **alias – canonical name (CNAME)** – další jméno (alias) pro existující záznam v doméně
- **mail exchanger (MX)** – adresa poštovního serveru
- **service location (SRV)** – adresa některé služby, jako ldap, kerberos, ftp, a další
- **name server (NS)** – seznam serverů, které zajišťují DNS služby pro doménu, záznam se nachází v nadřízené doméně a v aktuální doméně
- **pointer (PTR)** – užívají se pro reverzní překlad
- **start of authority (SOA)** – odkazuje na server, kde jsou primární údaje (primární NS), a obsahuje údaje pro zone transfer




## Typy zón

Služba DNS je systémovou komponentou na *Windows Serveru 2003*. V tomto DNS můžeme vytvořit celou řadu různých zón. Nejčastěji se používá ***primární zóna*** (primary zone) a pokud jsme v interní síti a používáme doménu, tak je nejlepší volbou ***integrovaná do Active Directory***.




### Primary zone

- obsahuje autoritativní kopii dat pro zónu a podporuje jejich zápis a změnu
- standardní primární zóna ukládá databázi zón do textového souboru
- může existovat pouze jeden primární server (používá hlavní kopii zóny)

### Secondary zone

- obsahuje autoritativní kopii dat pro zónu pouze pro čtení
- přidává se k primární zóně z důvodu zálohy, zvyšuje dostupnost a odolnost proti poruše
- data získává pomocí zone transferu z primárního serveru nebo z jiného sekundárního serveru
- zone transfer spotřebovává dost pásma (může být plný nebo inkrementální)

### Stub zone

- obsahuje autoritativní kopii dat pro zónu pouze pro čtení
- uchovává pouze odkazy na autoritativní DNS servery domény
- jinak řečeno jedná se o kopii zóny, která obsahuje pouze NS, SOA a glue A záznamy
- slouží hlavně ke zvýšení výkonu rozeznávání jmen (name resolution), není třeba kontaktovat několik DNS serverů v internetu, ale rovnou se nalezne autoritativní DNS pro doménu

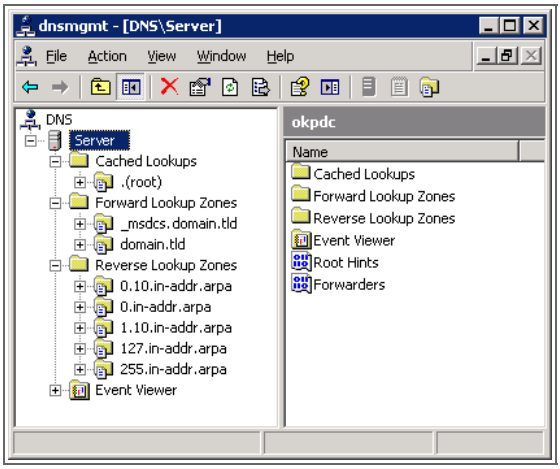
### Active Directory-Integrated zone

- jedná se o primární zónu (tedy autoritativní se zápisem)
- databáze zón je uložena v Active Directory databázi (místo v souboru)
- díky tomu se zjednodušuje administrace, zvyšuje bezpečnost a umožňuje automatické replikace (spolu s replikací AD)
- je možno použít pouze, pokud je DNS nainstalováno na doménovém kontroléru
- těchto serverů (kopií zóny) může existovat tolik, kolik je doménových řadičů (oproti standardnímu jednomu primárnímu serveru)

### Lookup Zone

Pro správu Microsoft DNS služby slouží snap-in do MMC konzole. V této konzoli se připojíme k určitému DNS serveru a ve stromu pod ním se nachází několik položek.

- ***Cached Lookups*** – každý DNS server standardně uchovává po určitou dobu záznamy, které v poslední době hledal, aby opakovaný dotaz mohl rychleji zodpovědět (pro zobrazení těchto záznamů musíme mít zapnuto Advanced View)
- ***Forward Lookup Zones*** – teprve zde jsou jednotlivé zóny (může jich být na serveru více), obsahuje mapování jmen na IP
- ***Revers Lookup Zones*** – obsahuje obrácené mapování IP adres na jména (PTR záznamy)




## Další možnosti

### Private root zone

Pro vnitřní síť firmy je často lepší (z důvodu bezpečnosti a správy) vytvořit vlastní oddělený prostor jmen, který je oddělený od veřejných sítí. K tomu využijeme ***interní DNS kořen*** (internal DNS root) a na něm vytvoříme ***privátní DNS kořenovou zónu*** (private DNS root zone).

> ***Pozn.:*** *I v odděleném jmenném prostoru by se neměla používat jména, která již existují v internetu.*

### Caching-only DNS

Tento server neobsahuje žádnou zónu, jeho účelem je ukládat vyřízené dotazy do keše, aby se při opakovaném dotazu mohli vyřídit okamžitě.

### Forwarding

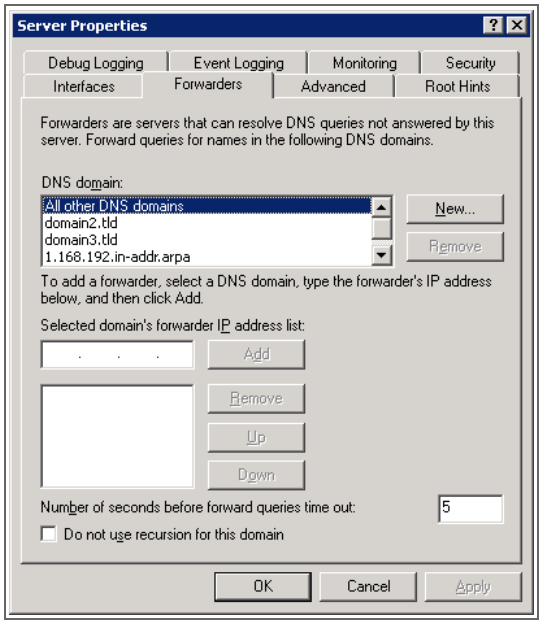
***Forwarding*** bychom mohli přeložit jako předávání (lépe řečeno přeposílání) a jedná se o metodu, která se v praxi hodí pro nasazení v řadě scénářů.

***Forwarding*** slouží k přeposlání dotazu pro *externí DNS jméno* na *DNS server* mimo naši síť. Oproti klasickému chování (podrobněji popsanému výše), kdy se začíná hledáním v *root hints* a pak následuje řada dotazů různým serverům. Pro konfiguraci ***forwardingu*** tedy pouze specifikujeme určitý *DNS server* a následně jsou všechny externí dotazy (které nevyřídí keš) přeposílány na tento server. Po přijetí odpovědi, je tato zaslána klientovi.

Použití ***forwardingu*** se může hodit z důvodu ***bezpečnosti***, ***zvýšení výkonu*** či lepšího ***využití internetové linky***.

- V případě špatného připojení k internetu, můžeme optimalizovat výkon překladu jmen tak, že všechny dotazy posíláme přes jeden ***forwarder*** (místo celé řady spojení dotaz-odpověď je pouze jedno).
- Při použití ***forwardingu*** můžeme na firewallu povolit komunikaci pouze na jeden venkovní DNS server a tím zvýšit bezpečnost.
- Můžeme nastavit ***conditional forwarding*** na autoritativní servery některých domén, takže z nich získáváme odpovědi přímo.

***Forwarder*** posílá trochu jiný typ DNS dotazu, jedná se o ***rekurzivní dotaz*** (dotazovaný server musí provést rekurzivně další dotazy a poslat odpověď), na rozdíl od klasického dotazu, který se označuje jako ***iterativní***.



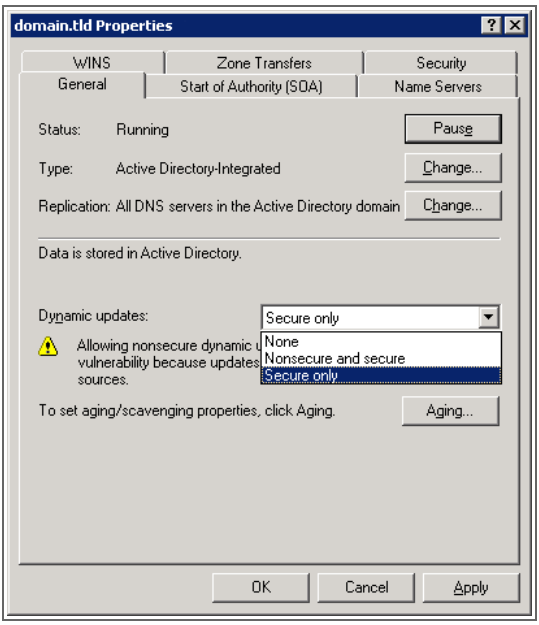

Na Microsoft DNS se ***Forwarding*** nastavuje pro určitý server na záložce ***Forwarders***. Od verze Windows Server 2003 zde můžeme nastavit ***klasický forwarding*** tak, že k položce ***All other DNS domains*** nastavíme adresy patřičných DNS serverů. A ***conditional forwarding*** tak, že přidáme novou doménu a k ní její DNS servery.

***Podmíněný forwarding*** (conditional forwarding) se od klasického forwardingu liší v tom, že přeposílá pouze dotazy, které vyhovují zadané podmínce na doménové jméno.

## Vkládání a aktualizace záznamů

**S**tandardní záznamy v DNS jsou ***statické*** (ručně vložené). Můžeme, ale také využít automatického vkládání a aktualizace záznamů. Tato metoda se označuje jako ***dynamické aktualizace*** (dynamic update). To se používá nejčastěji v součinnosti s DHCP serverem, když klient získá adresu z DHCP, tak si ji zaregistruje se svým jménem na DNS. Případně můžeme nastavit DHCP server, aby on registroval do DNS klienty, kterým přidělí adresu.

**N**a Microsoftím serveru můžeme nastavit, zda jsou dynamické aktualizace povolené a případně kdo je může provádět. Jako bezpečný (secure) se označuje klient, který je ověřený vůči Active Directory.




## Zajímavé odkazy

**T**éma DNS je mnohem širší, než co jsem zde zmínil. Nevěnoval jsem například způsobu, jakým mezi sebou komunikují DNS servery nebo klient se serverem. Nezmínil jsem speciální záznamy a subdomény, které využívá Microsoft doména. A také jsem vynechal informace o možnosti provázání DNS a WINS. Takže zde jsou nějaké odkazy, kde naleznete další podrobnosti.

- How DNS Works – velice rozsáhlý článek od obecných věcí po detaily u MS
- How Does DNS Work?
- Domain name system



Autor: Petr Bouška

**Související články:**

## Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

- OSI model [05.03.2007 08:09]
- VLAN - Virtual Local Area Network [02.06.2007 15:54]
- Počítačové sítě a jejich typy [09.07.2007 15:32]
- Počítačové sítě - základní topologie [30.07.2007 14:47]
- Ethernet - CSMA/CD, kolizní doména, duplex [03.08.2007 12:40]
- TCP/IP - model, encapsulace, paket vs. rámec [16.08.2007 16:43]
- TCP/IP - metody vysílání dat [02.09.2007 15:05]
- TCP/IP - adresy, masky, subnety a výpočty [05.09.2007 14:53]
- TCP/IP - navázání a ukončení spojení [13.09.2007 16:10]
- TCP/IP a ethernet - cesta v síti, aktivní síťové prvky [17.09.2007 10:22]
- TCP/IP - Routing - směrování [21.09.2007 15:34]
- TCP/IP - nalezení MAC adresy k IP - ARP [25.09.2007 17:35]
- Počítačové sítě - Computer Networks [30.09.2007 13:28]
- DNS (Domain Name System) zaměřeno na Microsoft [26.11.2007 14:26] ..... **právě čtete**
- Diagramy a schémata počítačové sítě (Visio) [13.01.2008 18:02]
- Wake on LAN - lokální i vzdálený subnet [10.08.2008 20:42]
- TCP/IP - Internet Protocol Version 6 - IPv6 [05.03.2009 15:41]
- TCP/IP - skupinové vysílání IP Multicast a Cisco [10.03.2009 20:05]
- Běžné útoky na switche, Cisco Dynamic ARP Inspection [18.06.2009 10:15]
- Cisco - Router Switching metody a související termíny - CAM, FIB, CEF [28.06.2009 17:15]
- Výstavba moderní sítě [28.04.2010 14:41]
- Microsoft Network Load Balancing (NLB) a Cisco switche [09.08.2010 18:26]
- Cisco Network Design - Enterprise Architecture [24.09.2010 13:47]
- VPN 1 - IPsec VPN a Cisco [10.04.2011 19:59]

# Diagramy a schémata počítačové sítě (Visio)

Neděle, 13.01.2008 18:02 | Samuraj - Petr Bouška | sítě

Grafická dokumentace sítě je velice důležitá. U malé sítě to tak nemusí vypadat, protože ten jeden admin, který ji spravuje ji má celou v hlavě nebo se dá vše jedním pohledem rozeznat. Ale jak síť narůstá, je dokumentace stále důležitější. Síť se stává složitější, na správě se podílí více lidí, do něčeho třeba musí zasáhnout externí firma nebo se mění správci. V článku budu používat pro kreslení diagramů MS Visio, které sice nepovažuji za dokonalé, ale je značně rozšířené (lze do něj získat obrázky většiny výrobců) a ve větších firmách asi není problém získat licenci.



> ***Pozn.:*** *Nevím, nakolik může být tento článek pro někoho zajímavý. Ale články pro web píšu primárně pro sebe, jako shrnutí něčeho co jsem nastudoval. A nedávno jsem dával dohromady dokumentaci k naší síti, tak jsem si to zkusil srovnat.*

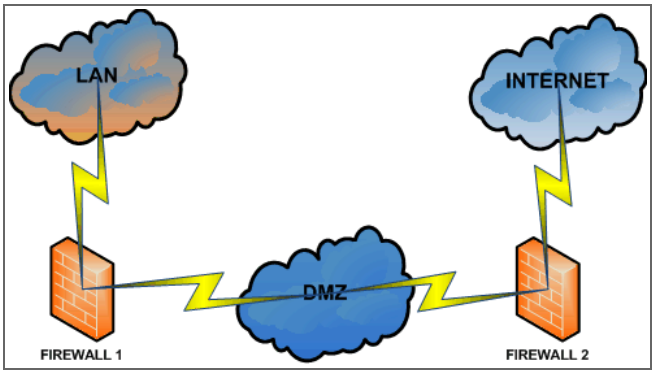
Do síťové dokumentace samozřejmě nepatří pouze ***diagramy***, ale také celá řada ***tabulek*** (třeba adresní rozsahy, pevně přidělené adresy, veřejně publikované servery, switche) apod. Nejvíce přehledné jsou však diagramy (pro obecný přehled, některé věci se však lépe hledají v tabulce) a řada věcí se prolíná (hodně textových informací se dá vložit do schématu).

Diagramů, které můžeme nakreslit, je celá řada. A i když u menší sítě nám postačí jedno jednoduché schéma, tak postupně potřebujeme další (značně odlišné) pohledy. Pokusím se ukázat několik diagramů z mé praxe.

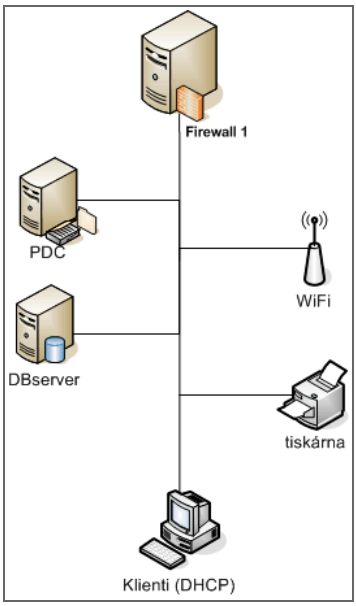
> ***Pozn.:*** *Uvedené obrázky zobrazují pouze velmi zjednodušené příklady. Chybí také řada popisků, které je důležité uvádět u jednotlivých objektů.*

## Topologie sítě

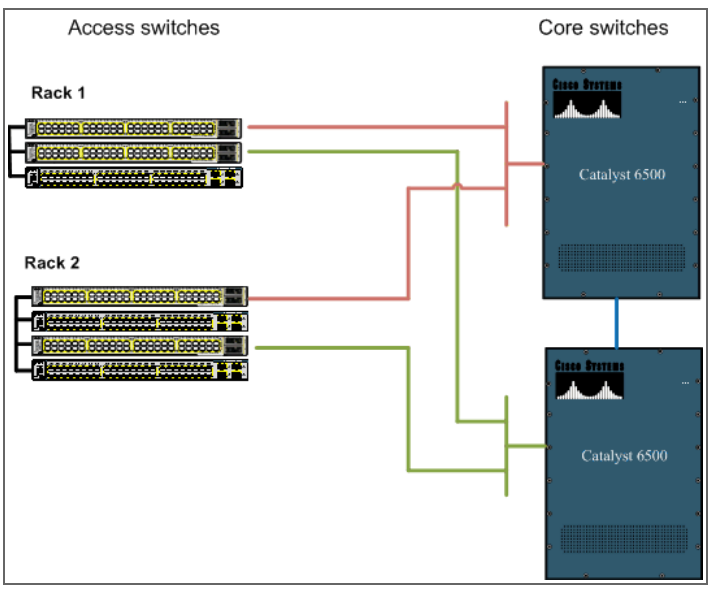
Zobrazení topologie sítě je to nejdůležitější a existuje řada pohledů, jak jej nakreslit. Můžeme začít od ***globálního pohledu***, kde zakreslíme jednotlivé sítě. Na obrázku je standardní provedení, kde je lokální síť oddělena firewallem od DMZ (demilitarizovaná zóna, kam se umísťují počítače viditelné z internetu) a DMZ oddělena druhým firewallem od internetu.




Pokračovat do ***větších detailů*** jednotlivých částí sítě. V příkladu je rozkreslena LAN (velmi jednoduše).

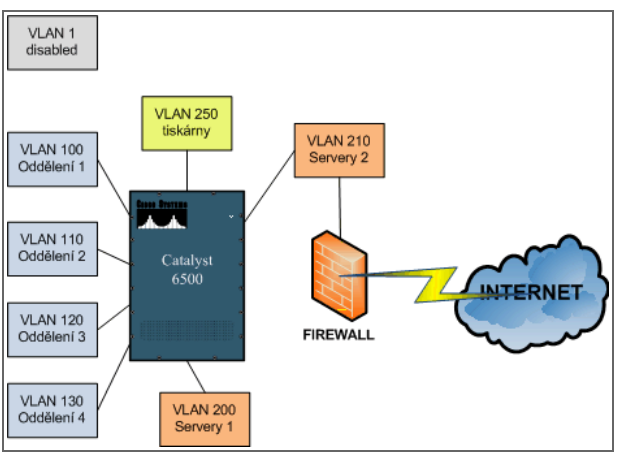



**A** dostat se až k ***logickému propojení switchů*** (které doplňuje jejich fyzické umístnění, v jiném obrázku či tabulce). Na obrázku jsou dva racky (například na dvou patrech), které připojují koncové stanice. Switche v racku jsou propojeny do stacku a celý stack je připojen ke dvěma páteřním switchům dvěma nezávislými cestami.



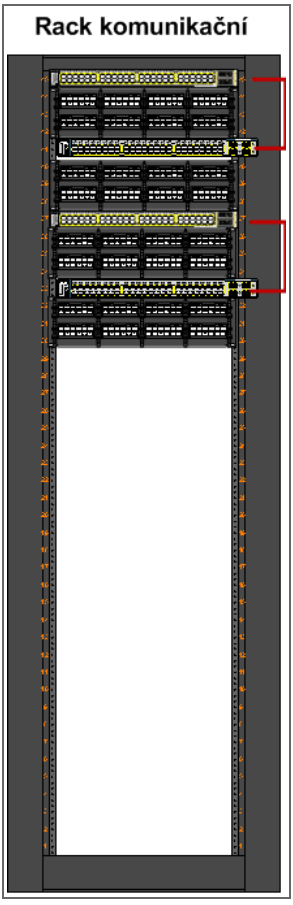

## Logické schéma VLAN

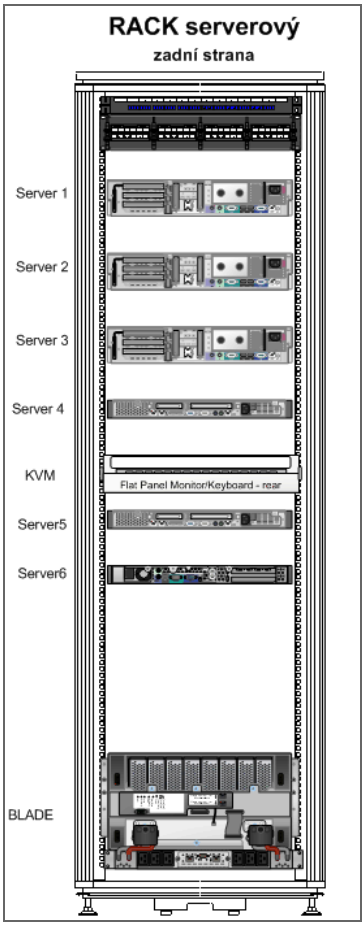
**P**ro ***VLANy*** samozřejmě potřebujeme tabulku, kde jsou uvedeny adresní rozsahy, brány apod. Ale přesto se může hodit i ***logické schéma***, které zobrazuje, jak jsou jednotlivé VLANy spolu propojeny z komunikačního hlediska. Například, které jsou routovány na switchi, kam je připojen firewall, apod.




## Obsazení Racků

***Racky*** máme několika typů, běžně ***serverové*** a ***komunikační*** (rozvodné). Občas se může hodit, mít nakreslené podrobné schéma racků, kde je zapojen který switch či server a kam vede které propojení. I když toto můžeme mít nakresleno i v logickém schématu rozmístění. Levý rack ve schématu je komunikační, slouží k propojení patch panelů (zásuvek) kam jsou připojeni klienti. Pravý rack je serverový.




## Šablony do Visia (Visio Stencils)

Výhodou je, že do Visia můžeme sehnat autentické obrázky různých prvků (serverů, switchů, apod.) přímo od výrobců. Zde jsou odkazy na některé zajímavé ***Visio Stencils*** (šablony).

- Dell
- Cisco
- Cisco ikony
- Systimax, ale odsud je odkaz k partnerskému webu NetZoom
- HP na webu VisioCafe (zde jsou schémata celé řady výrobců, třeba IBM, EMC, Hitachi, Fujitsu Siemens, Dell)
- Rittal – většinou najdeme schémata na webu Rittalu u jednotlivých produktů pod Downloads - CAD drawings – other, příkladem je třeba Serverová skříň Flex Rack
- visio.mvps.org – web několika Microsoftích MVP, obsahuje velké množství odkazů na Visio Stencils
- NetZoom – prodává Visio Stencils, dají se stáhnout nějaká dema



Autor: Petr Bouška

**Související články:**

## Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

- OSI model [05.03.2007 08:09]
- VLAN - Virtual Local Area Network [02.06.2007 15:54]
- Počítačové sítě a jejich typy [09.07.2007 15:32]
- Počítačové sítě - základní topologie [30.07.2007 14:47]
- Ethernet - CSMA/CD, kolizní doména, duplex [03.08.2007 12:40]
- TCP/IP - model, encapsulace, paket vs. rámec [16.08.2007 16:43]
- TCP/IP - metody vysílání dat [02.09.2007 15:05]
- TCP/IP - adresy, masky, subnety a výpočty [05.09.2007 14:53]
- TCP/IP - navázání a ukončení spojení [13.09.2007 16:10]
- TCP/IP a ethernet - cesta v síti, aktivní síťové prvky [17.09.2007 10:22]
- TCP/IP - Routing - směrování [21.09.2007 15:34]
- TCP/IP - nalezení MAC adresy k IP - ARP [25.09.2007 17:35]
- Počítačové sítě - Computer Networks [30.09.2007 13:28]
- DNS (Domain Name System) zaměřeno na Microsoft [26.11.2007 14:26]
- Diagramy a schémata počítačové sítě (Visio) [13.01.2008 18:02] ..... **právě čtete**
- Wake on LAN - lokální i vzdálený subnet [10.08.2008 20:42]
- TCP/IP - Internet Protocol Version 6 - IPv6 [05.03.2009 15:41]
- TCP/IP - skupinové vysílání IP Multicast a Cisco [10.03.2009 20:05]
- Běžné útoky na switche, Cisco Dynamic ARP Inspection [18.06.2009 10:15]
- Cisco - Router Switching metody a související termíny - CAM, FIB, CEF [28.06.2009 17:15]
- Výstavba moderní sítě [28.04.2010 14:41]
- Microsoft Network Load Balancing (NLB) a Cisco switche [09.08.2010 18:26]
- Cisco Network Design - Enterprise Architecture [24.09.2010 13:47]
- VPN 1 - IPsec VPN a Cisco [10.04.2011 19:59]

# Wake on LAN - lokální i vzdálený subnet

Neděle, 10.08.2008 20:42 | Samuraj - Petr Bouška | sítě

Asi každý má představu o tom, co je to Wake on LAN, tedy vzdálené zapnutí (probuzení) počítače. Když jsem ale začal přemýšlet o použití WOL na počítač, který je v jiném subnetu a je tedy mezi ním router, tak jsem si uvědomil řadu komplikací. Prostudoval jsem množství článků a diskuzí a zjistil jsem, že většinou je popis neúplný a často dokonce nesprávný. Proto v tomto článku shrnuji komplexně celou problematiku, kterou jsem nastudoval. Popisuji řešení pro probuzení počítače ve stejném, ale i vzdáleném (třeba přes internet) subnetu. A uvádím také příklad skriptu v PHP.



> *Vytvořil jsem webovou verzi aplikace, která provádí dále popsaný Wake on LAN. Dostupná je na adrese **Wake On LAN (WoL) - turning on your computer remotely** (v AJ).*

**Wake on LAN** (WOL) funguje na ***Ethernetu*** a je nezávislý protokolu (TCP/IP, IPX či jiný). Jak by název napovídal, tak buzení funguje v rámci *lokální sítě*. **LAN** dříve představoval ***jeden subnet***, takže vše bylo jednoduché a dále je popsáno v první variantě. Dnes je však *lokální síť* velmi často dělena do ***řady subnetů*** a jako LAN se označuje určitá geografická lokalita. Takže se WOL řeší do jiného subnetu, občas se označuje jako ***WOL over the internet***, a tuto situaci popisuji v dalších variantách.

> ***Pozn.:*** *V článku různě používám termíny **rámec** (frame) a **paket** (packet), často (ne vždy) se dají zaměnit, záleží na tom, zda nás v daném kontextu zajímají i MAC adresy nebo pouze IP adresy a výše. Podrobněji v článku **TCP/IP - model, encapsulace, paket vs. rámec**.*

## Co musí splňovat klient - zapnutí WOL

Nejprve začnu tím, co musí splňovat ***klient***, tedy buzený počítač, aby jej bylo možné ***vzdáleně zapnout***.

Je potřeba podpora u ***základní desky*** a u ***síťové karty*** (NIC). Funguje to tak, že i když je počítač vypnutý, tak síťová karta je stále napájena a poslouchá komunikaci, zda nepřišel speciální rámec (frame), který se jmenuje ***Magic Packet***. Síťová karta pracuje ve speciálním režimu ***Magic Packet Mode***. Ve chvíli, kdy zachytí *Magic Packet*, který je určen pro ni (podle MAC adresy, viz dále), tak pošle signál základní desce, aby zapnula počítač.

Dříve se pro ***externí síťové karty*** používal speciální kabel, který je spojoval se základní deskou. Později se tento budící signál začal posílat po ***PCI sběrnici*** (od verze PCI 2.2). Dnes máme síťovou kartu integrovanou na základní desce, takže je vše jednodušší (ale stejně to ve skutečnosti funguje po PCI či PCIe).

Většinou stačí v **BIOSu** zapnout funkci ***Wake On LAN***, která bývá nazvána různě, například na mé domácí základní desce se skrývá v ***AMI BIOSu*** pod ***Power - APM Configuration - Power On By PCIE Device***, protože integrovaná síťová karta je na PCI Express sběrnici.

Někdy je třeba ještě nastavit síťovou kartu, což můžeme udělat z Windows pod ***Control Panel - System - Hardware - Device Manager - daný síťový adaptér - Properties - Advanced*** - zde nastavení ***Wake Up Capabilities***, případně ještě ***Wake From Shutdown***.

Důležité je, že standardně musí být síťová karta přepnuta do speciálního režimu ***Magic Packet Mode*** (to se provede nastavením bitu v interním registru), kdy kontroluje síťový provoz, zda nepřišel ***Magic Packet***. To dosáhneme korektním vypnutím počítače (při zapnutí se tento speciální mód automaticky vypne), někde se tomu říká ***Soft Shutdown***. K tomu nedojde například při přerušení dodávky elektrické energie nebo při vypnutí počítače dlouhým podržením *Power Button*. Vše ještě záleží na to, zda do ***Magic Packet Mode*** přepíná ***ovladač síťové karty***, pak vše musí být nastaveno v tomto ovladači a operační systém musí být korektně ukončen. Nebo je to řešeno v ***hardwaru*** či ***BIOSu***.

> ***Pozn.:*** *To, že je síťová karta napájena i při vypnutém počítači poznáme podle toho, že má stále **link state** (svítí dioda na síťové kartě i na portu switche, na switchi se hlásí jako connected).*

## Základní princip WOL - Magic Packet

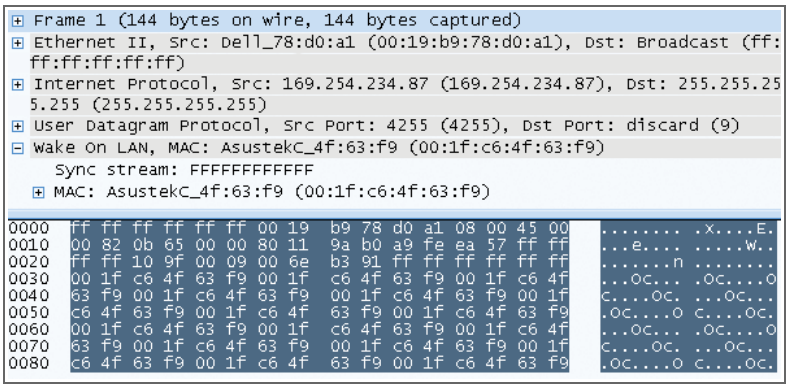
**Wake on LAN** funguje tak, že z nějaké řídící stanice odešleme speciální rámec (frame) tzv. ***Magic Packet***. ***Magic Packet*** je standardní rámec, který obsahuje ***zdrojovou adresu***, ***cílovou adresu***, která může být adresou cílové stanice nebo ***multicastovou*** (tzn. i ***broadcast***) adresou. Datový obsah paketu musí kdekoliv uvnitř (ale většinou zde jiný obsah není) obsahovat ***synhronizační stream***, což je **6 bytů o hodnotě FF**. Následovaný ***16 krát*** zopakovanou ***MAC adresou*** cílové stanice (bez oddělovače).

Rámec se odesílá jako **UDP** a doporučuje se použít port 9 discard nebo 7 echo, ale většinou se port nekontroluje, takže se může použít libovolný od 0 do 65535. Zdrojový port je dynamický.

## WOL v lokálním subnetu

### Pomocí broadcastu

Pokud jsou obě stanice (vysílající i buzená) ve ***stejném subnetu***, tak je situace jednoduchá a bezproblémové. Běžně to funguje tak, že se ***Magic Packet*** pošle jako ***broadcast***. Tedy hodnoty v rámci jsou na L2 (druhá vrstva dle OSI modelu) `zdrojová MAC = vysílající stanice`, `cílová MAC = FF:FF:FF:FF:FF:FF`. L3 `zdrojová IP adresa = vysílající stanice`, `cílová IP = 255.255.255.255`.




> ***Pozn.:*** *Z toho je vidět, že pro použití potřebujeme znát pouze MAC adresu probouzené stanice.*

> ***Pozn.:*** *Pokud bychom Magic Packet v rámci stejného subnetu chtěli poslat jako unicast, tak dojde k problému. Když fyzická vrstva síťového rozhranní vytváří rámec a nezná MAC adresu pro zadanou IP adresu (z lokální ARP tabulky), tak odešle ARP dotaz (broadcast), na který nedostane odpověď, takže rámec nikdy nevytvoří. My sice MAC adresu známe (a navíc není důležitá), ale to bychom museli vytvořit rámec sami na nižší vrstvě (než asi běžně použijeme v nějakém programovacím jazyku).*

## WOL do vzdáleného subnetu (přes internet)

Složitější situace nastane, pokud buzená stanice není ve stejném subnetu, to znamená, že v cestě se nachází jeden nebo více ***routerů***. Vše plyne z principu komunikace v síti (nebo funkce aktivních síťových prvků), podrobněji se můžete dočíst v článku TCP/IP a ethernet - cesta v síti, aktivní síťové prvky.

První věc je, že ***router*** odděluje ***broadcastové domény***, jednodušeji řečeno, broadcasty se skrze router nešíří (standardně, dá se to obejít speciální konfigurací routeru). Aplikace, které se dají získat na internetu, většinou nabízí i možnost zadání ***IP adresy*** buzené stanice (a bohužel většinou bez popisu co jak zadávat). A od toho se na internetu vede řada diskusí. Vždyť přece vypnutá stanice nemá IP adresu, síťová karta má pouze svoji hardwarovou adresu, tedy MAC adresu. Ale použití této ***IP adresy*** neslouží k identifikaci stanice, ale k ***identifikaci subnetu***.

> ***Pozn.:*** *Síťová karta v **Magic Packet Mode** se většinou nedívá na L2, L3 ani L4, ale pouze do datového obsahu rámce.*

### Pomocí unicastu

Pokud je cíl v ***jiném subnetu***, tak můžeme použít ***unicast***, ale to funguje jen chvíli (možné časy jsou uvedeny dále) po vypnutí počítače. Při odesílaní paketu naše stanice zjistí, že cíl je v jiném subnetu, takže zprávu pošle na ***bránu*** (gateway nebo podle své routovací tabulky, do rámce se použije MAC adresa brány a IP adresa cíle, to řeší fyzická vrstva síťového rozhranní). Pokud **router** má v ***ARP tabulce*** uloženu danou cílovou IP adresu, tak použije správnou MAC adresu a odešle rámec. Pokud dostane rámec **switch** a v ***CAM tabulce*** (tabulka MAC adres a portů) nemá záznam, tak pošle zprávu na všechny porty mimo příchozího. Pokud ještě záznam má, tak pošle přímo na daný port.

Problém ale nastane, když ***router*** nemá záznam v ARP tabulce. Tehdy pošle ***broadcast***, kde se ptá, kdo má danou IP adresu, ale protože nedostane odpověď (od vypnuté stanice), tak ***paket zahodí***.

> ***Pozn.:*** *V tomto případě tedy potřebujeme MAC adresu cílové stanice a její poslední IP adresu. Teoreticky bychom mohli použít i jinou IP adresu z daného subnetu, ale jde o to, aby ji router měl ve své ARP tabulce (je nám jedno, že ji na L2 přiřadí jinou MAC adresu) a naopak switch, aby tuto přiřazenou MAC adresu neměl v CAM tabulce, protože by rámec odeslal jen na jiný port. Pokud bychom za routerem měli hub, tak bychom mohli použít IP adresu libovolné běžící stanice v daném subnetu a vždy by nám to fungovalo.*

### Pomocí směrovaného broadcastu

Abychom tedy dostali paket do cílového subnetu, můžeme použít ***směrovaný broadcast na subnet*** (Subnet Directed Broadcast, např. 10.0.0.255), který používá ***boradcastovou adresu daného subnetu***, více v článcích TCP/IP - metody vysílání dat a TCP/IP - adresy, masky, subnety a výpočty. Ten by měl cestovat přes routery až na ten, který má daný subnet jako přímo připojený. Tento ***router*** upraví rámec na běžný ***broadcast*** a odešle jej na odpovídající rozhranní.

> ***Pozn.:*** *V tomto případě tedy potřebujeme MAC adresu cílové stanice a broadcastovou adresu subnetu, v kterém se nachází.*

Protože však tato technika byla používána k útokům, tak bývá na routerech nebo i firewallech většinou ***zakázána***. To se týká routeru, který by měl rámec převést na broadcast, přes ostatní by měl rámec projít normálně.

> ***Pozn.:*** *Nejběžnější je **Smurf Attack**, který má zařídit **Denial of Services** (DoS). Útočník odesílá množství paketů `ICMP echo reply` s podvrženou zdrojovou adresou jako directed broadcast, všechny stanice v subnetu odpovídají `echo reply` a stanice, jejíž adresa je ta podvržená, je zahlcena.*

Na novějších ***Cisco routerech*** a ***switchích*** (L3) je tato funkce defaultně zakázána, ale je možno ji povolit pro daný interface (port či VLANu). Následuje ukázka nastavení.

```
Switch(config)#interface vlan 100
Switch(config-if)#ip directed-broadcast   // povolení pro interface, může být následováno označením ACL
Switch(config-if)#exit
Switch(config)#ip forward-protocol udp 9  // určení povoleného protokolu a portu (volitelně) pro directed broadcast
```

***Z bezpečnostních důvodů*** se vždy doporučuje povolení směrovaných broadcastů doplnit o **ACL** (kde se povolí odesílání třeba jen z jedné IP adresy).

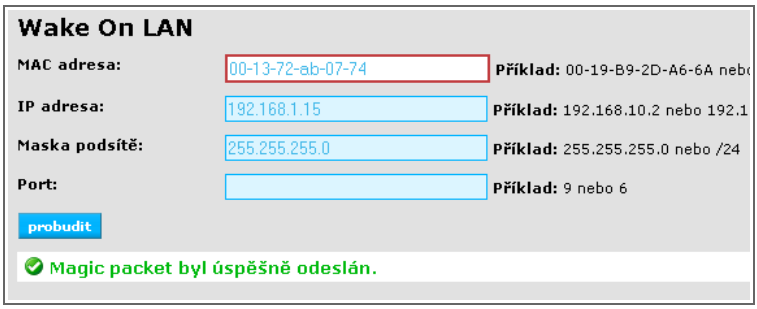
Pro zajímavost se může hodit informace o ***defaultním timeoutu*** hodnot na některých Cisco zařízeních. ***Timeout*** pro ***CAM tabulku*** bývá 5 minut, timeout pro ***ARP tabulku*** bývá 4 hodiny. Tyto hodnoty jsou nastavitelné.

## Příklad skriptu Wake On LAN v PHP

Zde je uvedený jednoduchý příklad implementace ***Wake On LAN pomocí PHP***. Pokud známe princip WOL, tak není nic těžkého takovýto skript napsat.

```
// použití WakeOnLAN("00-02-A5-31-CE-5E");
// nebo    WakeOnLAN("00-02-A5-31-CE-5E", "192.168.10.15");
// nebo    WakeOnLAN("00-02-A5-31-CE-5E", "192.168.10.255");
// vrací   true - odeslání proběhlo OK (to neznamená, že cílová stanice přijala rámec)
//         false - došle k chybě při vytváření nebo odeslání rámce
function WakeOnLAN($mac, $ip = "255.255.255.255", $port = 9) {
  $mac = strtoupper($mac);          // převedeme písmena v MAC adrese na velká
  if(ereg("^([0-9|A-F]{2})[.|:|-]?([0-9|A-F]{2})[.|:|-]?([0-9|A-F]{2})[.|:|-]?([0-9|A-F]{2})[.|:|-]?([0-9|A-F]{2})[.|:|-]?([0
    $mac = chr(hexdec($val[1])).chr(hexdec($val[2])).chr(hexdec($val[3])).chr(hexdec($val[4])).chr(hexdec($val[5])).chr(hexde
  else return false;                // MAC adresa je v neplatném tvaru
  $msg = str_repeat(chr(255), 6);   // úvodní synchronizační stream 6x FF
  $msg .= str_repeat($mac, 16);     // následovaný 16x zopakovanou MAC adresu cíle
  if(($s = socket_create(AF_INET, SOCK_DGRAM, SOL_UDP)) == false) return false;    // otevře UDP socket
  if(socket_set_option($s, SOL_SOCKET, SO_BROADCAST, true) == false) return false; // nastaví jako broadcast
  if(socket_sendto($s, $msg, strlen($msg), 0, $ip, $port) == false) return false;  // odešle Magic Packet na danou IP a UDP p
  socket_close($s);                 // uzavření socketu
  return true;
}
```

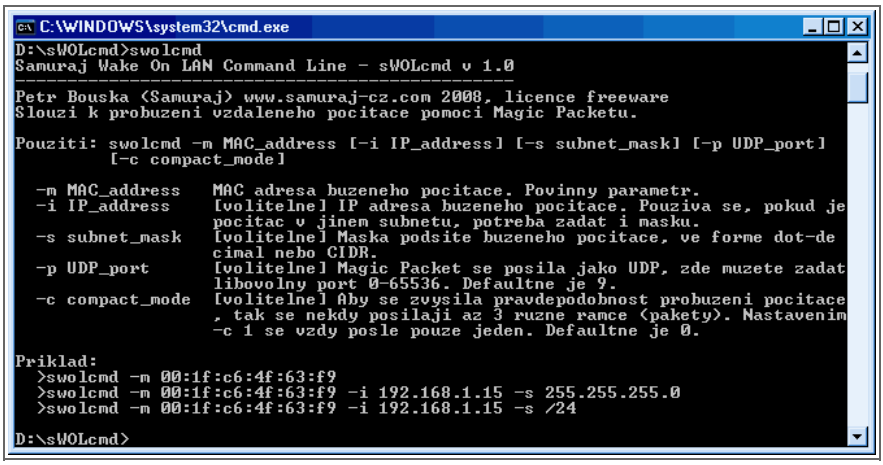
Tento kód můžeme doplnit o nějaké kontroly a vytvořit jednoduché webové rozhraní a máme aplikaci.




## Aplikace pro WOL

### Samuraj Wake On LAN Command Line - sWOLcmd

Jako první musíme zmínit aplikaci, kterou jsem vytvořil po napsání tohoto článku (a kde jsem zúročil nabité zkušenosti). Podrobný popis naleznete v Samuraj Wake On LAN Command Line - aplikace sWOLcmd. Jedná se o WOL aplikaci pro příkazový řádek a funkční v 32 bitových Windows. Stáhnout si můžete komprimovaný archiv sWOLcmd.




### Další aplikace

Aplikací, které slouží k probuzení počítače pomocí WOL, je velká řada. Ať už grafických nebo pro příkazovou řádku a pro rozličné operační systémy. Často je funkce WOL doplněna do různých aplikací pro správu síťového prostředí. Uvádím zde pár příkazů grafických aplikací pro Windows.

- Advanced IP scanner
- WOL Magic Packet Sender
- WakeOnLanGui

## Odkazy

- nejvíce informací jsem získal z ***White Paper firmy AMD*** Magic Packet Technology
- konfiguraci Directed Broadcast na Cisco Catalyst 3750 se dočtete v článku Configuring Broadcast Packet Handling



Autor: Petr Bouška



Související články:

# TCP/IP - Internet Protocol Version 6 - IPv6

Čtvrtek, 05.03.2009 15:41 | Samuraj - Petr Bouška | sítě

Internet Protocol (IP) se používá pro datovou komunikaci v přepínaných počítačových sítích, které používají TCP/IP. Jedná se o nejrozšířenější protokol na internetu i v LAN. První hlavní verze se dnes označuje jako IPv4 a stále se jedná o majoritně používanou verzi. Její hlavní nevýhodou je, že adresy jsou velké 32 bitů. Proto vznikla nová verze IPv6, která přináší řadu výhod, ale hlavní rozdíl je, že adresy jsou velké 128 bitů. IPv6 se dnes již celosvětově nasazuje.



> *Tento článek není žádným vyčerpávajícím popisem IPv6. Vznikl jako moje poznámky, když jsem se připravoval na Cisco test **642-901 BSCI**, takže i jako poznámky vypadá. Pokud někdo chcete doplnit nějakou podstatnou informaci, je vítán v komentářích.*

## Vlastnosti IPv6

- adresy jsou dlouhé `128 bit = 16 bytů`
- to znamená `3,2*10na38` adres (IPv4 cca 4 miliardy)
- integrovaný ***IPsec***
- podpora ***mobilních připojení*** (mobile IP - MIPv6)
- používají se zápis podsítí pomocí ***CIDR*** adresa/prefix

## Bezpečnost v IPv6 - IPsec

IPv4 původně neobsahovalo žádné bezpečnostní mechanismy a bezpečnost se řešila až na vyšších vrstvách (aplikačně). Až časem byl doplněn bezpečnostní mechanismus, který funguje přímo na 3. vrstvě OSI modelu, tedy na IP, a to **IPsec - Internet Protocol Security**. IPsec poskytuje ***šifrování*** a ***autentizaci***. V ***IPv6*** je jeho implementace povinná, ale nepoužívá se příliš často.

## Podpora mobility v IPv6

Další integrální součást IPv6 je mobilita. Dnes je velké množství mobilních zařízení, která se připojují do sítě z různých míst. Podpora ***mobility*** má zajistit, aby mobilní zařízení mělo adresu ze své domácí sítě a ta se neměnila ani při připojení jinde. ***Mobile IPv6*** obchází problémy s ***triangulárním routingem*** (komunikace od zdroje jde do mé domácí sítě a odtud k mému mobilnímu zařízení připojenému jinde, ale zde se provádí zabezpečeně optimalizace, aby se komunikovalo napřímo) a tudíž je efektivní jako normální IPv6. Zatím se ale MIPv6 příliš nerozšířilo.

## Příkazy Cisco IOSu pro IPv6

```
ROUTER(config)#ipv6 unicast-routing                 // zapne podporu IPv6
ROUTER(config)#ipv6 cef                             // zapne Cisco Express Forwarding (CEF) pro IPv6
ROUTER(config-if)#ipv6 address 2002:C0A8:2101::1/128 // definice IPv6 adresy
ROUTER(config-if)#ipv6 enable                       // zapne IPv6 pouze pro interface
```

## Bezstavová autokonfigurace IP adres - stateless address autoconfiguration (SLAAC)

Když je klient připojen do routované IPv6 sítě, tak se může automaticky nakonfigurovat pomocí ***ICMPv6 router discovery message***. Když se ***klient*** poprvé připojí do sítě (třeba při bootování), tak posílá ***router solicitation*** (RS) zprávu (link-local multicast) jako požadavek o konfigurační parametry, router odpovídá pomocí ***router advertisement*** (RA) zprávy. RA obsahuje 64bit prefix pro linku a lifetime. Adresa se vytvoří ze zaslaného prefixu a doplní se MAC adresa klienta (v případě Ethernetu). Pro úvodní komunikaci si klient vytvoří lokální adresu s prefixem `FF80::`.

> ***Pozn.:*** *V IPv6 existuje i stavová autokonfigurace, podobně jako v IPv4, pomocí DHCPv6.*

## Adresy v IPv6

### Formát IPv6 adres a jejich zkracování

- adresy se skládají z ***8 skupin*** po ***4 hexa číslech*** (16bitů) oddělených dvojtečkou
- `0123:BB99:3210:FE00:58A0:4565:98AE:1245`
- adresy se dají zapisovat zkráceně, počítá se, že budou existovat adresy s hodně nulami, ***úvodní 0*** ve skupině se dají vynechat (až na jednu), skupiny nulových ***16bitových bloků*** se můžeme zapsat jako `::` (dvě dvojtečky), ale může se použít pouze jednou
- `1080:0000:0000:0000:0008:0800:0000:417A = 1080:0:0:0:8:800:0000:417A = 1080::8:800:0:417A`

### Typy IP adres

- **unicast** - identifikátor jednoho interfacu na jednom nodu
- **anycast** - novinka v IPv6, stejná adresa přiřazená skupině interfaců, které typicky patří jiným nodům, paket je doručen pouze jednomu - nejbližšímu interfacu (podle daného routovacího protokolu), který je identifikován touto adresou, anycast adresa je normální unicast adresa, takže se špatně identifikuje
- **multicast** - identifikátor skupiny interfaců, které typicky patří jiným nodům, paket je doručen všem interfacům

> ***Pozn.:*** *v IPv6 není broadcast, nahrazuje se pomocí link-local all hosts multicast group*

### Základní adresy

- **::/0** je default route.
- **::1/128** loopback adresa
- **::/128** je nespecifikovaná adresa
- **FF00::/8** jsou multicast adresy
- **FC00::/7** jsou unikátní lokální adresy

### Multicast adresy

Multicast adresy začínající FF (prvních 8 bitů je 1) `FF::/8`

- **FF01::1** lokální node - všichni hosté
- **FF01::2** lokální node - všechny routery
- **FF02::1** lokální link - všichni hosté
- **FF02::2** lokální link - všechny routery
- **FF05::2** lokální site - všechny routery
- **FF02::5** všechny OSPF routery
- **FF02::6** všechny designated routery

## IPv6 paket

**U** IPv6 je snaha, aby hlavička byla co nejmenší. Některé méně užívané části byli přesunuty do ***rozšiřujících hlaviček***. Velikost hlavičky je pevná a rovná se **40B**, z čehož 32B jsou adresy. Takže neobsahuje údaj o velikosti hlavičky, také se odstranil kontrolní součet (checksum).

**I**Pv6 paket obsahuje ***verzi***, ***traffic class*** (priorita - QoS), ***flow label*** (QoS, zatím se nepoužívá), ***payload length*** (velikost dat, max 64kB, nebo se nastaví 0 a pak jde o jumbo až 4GB), ***next header*** (obdoba protokolu u IPv4), ***hop limit*** (místo TTL), ***zdrojová a cílová adresa***.

<table>
<tr><th>bits</th><th>0-3</th><th>4-11</th><th>12-15</th><th>16-23</th><th>24-31</th></tr>
<tr><td>0</td><td>version = 6</td><td>traffic class</td><td colspan="3">flow label</td></tr>
<tr><td>32</td><td colspan="2">payload length</td><td colspan="2">next header</td><td>hop limit</td></tr>
<tr><td>64</td><td colspan="5" rowspan="4">source address</td></tr>
<tr><td>96</td></tr>
<tr><td>128</td></tr>
<tr><td>160</td></tr>
<tr><td>192</td><td colspan="5" rowspan="4">destination address</td></tr>
<tr><td>224</td></tr>
<tr><td>256</td></tr>
<tr><td>288</td></tr>
</table>

***Next Header*** může také indikovat jednu z 6 ***rozšiřujících hlaviček***. V tom případě ze normální hlavičkou následuje další jedna nebo i více hlaviček se speciálními údaji. Může se jednat třeba o ***směrování*** (seznam uzlů, přes které má paket projít) a ***fragmentaci*** (linková vrstva podle technologie může přepravovat nějak max. velké rámce, Ethernet MTU = 1500B, fragmentace rozdělí větší data, u IPv6 může provést pouze odesílatel).

## Při přechodu z IPv4 na IPv6 se mohou použít techniky

### Dual Stack

Současně se používá IPv4 i IPv6, routery mají dva oddělené stacky, na doplnění je třeba NAT nebo dual-stack servery, aby se mohlo komunikovat mezi IPv4 a IPv6.

### 6to4 tunneling

Pro komunikaci izolovaných IPv6 sítí přes IPv4 páteř, kde se pakety balí do IPv4 (IP typ 41), na routerech se mohou použít tunely - manuální, GRE tunely, polo či plně automatické tunely, IPv6 adresa se vytvoří z prefixu `2002::/16` a IPv4 převedená na hexa, př: `172.31.100.1 -> 2002:AC1F:6401::/48`, hraniční router musí mít IPv6 adresu `2002::/16` v prefixu.

### IPv6 na vyhrazených linkách

Stejná L2 infrastruktura, ale oddělené Frame Relay nebo ATM PVC.

### IPv6 na MPLS páteři

Oddělené IPv6 sítě komunikují přes MPLS IPv4 páteř.



Autor: Petr Bouška

**Související články:**

### Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

# TCP/IP - skupinové vysílání IP Multicast a Cisco

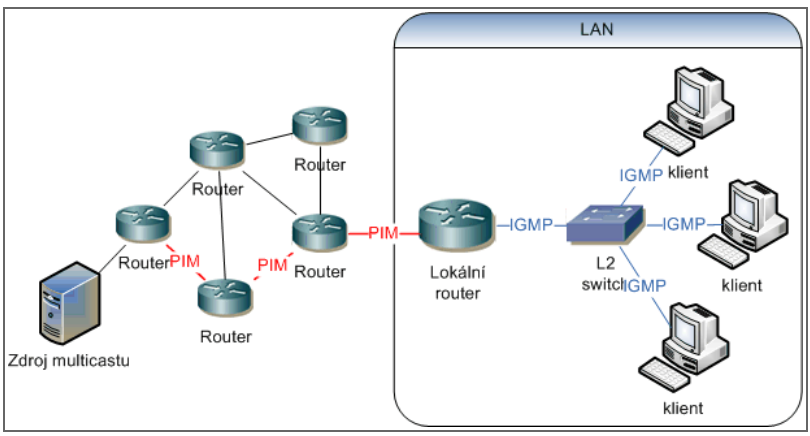
Úterý, 10.03.2009 20:05 | Samuraj - Petr Bouška | sítě

Multicast je metoda efektivní komunikace jednoho odesílatele více příjemcům. Příkladem může být internetové rádio (a přirovnání běžné rádio), kdy je jeden zdroj a mnoho příjemců, kteří přijímají stejná data ve stejnou chvíli. V praxi se to často řeší tak, že se vytvoří jednotlivá spojení pro každého příjemce. Takže je značně zatěžován server a část síťové infrastruktury je zbytečně přetížená přenosem duplicitních dat. Pomocí multicastu doručujeme informaci současně skupině příjemců co nejefektivnějším způsobem, aby zpráva přes každý síťový uzel cestovala pouze jednou, kopie se vytváří pouze, když se cesty k příjemcům rozdělují. V článku je vysvětlen obecný princip multicastu a dále se věnuje skupinovému protokolu IGMP (Internet Group Management Protocol) a směrovacímu protokolu PIM (Protocol Independent Multicast) ve všech jeho variantách.



## Jak multicast funguje

**Vysílač** (zdroj dat) odesílá pakety na ***multicastovou adresu*** (která neslouží k identifikaci příjemce, ale skupiny), zdrojová adresa je jeho normální unicastová adresa. Na ***routerech*** se pak paket odesílá do všech směrů, kde je nějaký příjemce (provádí se duplikace paketu). Mezi routery se multicast pakety přenáší pomocí multicast směrovacího protokolu (nejčastěji) **PIM**. Ten vytváří distribuční strom skrze síť a provádí přeposílání paketů. **Příjemci** (stanice) se musí zaregistrovat do ***multicast skupiny*** pomocí protokolu **IGMP**. Tok paketů je určován příjemci, detaily závisí na použitém směrovacím protokolu (případně variantě PIM protokolu).




## Multicast na síťové kartě

Na ***2. vrstvě*** OSI modelu (L2) se používají speciální ***multicast MAC adresy*** (které se vytváří z multicast IP adres). Tak jako ***síťová karta*** na nejnižší vrstvě kontroluje přicházející rámce a vyšším vrstvám posílá pouze provoz určený pro ni (s její cílovou MAC adresou nebo broadcast provoz), tak má podporu pro ***multicast*** a podle nastavení z vyšších vrstev ***filtruje multicast*** a dále posílá pouze rámce, které jsou určeny skupině, která klienta zajímá.

## Multicast v rámci jednoho subnetu - switch

V rámci běžného switche se ***multicast*** chová jako ***broadcast***, tedy jsou jím zaplaveny všechny porty. Broadcast je vlastně speciálním případem multicastu (příjemci jsou všichni na stejném subnetu). Například v IPv6 již broadcast neexistuje, ale používá se ***link local multicast***. Multicast je sice rozesílán všude v rámci broadcast domény. Ale díky tomu, že se používá cílová IP adresa, tak podle ní může klient filtrovat, zda chce paket přijmout. Na rozdíl od broadcastu, kde se klient musí dívat až na datový obsah.

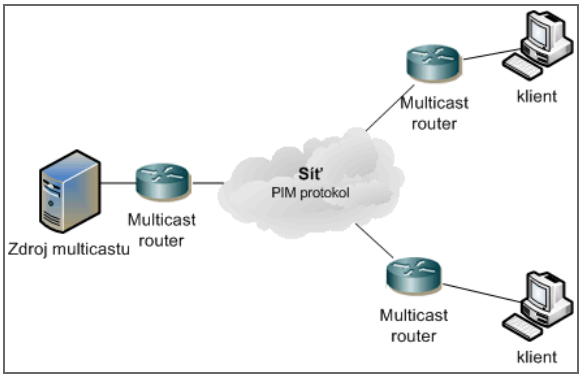
Důvod, proč se multicast chová jako broadcast v daném subnetu, i když chceme doručovat pouze určitým klientům, je následující. Na **switchi** v rámci L2 (ethernet rámce) nemůžeme zjistit, který klient (na kterém portu) chce odesílaný rámec přijmout. U unicastu je cílová MAC rámce adresou klienta a tu switch zná (má ji v CAM tabulce), ale to neplatí u multicastu, kde je cílová MAC adresa adresou skupiny. Aby mohl switch provádět lepší rozhodnutí, tak musí analyzovat veškerý multicast provoz a podle `join` a `leave` zpráv sestavovat tabulku, kam multicast směrovat. Cisco má metodu zvanou ***CGMP*** (Cisco Group Management Protocol), ale dnes se používá hlavně **IGMP Snooping**.

## Multicast mezi subnety - router

Na **routeru** se již řeší více úloh. Router musí zjistit, do kterých přímo připojených sítí má multicast odesílat. Tedy řeší přihlašování klientů do ***multicast skupin***. K tomu se používá protokol **IGMP**. Router periodicky odesílá do připojených sítí dotaz na multicast adresu `224.0.0.1` a stanice odpovídají (s různým zpožděním, aby nedošlo k zahlcení) s informací, kterou multicast skupinu chtějí přijímat (posílá se jedna zpráva pro každou skupinu), na adresu skupiny (kterou chce přijímat). Po přijetí dotazu host čeká, a pokud v dané době zaznamená odpověď jiného klienta pro stejnou skupinu, kterou chce přijímat, tak již odpověď sám neposílá (znamená to, že jsou na stejném subnetu, takže mu daný multicast provoz stejně přijde). SW na stanicích tedy musí také podporovat IGMP stejně jako router.

Druhý úkol routerů je ***optimální směrování*** multicastu do vzdálených sítí. Vytváří se minimální strom spojů, který se dynamicky aktualizuje. Dnes se používá primárně protokol **PIM**. Když si situaci představíme, tak komunikace probíhá tak, že server začne vysílat nějaký multicast provoz a ten se dostane na lokální router. Na druhé straně (třeba i v jiném státě přes internet) máme klienta, který chce daný provoz přijímat. Ten si vyjedná se svým lokálním routerem, že je členem dané multicast skupiny. A nyní se musí vyřešit ta složitá situace, aby se provoz dostal z jednoho routeru na druhý. Nejčastěji se používá jedna ze dvou variant **PIM**

protokolu. **Dense mode** vychází z představy, že téměř všichni chtějí provoz přijmout, takže jej odesílá do všech směrů (na všechny routery mimo toho, od kterého přišel). Pokud některý sousední router provoz nechce, tak to musí oznámit. Naproti tomu **Sparse mode** neposílá provoz žádnému routeru, který si o něj nepožádá.




Pro zabránění smyčkám se v multicastu používá metoda zvaná ***Reverse Path Forwarding*** (RPF), router přijímá multicast provoz pouze ze směrů, z kterých vede zpáteční cesta ke zdroji vysílání (jeho IP adresa je v každém paketu).

## Vlastnosti Multicastu

- ***IP multicast*** - komunikace jednoho několika, efektivní využití infrastruktury
- základem jsou multicastové ***skupinové adresy*** (group address), multicastový ***distribuční strom*** (distribution tree) a klientem řízené vytváření stromu
- ***vysílač*** používá skupinovou adresu jako cílovou adresu, ***příjemci*** užívají tuto adresu, aby informovali síť, že chtějí tuto komunikaci přijímat
- nějaký obsah je spojen se skupinou 239.1.1.1, vysílač odesílá data na tuto adresu, příjemci se připojí do této skupiny pomocí ***protokolu IGMP***
- pokud získá skupina nějaké členy, tak se pro ni vytvoří distribuční strom, nejčastěji pomocí ***protokolu PIM***, ten zajistí, že se pakety dostanou od vysílače ke všem příjemcům ve skupině
- vysílač nemusí vědět o příjemcích, vytvoření stromu je iniciováno aktivními prvky blízko příjemců nebo přímo příjemci, síť zařizuji doručení příjemcům
- multicast routery musí vědět o všech distribučních stromech, které prochází skrze něj, ale o ostatních neví nic, pro každý strom si vytváří záznam, ale pouze po dobu existence přenosu, nepoužívá se agregace

## Multicast IP adresy

Multicast adresní rozsahy:

- **224.0.0.0 až 224.0.0.255** - určené pro síťové protokoly uvnitř LAN, mají TTL = 1, takže nepřejdou přes router
- **224.0.1.0 až 238.255.255.255** - jsou globální multicast adresy, které se používají mezi organizacemi a přes internet
- **239.0.0.0 až 239.255.255.255** - je omezený rozsah adres, pro použití uvnitř organizace (LAN)

Některé speciální adresy:

- **224.0.0.1** - všechny systémy
- **224.0.0.2** - všechny lokální (na stejném subnetu) routery
- **224.0.0.5** - OSPF všechny routery
- **224.0.0.6** - OSPF DR a BDR routery
- **224.0.0.10** - EIGRP routery

## IGMP - Internet Group Management Protocol

- používá se mezi klientem a aktivním prvkem (routry, switche)
- IGMP slouží k registraci jednotlivých hostů do multicast skupin
- multicast router odesílá ***membership query message*** (host query message), aby objevil, jaké multicast skupiny mají členy na připojené síti, to se odesílá všem pomocí IP `224.0.0.1` s TTL = 1
- hosté odpovídají pomocí IGMP ***report message***, která značí, že chtějí přijímat multicast pakety pro danou skupinu
- ***designated router*** (designated querier) se používá u multiaccess, pro podsíť je jediný, kdo posílá ***host query message***, pro IGMP verze 1 se volí pomocí multicast routovacího protokolu, pro verzi 2 a 3 je to multicast router s nejvyšší IP adresou, Point-to-point linky nezobrazují informace o DR
- ***IGMP verze 3*** umožňuje filtrování zdrojů, tedy aby si host řekl routeru, které zdroje chce přijímat
- ***IGMP odhlašovací*** (leave) zprávy jsou IP datagramy s cílovou adresou `224.0.0.2`, TTL 1

## IGMP snooping

Jedná se o ochranný/optimalizační mechanizmus pro ***L2 switche***. Standardně se multicast na switchi šíří jako broadcast, tedy na všechny porty mimo příchozího. ***IGMP snooping*** zajišťuje zkoumání multicast provozu a detekci `join` a `leave` zpráv. Podle toho se učí, na kterém portu se nachází router a kde klienti, a sestavuje tabulku, podle které přeposílá multicast pouze na ty porty, kde klient tento provoz požaduje. Také odpovědi klientů odesílá pouze na router a ne ostatním klientům. Dynamicky tedy konfiguruje porty pro příjem multicastu.

## Multicast MAC adresa

Multicast MAC vždy začíná `01.00.5e` - rezervovaná část značící multicast. Pak vezmeme posledních ***23 bitů IP adresy***, před to dáme `0` a převedeme na hexa, to je druhá část MAC. Příklad IP `228.10.10.7` a `229.138.10.7` mohou být reprezentovány pomocí MAC `0100.5e0a.0a07`.

## PIM - Protocol Independent Multicast

- používá se mezi lokálním a vzdáleným multicast routerem pro směrování multicastu od serveru k řadě klientů
- je to skupina routovacích protokolů, které zařizují distribuci jeden-mnoha nebo mnoho-mnoha přes internet
- pro směrování používá klasické routovací protokoly

***Rendezvous Point*** (RP) - setkávací místo pro zdroje a příjemce multicast provozu (obecně známé místo pro obě strany), jedná se společný o kořen pro sdílené stromy, zdroje multicastu posílají provoz na tento bod a ten je přeposílá přes sdílené stromy všem členům skupin, díky RP se lépe využijí síťové zdroje, ale nezaručuje optimální cestu

## Sparse Mode - PIM-SM

Vychází z představy, že klienti, kteří chtějí přijímat multicast, se v síti nachází velmi řídce. Takže **Sparse mode** posílá provoz pouze routerům, kteří si o něj požádají.

Používá ***jednosměrné sdílené stromy*** s kořenem v RP a může vytvářet stromy nejkratších cest pro zdroje, vyžaduje na síti ***Rendezvous Point*** (RP). Zdroje posílají multicast přímo připojeným routerům (DR), DR (router s nejvyšší IP) je zabalí a jako unicast pošle na RP, ten je posílá členům multicast skupiny. **RP** oznamuje zdroje a vytváří cestu od zdroje ke členům skupiny, teprve potom posílá multicast datagramy.

Rozšířený routovací protokol pro multicast. Použijeme, pokud ostatní routery jsou různé. Dobře škálovatelný. Router se musí přihlásit do skupiny, aby přijímal provoz.

## Dense Mode - PIM-DM

**Dense mode** vychází z představy, že téměř všichni chtějí provoz přijmout, takže jej odesílá do všech směrů (na všechny routery mimo toho, od kterého přišel). Pokud některý sousední router provoz nechce, tak to musí oznámit.

Vytváří ***strom nejkratších cest***, používá **flood and prune** metodu (nejprve zaplaví doménu multicastem a pak ořezává větve, kde nejsou příjemci). Interfacy se přidávají do multicast routovací tabulky na routeru. Špatně škálovatelný, je ideální pro LAN, kde jsou členové hustě (densely) umístěni v síti.

## Sparse-Dense Mode - PIM-SDM

Pokud máme ***Rendezvous Point*** (RP), tak funguje jako **PIM-SM**, pokud ne, tak funguje jako **PIM-DM**.

## Bidirectional Mode - PIM-BM

Vytváří obousměrné sdílené stromy, ale nikdy ne strom nejkratších cest, takže může mít delší end-end vzdálenost, ale dobře škáluje.

## Source Specific Multicast - PIM-SSM

Vytváří stromy, které mají kořen pouze v jednom zdroji. Adresa vysílače je známá a příjemci se registrují přímo ke zdroji vysílání.

# Konfigurace multicastu v Cisco IOSu

```
ROUTER(config)#ip multicast-routing          // zapne multicast na routeru
ROUTER(config)#ip pim sparse-mode            // nastaví PIM mód
ROUTER(config)#ip pim rp-address 10.10.10.1 // definice Rendezvous Point (RP)
SWITCH(config)#ip igmp snooping              // zapne IGMP snooping pro switch
SWITCH(config)#ip igmp snooping vlan 100     // zapne IGMP snooping pro VLAN 100
ROUTER#show ip pim interface
SWITCH#show ip igmp snooping
SWITCH#show mac-address-table multicast
ROUTER#clear ip mroute *                     // vymaže IP multicast routovací tabulku
```



Autor: Petr Bouška

**Související články:**

## Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

- OSI model [05.03.2007 08:09]
- VLAN - Virtual Local Area Network [02.06.2007 15:54]
- Počítačové sítě a jejich typy [09.07.2007 15:32]
- Počítačové sítě - základní topologie [30.07.2007 14:47]
- Ethernet - CSMA/CD, kolizní doména, duplex [03.08.2007 12:40]
- TCP/IP - model, encapsulace, paket vs. rámec [16.08.2007 16:43]
- TCP/IP - metody vysílání dat [02.09.2007 15:05]
- TCP/IP - adresy, masky, subnety a výpočty [05.09.2007 14:53]
- TCP/IP - navázání a ukončení spojení [13.09.2007 16:10]
- TCP/IP a ethernet - cesta v síti, aktivní síťové prvky [17.09.2007 10:22]
- TCP/IP - Routing - směrování [21.09.2007 15:34]
- TCP/IP - nalezení MAC adresy k IP - ARP [25.09.2007 17:35]
- Počítačové sítě - Computer Networks [30.09.2007 13:28]
- DNS (Domain Name System) zaměřeno na Microsoft [26.11.2007 14:26]
- Diagramy a schémata počítačové sítě (Visio) [13.01.2008 18:02]
- Wake on LAN - lokální i vzdálený subnet [10.08.2008 20:42]
- TCP/IP - Internet Protocol Version 6 - IPv6 [05.03.2009 15:41]
- TCP/IP - skupinové vysílání IP Multicast a Cisco [10.03.2009 20:05] ..... **právě čtete**
- Běžné útoky na switche, Cisco Dynamic ARP Inspection [18.06.2009 10:15]
- Cisco - Router Switching metody a související termíny - CAM, FIB, CEF [28.06.2009 17:15]
- Výstavba moderní sítě [28.04.2010 14:41]
- Microsoft Network Load Balancing (NLB) a Cisco switche [09.08.2010 18:26]
- Cisco Network Design - Enterprise Architecture [24.09.2010 13:47]
- VPN 1 - IPsec VPN a Cisco [10.04.2011 19:59]

# Běžné útoky na switche, Cisco Dynamic ARP Inspection

Čtvrtek, 18.06.2009 10:15 | Samuraj - Petr Bouška | sítě

Tento článek pouze shrnuje základní informace o nejběžnějších typech útoků na switche. Informace, které přináší, nalezneme na mnoha místech, zde je uvádím kvůli úplnosti. Popsány jsou útoky MAC flooding, ARP spoofing, VLAN hopping a zmíněny jsou metody, jak se jim bránit na Cisco switchích. Také je rozebrána metoda obrany zvaná Dynamic ARP Inspection.



## MAC address flooding

Útok pomocí zaplavení MAC adresami. Útočník se snaží ***vyčerpat paměť switche***, určenou pro ukládání tabulky MAC adres (CAM tabulka), pomocí zasílání velkého množství rámců s unikátními (neplatnými) zdrojovými MAC adresami. Ve chvíli, kdy je CAM tabulka plná, tak se nevytváří nové záznamy. Unicastová komunikace, která je určená pro cílovou MAC adresu, která se nenachází v CAM tabulce, je ***zaslána na všechny porty*** mimo příchozího (tak se chová hub, provoz je jako broadcast).

Tento útok se používá k tomu, aby útočníkova stanice dostávala i provoz, který není určen jí. Navíc se zvýší celkový provoz v síti a ve výsledku může způsobit příliš velké vytížení switche, takže se jedná o ***DoS útok*** (Denial Of Services).

> **Obrana:** ***Port Security*** nebo ***Port Based Authentication***

## VLAN hopping

Poskakování po VLANách je útok, kdy se útočník snaží dostat k provozu, který je na ***jiné VLANě*** (kam by on neměl mít přístup). Využívá se tagování protokolem **IEEE 802.1q** a jedná se buď o metodu ***Switch Spoofing*** nebo ***Double Tagging***.

***Switch Spoofing*** spočívá v tom, že se útočníkova stanice vydává za switch a získává data z trunku, kde je přenášeno množství VLAN (nebo všechny). Může se jednat o zneužití protokolu **DTP** (Dynamic Trunking Protocol), kdy stanice vyjedná na svém portu trunk.

***Double Tagging*** odesílá rámce s přidanými ***dvěma 802.1q tagy***. Switch přijme rámec, první tag je do jeho správné VLANy, odstraní se, ale rámec se dále nekontroluje a zpracovává switchem, jako by byl v první VLANě. Pokud máme trunk port, kde je nativní VLAN nastavena stejná, jako měl port uživatele. Tak se rámec odesílá netagovaný, jenže on již tag má a ten směřuje do jiné VLANy. Druhý switch odstraní druhý tag a rámec již putuje novou VLANou.

> **Obrana:** nastavovat porty napevno jako ***switchport mode access*** (zabránit DTP), nepoužívané porty umísťovat do ***neroutované VLANy***, nepoužívat ***native VLAN*** (nastavit ji na nepoužívanou VLAN)

## ARP spoofing / poisoning

Tento útok se snaží ***přesměrovat provoz*** k útočící stanici pomocí ***falešných ARP zpráv***. Protokol **ARP** (Address Resolution Protocol) je velice starý a nemá v sobě zabudovaný žádný bezpečnostní nebo ověřovací mechanismus.

Pro komunikaci v rámci subnetu potřebuji znát cílovou MAC adresu, ale většinou mám zadánu pouze IP adresu. Proto se odešle ARP dotaz jako broadcast, kdo má danou IP. Stanice s hledanou IP adresou by měla odpovědět pomocí ARP zprávy. Klient si potom tento údaj uloží do lokální ARP cache. V ARP protokolu se také používají ***nevyžádané ARP*** odpovědi zvané ***Gratuitous ARP*** (GARP). Pomocí GARP zprávy (nebo v určité situaci normální ARP odpovědi) můžeme jednoduše podvrhnout naši MAC adresu k určité IP.

Často se tato technika využívá pro útok typu ***Man in the Middle***, kdy podvrhneme adresu brány, potom dostáváme veškerý provoz mimo náš subnet, můžeme jej dále přeposílat na standardní gateway a přitom monitorovat/upravovat veškeré informace. Podvržená informace (kombinace IP a MAC) se uloží do ARP cache klienta a CAM tabulky switche.

> **Obrana:** využití ***Private VLAN*** nebo ***ARP Inspection***

## Dynamic ARP Inspection - DAI

Jedná se o bezpečnostní funkci, která zabraňuje přeposílání ***neplatných ARP*** dotazů a odpovědí na jiné porty switche ve stejné VLANě. Tato metoda vyžaduje, aby se na switchích používal ***DHCP Snooping*** a vytvářela se ***DHCP snooping binding database***, která obsahuje přiřazení IP adres a MAC adres. Nebo aby se použily ***ARP ACL***, v kterých se tato kombinace zadává ručně.

Jednou funkcí ***DAI*** je kontrolovat veškerý ARP provoz přicházející na port a zahazovat rámce, kde informace neodpovídají uloženým kombinacím IP a MAC adres. Druhou funkcí je ***rate-limiting*** (omezování množství) ARP rámců. To je z důvodu zabránění ***DoS útokům***. Pokud počet přijatých ARP paketů překročí (defaultně) 15 za sekundu, tak se port přepne do ***error-disabled*** stavu.

DAI používá dva stavy interfaců, jedná se o **trusted** (důvěryhodný), na tomto portu se neprovádí žádná kontrola. A **untrusted** (nedůvěryhodný), kde se kontrolují všechny ARP rámce. Defaultně je DAI vypnuté pro všechny VLANy a všechny interfacy jsou ve stavu untrusted. DAI funguje per VLAN.

Kontrola pomocí DAI funguje pouze na vstupu (ingress) a může se jednat o přístupové, trunk nebo EtherChannel porty. DAI je doporučeno nastavit na všech ***přístupových*** (koncových) portech jako ***untrusted*** (nedůvěryhodné) a na ***trunk*** portech mezi switchi jako ***trusted***. Konfigurace se ovšem musí provádět s rozmyslem a důkladně. ***DHCP snooping binding database*** je lokální pro switch/stack, takže pokud by trunk port z jiného switche nebyl nastaven na trusted, tak by se vše filtrovalo. Nebo pokud na druhém switchi DAI nezapneme a trunk máme trusted, tak přes tento switch může dojít k otrávení ARP tabulky i na filtrovaném switchi.

Jako doplněk můžeme provádět i kontrolu příchozích ARP paketů, zda souhlasí zdrojová MAC adresa odesílatele s odesílatelovou MAC v ARP těle u ARP dotazů a odpovědí. Nebo cílová MAC v rámci u odpovědí. Případně i

kontrolu IP adres.

## Konfigurace DAI v Cisco IOSu

Zapnutí DAI se provede vyjmenováním VLAN, kde se má povolit.

```
SWITCH(config)#ip arp inspection vlan 100,200
```

Důvěryhodné porty musíme přepnout do trust stavu.

```
SWITCH(config-if)#ip arp inspection trust
```

Konfigurace DHCP snooping binding diabase je popsána v článku o **DHCP na switchi**. Pokud chceme použít statické záznamy, tak musíme nakonfigurovat ARP ACL.

```
SWITCH(config)#arp access-list ARPtest
SWITCH(config-arp-acl)#permit ip host 192.168.1.10 mac host 0011.70f1.e051
```

Po nakonfigurování ARP ACL jej musíme aplikovat pro určitou VLANu.

```
SWITCH(config)#ip arp inspection filter ARPtest vlan 100
```

Příkazy, které zobrazí různé informace o DAI.

```
SWITCH#show ip arp inspection interfaces
SWITCH#show ip dhcp snooping binding
SWITCH#show ip arp inspection vlan 100,200
SWITCH#show ip arp inspection statistics vlan 100,200
```

## Odkazy

- popis Port Security se nachází v článku **Cisco IOS 3 - nastavení interface/portu - access, trunk, port securit**
- Port Based Authentication, tzn. protokol IEEE 802.1x, je stručně popsán v **Cisco IOS 11 - IEEE 802.1x, autentizace k portu, MS IAS**
- konfigurace VLAN a DTP protokol - **Cisco IOS 7 - konfigurace VLAN, VTP**
- technologie Private VLAN - **Cisco IOS 19 - Private VLAN a Protected Port**
- DHCP Snooping - **Cisco IOS 13 - DHCP služby na switchi**
- popis útoků na switche z Cisco webu - **VLAN Security White Paper**



Autor: **Petr Bouška**

**Související články:**

## Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

# Cisco - Router Switching metody a související termíny - CAM, FIB, CEF

Neděle, 28.06.2009 17:15 | Samuraj - Petr Bouška | sítě

Když router nebo L3 switch přeposílá pakety (routuje), tak používá některou z metod označovaných jako Router Switching Path, od ní se odvíjí, jak rychle může pracovat. Některé z těchto metod jsou Process Switching, Fast Switching a Cisco Express Forwarding. Článek se stručně zmiňuje o těchto metodách a popisuje i související termíny, jako CAM tabulka, FIB, ARP, ale také switch, MLS a router. Některé z těchto termínů jsou podrobněji popsány ve starších článcích.



## Aktivní prvky

Následuje velice stručný popis aktivních síťových prvků, kterých se týkají dále popisované funkce.

### Switch

Aktivní prvek s řadou portů pro propojení síťových zařízení. Rámec, který přijde přes jeden port, přeposílá na jiný podle cílové MAC adresy. Vyhledává v CAM tabulce, pokud neexistuje záznam, tak se odesílá na všechny porty mimo příchozího. Ze zdrojových MAC adres si vytváří CAM tabulku. Nijak nemodifikuje rámec.

### MultiLayer Switch (MLS)

Jedná se o rozšířený switch, který mimo klasického switchování na 2. vrstvě OSI, nabízí nějaké další funkce na vyšších vrstvách OSI. Například podporuje routing, takový switch se často označuje jako L3 switch. Oproti klasickým routerům se L3 switch liší v tom, že pro přepínání paketů používá speciální HW ASIC (Application-Specific Integrated Circuit) místo procesoru. Mezi další rozšířené vlastnosti může patřit například podpora *DiffServ QoS*.

### Router

Aktivní prvek pro propojení jednotlivých sítí (subnetů). U přicházejících rámců se odstraní hlavička a při odeslání se doplní nová (v hlavičce rámce jsou vždy údaje o aktuálním hopu). Paket se směruje podle adresy síťové vrstvy (Layer 3 OSI), nejčastěji IP adresy. Využívá se routovací tabulka (RIB), která se může vytvářet dynamicky pomocí routovacího protokolu.

### Operace aktivních prvků

**Switch** provádí ***přepínání*** (***switching***) ***rámců*** (frames, dívá se na MAC adresy do 2. vrstvy, takže můžeme říci, že pracuje s rámci). Někde se používá i termín bridging. Obecně se při popisu funkce říká, že switch ***přeposílá*** (***forward***) rámce a k tomu používá některou forwardovací metodu (Store and forward, Cut through, Fragment-Free).

**Router** provádí ***směrování*** (***routing***) ***paketů*** (packets, dívá se na IP adresy do 3. vrstvy, takže pracuje s pakety, i když při odesílání opět sestaví rámec). Obecně se opět používá termín, že router ***přeposílá*** (***forward***) pakety. Ovšem někde (často u MLS) se používá i termín ***packet switching*** nebo ***router switching***.

## Tabulky u switchů a routerů

### Content Addressable Memory (CAM) table

Zjednodušeně řečeno tabulka MAC adres. Tato tabulka obsahuje MAC adresu, port, VLAN a časové razítko. Záznam vzniká při příchodu rámce a bere se zdrojová MAC adresa (pokud již existuje, tak se pouze obnoví časové razítko, pokud existuje MAC pro jiný port, tak se původní smaže). Používá se pro L2 switching. Timeout záznamů v tabulce je defaultně 5 minut.

### Ternary CAM (TCAM) table

Běžná CAM tabulka používá binární hodnoty (tedy stav 0 a 1), pomocí ní se velice rychle porovnávají adresy. TCAM používá ještě třetí stav (0, 1 a X), takže se dají porovnávat jen části řetězců. To se využívá pro rychlé HW vyhodnocení ACL u MLS (Multilayer Switch).

### Forward Information Base (FIB)

Koncepčně podobné jako routovací tabulka (Routing Information Base). Obsahuje obraz routovací tabulky a udržuje next-hop adresy. Slouží k rozhodování o IP switchování podle cílového prefixu. Je optimalizovaná pro rychlé hledání podle cílové IP adresy.

### Address Resolution Protocol (ARP) table

Tabulka přiřazení IP adres a MAC adres spolu s časovou známkou (kdy naposled byl záznam validní). Slouží pro komunikaci v rámci subnetu (L2 broadcast domain). Timeout je defaultně 4 hodiny.

## Router Switching metody

Zde se rozebírají metody, jak Router nebo L3 switch přeposílá pakety. Od dané metody se odvíjí rychlost, jakou zařízení pracuje.

### Základní princip

Rámec dorazí na interface, odstraní se hlavička rámce (ta se skládá hlavně z MAC adres) a zakončení (trailer, což je kontrolní součet). Podle údajů síťové vrstvy (L3 OSI modelu) a routovací tabulky se určí, kam se má odeslat. Ověří se, že je cíl dosažitelný a určí se next hop a interface, přes který se odešle. Znovu se vyrobí rámec (vytvoří se nová hlavička a spočítá se kontrolní součet) a odešle se přes jiný interface na next hop nebo k cíli.

Při routingu se každý přijatý rámec musí umístit do fronty, z této fronty se následně vyjme a odesílá. Jakým

způsobem detailně probíhá zpracování rámce, záleží na tzv. ***Router Switching Path***. Jednotlivé metody jsou popsány dále.

### Process Switching

Nejběžnější, ale také nejhorší metoda, kterou obsahuje každá verze IOSu na každé platformě. Pro forwardovací rozhodnutí se používá ***routovací tabulka*** (Routing Information Base - RIB) a ***ARP cache***. Paket se zpracovává (switched) normálním procesem uvnitř IOSu, který se plánuje, tedy na stejné úrovni jako jiné procesy (třeba routovací protokol). Ostatní procesy nejsou přerušeny, když se přepíná paket. Proces, který zjišťuje, jestli nějaký paket nečeká na odeslání je pravidelně naplánovaný (paket neodesílá se ve chvíli, kdy dorazí). Paket se umisťuje do bufferu, pak se čeká na přerušení CPU, které zkontroluje, zda je něco ve frontě. Takže CPU zpracovává každý paket.

### Fast Switching

Při příchodu prvního paketu se vytvoří ***záznam do cache*** (binární strom s rychlým vyhledáváním), ten obsahuje forwardovací informace a MAC hlavičku. Další pakety se pak porovnávají s touto cachí a pokud jsou ze stejného streamu, tak se rychle odešlou (nemusí se dívat do routovací tabulky). Dnes se jedná o defaultní metodu u levných routerů/switchů. Switchování se provádí na požádání (nečeká se na naplánovanou úlohu).

### Cisco Express Forwarding - CEF

Rozšířená L3 switchovací technologie. Využívá ***FIB*** (Forward Information Base) a ***tabulku sousedství*** (obsahuje L2 nebo switchovací informace mapované na záznamy ve FIB). Zvyšuje rychlost switchování rámců (výrazně rychlejší než jiné routovací techniky). Ještě lepší je distributed CEF (FIB tabulky se distribuují na každou line card). Řada funkcí (jako NBAR a QoS Policing) vyžaduje zapnutý CEF. Nové modely mají CEF defaultně zapnutý.

CEF je podobný jako Fast Switching, ale neukládá data (odchozí interface, MAC hlavička) do jedné stromové cache. Ale používá FIB pro hledání a vlastní data jsou uložena v tabulce sousedství (adjacency table), do které FIB odkazuje. Tabulku sousedství můžeme vytvářet samostatně (nemusí to být při forwardování paketů) a můžeme ji měnit. Takže nemusí docházet k invalidaci (a stárnutí) záznamů v cache a tabulku můžeme předpřipravit z hodnot v routovací tabulce a ARP cachi.

### Router Switching Path

Podle rychlosti můžeme metody seřadit (od nejrychlejší dolů):

- distributed Cisco Express Forwarding - dCEF
- Cisco Express Forwarding - CEF
- Fast Switching
- Process Switching

## Konfigurace v Cisco IOSu

Následují některé příkazy Cisco IOSu pro konfiguraci výše popsaných vlastností. I když většinou není potřeba konfigurovat nic.

```
SWITCH(config-if)#ip route-cache // zapnutí Fast Switching pro interface
SWITCH(config-if)#ip route-cache cef  // zapnutí CEF pro interface
SWITCH(config)#ip cef distributed     // zapnutí dCEF

SWITCH#show ip cef                    // zobrazí záznamy z FIB
SWITCH#show ip cef detail             // zobrazí záznamy z FIB detailně
SWITCH#show adjacency                 // zobrazí tabulku sousedství
SWITCH#show cef interface             // detailní CEF informace pro interfacy

SWITCH(config)#mac address-table aging-time 300 // nastaví CAM životnost záznamů, default 5 min, zadáním 0 se vypne mazání zá
SWITCH(config-if)#arp timeout 14400      // nastaví ARP životnost záznamů, default 4 hod

SWITCH#show mac address-table aging-time // zobrazí nastavený čas
SWITCH#show mac address-table aging-time // zobrazí nastavený čas
SWITCH#show mac address-table            // záznamy v CAM tabulce
SWITCH#show ip arp                       // záznamy v ARP tabulce
```

## Odkazy

- How to Choose the Best Router Switching Path for Your Network
- IP Switching fast switching, process switching, CEF switching
- Router Switching Methods



Autor: Petr Bouška

**Související články:**

### Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

- OSI model [05.03.2007 08:09]
- VLAN - Virtual Local Area Network [02.06.2007 15:54]
- Počítačové sítě a jejich typy [09.07.2007 15:32]
- Počítačové sítě - základní topologie [30.07.2007 14:47]
- Ethernet - CSMA/CD, kolizní doména, duplex [03.08.2007 12:40]
- TCP/IP - model, encapsulace, paket vs. rámec [16.08.2007 16:43]
- TCP/IP - metody vysílání dat [02.09.2007 15:05]
- TCP/IP - adresy, masky, subnety a výpočty [05.09.2007 14:53]
- TCP/IP - navázání a ukončení spojení [13.09.2007 16:10]
- TCP/IP a ethernet - cesta v síti, aktivní síťové prvky [17.09.2007 10:22]
- TCP/IP - Routing - směrování [21.09.2007 15:34]
- TCP/IP - nalezení MAC adresy k IP - ARP [25.09.2007 17:35]
- Počítačové sítě - Computer Networks [30.09.2007 13:28]
- DNS (Domain Name System) zaměřeno na Microsoft [26.11.2007 14:26]
- Diagramy a schémata počítačové sítě (Visio) [13.01.2008 18:02]

# Výstavba moderní sítě

Středa, 28.04.2010 14:41 | Samuraj - Petr Bouška | sítě

Návrh a realizaci rozsáhlejší počítačové sítě asi nebudeme provádět vlastními silami, ale svěříme nějaké specializované firmě. Přesto je dobré mít minimálně základní přehled o této oblasti a ideálně detailní znalosti, aby výsledná síť odpovídala našim představám a ne pohledu realizátora. Podíváme se tedy na oblasti a problémy, které musíme při návrhu sítě vzít v úvahu.



*Tento článek jsem napsal pro časopis Connect a vyšel v čísle Connect 04/10, zde jej publikuji s laskavým svolením redakce.*

## Strukturovaná kabeláž a aktivní prvky pro rozsáhlejší síť

Celá tato problematika je značně rozsáhlá a variabilní. Vše záleží na našich požadavcích, jaké parametry má výsledná síť splňovat. A také, aby pro nás byla pohodlná správa a další rozšiřování. Každému je určitě jasné, že jiné požadavky na síť má výrobní závod, právnická firma či IT firma (která vyvíjí software či poskytuje školení). A samozřejmě také záleží na rozsahu, tedy počtu připojených stanic, serverů a dalších zařízení, stejně jako na tom, jakou oblast má zabírat geograficky.

Z výše uvedeného je jasné, že není možno sepsat univerzální kuchařku na návrh počítačové sítě. Je třeba vycházet ze znalostí problematiky a zkušeností. Můžeme použít určitá obecná doporučení či návrh velkých firem vyrábějících aktivní prvky, ale nepovažuji to za dogma. Důležité je vše přizpůsobit potřebám a aktuálnímu prostředí. Jedním z rozhodujících parametrů je také poměr cena/výkon.

V tomto článku uvedu svoje praktické zkušenosti z návrhu, budování a provozování rozsáhlejší sítě. Pokusím se uvést i další možnosti v jednotlivých oblastech, než jsou ty použité v našem projektu, ale výčet určitě nebude vyčerpávající. Aktuální síť byla navrhována pro IT firmu, která má větší nároky na síťové prostředí, v síti je zapojeno přes 150 serverů, více než 400 stanic a desítky dalších síťových zařízení. Co se týká rozlohy, tak se jedná pouze o centrálu, tedy jednu několikapatrovou budovu. Budeme se zabývat strukturovanou kabeláží a aktivními prvky.

## Soupis požadavků

Na začátku je jistě třeba sepsat hlavní požadavky na výslednou síť. Jako hlavní jsou asi výkonnostní parametry a dostupnost, jinak řečeno rychlost portů (a přepínacího/směrovacího HW) a redundance. Také je třeba specifikovat požadavky na porty, tedy počet zásuvek na pracoviště, umístění speciálních zásuvek, z toho nám vyplyne celkový počet portů a hustota portů na lokalitu (místnost/patro/budova).

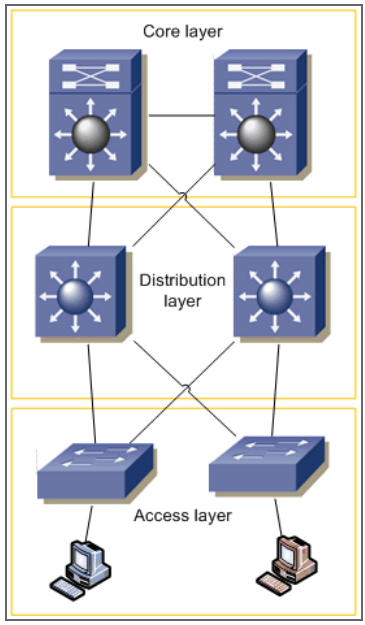
Dále můžeme již na začátku určit další preference. Například preferovanou značku aktivních prvků (ať již z důvodu, že máme vyškolené správce, či dalších), nároky na komunikační a serverové rozvaděče, fyzickou topologii sítě, napájení portů, apod. Jednotlivé body budeme řešit dále v textu.

V určitých místech musíme návrh rozdělit na dvě části a posuzovat je odděleně. Jednou oblastí je připojení koncových stanic a síťových zařízení. Druhou je datové centrum a připojení serverů (zde budeme řešit pouze LAN síť a nebudeme se věnovat SAN a dalším).

## Struktura sítě

Když se na síť podíváme z pohledu aktivních prvků, tak se často mluví o **hierarchickém designu**, který má tři vrstvy. Centrální nejvýkonnější část se označuje jako jádro (core layer), pod ním se nachází distribuční (distribution layer) a nejníže přístupová (access layer) vrstva. Takovýto design má řadu výhod, hlavně se jedná o jeho přehlednost a rozšiřitelnost.

V praxi se ovšem častěji setkáme pouze s ***dvouvrstvým modelem***, protože pro středně velké firmy v našem prostředí je naprosto dostačující. Takže i my budeme uvažovat použití **jádra**, které bude zařizovat rozdělení sítě na VLANy, routovaní mezi nimi, kontrolu provozu (uplatnění ACL) a propojení switchů z přístupové vrstvy. K tomu druhá vrstva bude **přístupová**, která má vysokou hustotu portů, připojuje klienty a zajišťuje bezpečnost na úrovni portu. Do této vrstvy patří i přístupové body bezdrátové sítě. Protože máme dvojvrstvý model, tak i servery budeme připojovat do přístupové vrstvy, ale ta může být v datovém centru výkonnější.




**N**a začátku se musíme rozhodnout, jakou rychlost připojení budeme poskytovat jednotlivým uživatelům, tedy jak rychlé budou porty na přístupovém switchi. V dnešní době se dá uvažovat mezi mnoha lety prověřenou a používanou 100 Mbps nebo moderní a dnes již široce podporovanou rychlostí 1 Gbps (asi všechny dnešní stanice i servery jsou vybaveny síťovou kartou této rychlosti). Protože vybavujeme IT firmu, která pomýšlí na budoucnost, tak jednoznačně volíme ***gigabitovou rychlost***.

**K**omunikace mezi jednotlivými switchi se označuje jako páteř. Logicky by tato komunikace měla mít větší rychlost než koncové porty, aby se uplink port přístupového switche nestal úzkým hrdlem. Dnes s přihlédnutím k cenám nemáme příliš možností a volíme rychlost páteře ***10 Gbps***. Jinou možností by bylo použít agregaci více gigabitových portů.

**D**o obecné struktury sítě jistě patří i připojení k internetu (nejlépe ke dvěma nezávislým poskytovatelům), jak bude provedeno, jak se napojí do sítě, apod. Na tuto oblast, zde ale nemáme prostor.

## Strukturovaná kabeláž

**N**a základě předchozích odstavců musíme zvolit strukturovanou kabeláž jak pro páteřní spoje, tak pro klientské stanice a servery. U klientů se hned rozhodneme pro **metalické kabely UTP** (nestíněná kroucená dvojlinka) se standardními RJ-45 konektory. Nyní ovšem záleží na kategorii kabeláže. K dispozici pro gigabitovou rychlost máme, v současnosti stále nejrozšířenější, kategorii 5e, která má nejpříznivější cenu. Dále novější a lepší kategorii 6 (případně novou vylepšenou 6a) a ještě kategorii 7, která ovšem vyžaduje speciální konektory.

**V** našem projektu jsme se rozhodli investovat do lepší ***kategorie 6***, která podporuje do budoucna i 10Gbps ethernet. A v současnosti se jedná o nejrozšířenější možnost pro nové budovy. Jako protokol volíme standardní ***1000Base-T***. Následně je ještě třeba vybrat vhodného výrobce kabeláže.

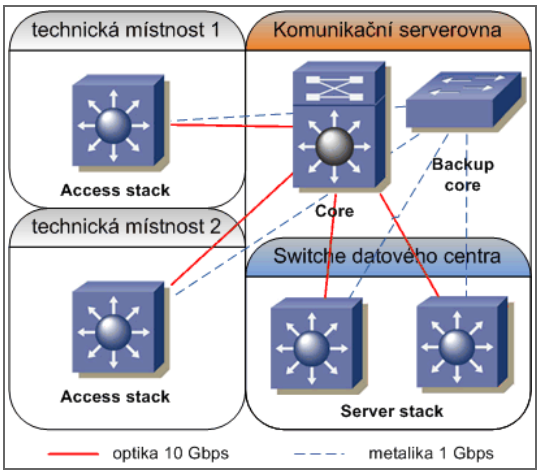
**K**dyž se podíváme na připojení serverů, tak zatím stále připojení pomocí optiky (neuvažujeme SAN síť) nebo 10 Gbps je extrémně nákladné a používané málo. Takže použijeme stejné řešení jako pro koncové stanice, rychlost 1 Gbps a kabeláž kategorie 6.

**Z**bývá nám rozhodnout o propojení jednotlivých switchů. Rozhodli jsme se pro rychlost 10 Gbps, pro kterou dnes existuje protokol 10GBase-T, který přenáší 10 Gbps na metalickém kabelu, ale pro páteř se převážně používají **optické kabely**. Ty jsou navíc odolné proti elektromagnetickému rušení. K tomu budeme používat protokol ***10GBase-SR***.

## Rozvody pro stanice

**M**ezi strukturovanou kabeláž patří i další prvky. Jasné jsou ***patch kabely***, které se používají pro připojení koncových zařízení nebo propojení na patch panelu. Dále telekomunikační ***zásuvky***, které se umísťují na zeď (přímo nebo do parapetního kanálu) či do podlahy (z praxe doporučuji použít všude, kde je to možné, zásuvky na zeď). A ***patch panely*** (pro optiku i metaliku), které jsou umístěny v rozvaděči (racku) a na nich jsou zakončeny kabely, které vedou ze zásuvky u uživatele. Všechny tyto prvky musí dohromady splňovat zvolenou kategorii 6.

**D**ůležité je také rozhodnutí, jak bude vypadat fyzická topologie kabeláže a aktivních prvků. Možnost první je centrální místo, do kterého budou svedeny veškeré rozvody z celé budovy. Výhodou je, že máme všechny aktivní prvky na jednom místě. Ale pro větší sítě je to nerealizovatelné. Proto musíme zvolit druhou možnost, kdy máme pro každé patro **technickou místnost**, kam jsou svedeny rozvody daného patra. V této místnosti máme rozvaděč (rack), který obsahuje patch panely a switche pro dané patro. Zde administrátor připojuje zásuvky uživatelů do sítě. Následně máme jedno centrální místo, **komunikační serverovnu**, kam jsou vyvedeny uplinky z přístupových switchů jednotlivých pater, a kde se nachází aktivní prvky vrstvy jádra.




**Z** praktického pohledu je důležité rozmyslet i to, jakým způsobem budou umístěny prvky v komunikačním rozvaděči. Běžná přístup je umístit do jedné poloviny patch panely a do druhé switche. Pak ovšem vzniká velký zmatek u vedení patch kabelů, kdežto když proložíme vždy patch panel a switch, tak můžeme použít krátké patch kabely a dosáhneme většího pořádku.

## Rozvody pro servery

**T**rochu odlišnou oblastí je připojení **datového centra**, kde jsou umístěny servery. Do každého serverového rozvaděče můžeme umístit switch nebo pouze patch panel a switche koncentrovat na jiném místě v komunikačním rozvaděči (podobně jako u stanic).

**M**y jsme zvolili možnost, kdy v komunikační serverovně jsou umístěny další rozvaděče, kam jsou vyvedeny patch panely ze serverových rozvaděčů a umístěny switche pro servery. Uplinky z těchto switchů jsou vyvedeny do prvku jádra, stejně jako uplinky z jednotlivých technických místností. Toto řešení nám dovolí, v případě potřeby, dovést k severu i komunikaci, která neprochází přes switche (pomocí propojení přes jednotlivé patch panely).

## Aktivní prvky

**V** našem projektu bylo již dopředu rozhodnuto, že budeme volit aktivní prvky firmy Cisco. Z našich předchozích rozhodnutí plyne, že potřebujeme aktivní prvky na přístupové vrstvě, které mají gigabitové metalické porty plus desetigigabitový optický uplink. Centrální jádro musí zvládat 10 Gbps optické linky. Také jsme spočítali, kolik zásuvek budeme chtít v budově umístit, a tedy kolik portů budeme potřebovat.

**P**řed pár lety, kdy se síť navrhovala, tak byla pouze jediná možnost pro přístupové switche, které by splnili dané parametry. Jednalo se o **Catalyst 3750E**, který může mít dva ***X2 porty***, které můžeme osadit 10 Gbps X2 modulem. Kdybychom se rozhodli slevit z některých parametrů a zvolily například agregaci 5 gigabitových portů do uplinku, tak bychom mohli použít výrazně levnější (a pro tento účel běžné) switche Catalyst 2960G. V dnešní době již máme k dispozici novou řadu ***Catalyst 2960S***, která obsahuje 2 ***SFP+ porty***, které můžeme osadit 10 Gbps modulem SFP+.

**N**ávrh sítě počítal také s nasazením IP telefonie a bezdrátové sítě, takže switche, které jsou umístěny na patrech, jsou modely s napájenými porty (Power over Ethernet). Switche, které slouží k připojení serverů, tuto vlastnost nepotřebují, takže se volí levnější modely bez PoE.

**Z**e stanoveného množství zásuvek nám vychází použití pěti až šesti 48-portových switchů na patro. Dalo by se uvažovat i použití dvojnásobného množství 24-portových switchů, ale tuto otázku zde nebudeme rozebírat. Šest switchů představuje vyvedení šesti optických kabelů z každého patra do centrály a také odpovídající množství portů na centrálním prvku. Ale díky tomu, že jsme zvolily řadu switchů Catalyst 3750E, tak máme k dispozici možnost ***stohovat*** (využít stack) tyto přepínače. To znamená, že se propojí speciálním kabelem o vysoké rychlosti. Zjednoduší se tím správa, celý stack se chová jako jeden switch, a můžeme použít méně uplink tras.

**P**řístupové přepínače jsme již určily a nyní musíme rozhodnout a centrálním prvku. Vzhledem k počtu portů, výkonu a požadavkům se nabízí modulární **Catalyst 6500-E**. Jedná se o velice výkonný, ale také nákladný prvek. Nebudeme zde již rozebírat volbu modelu, supervizoru (řídícího prvku) ani použitých modulů.

## Dostupnost – redundance

**Z**atím jsme neřešili dostupnost celé sítě, tedy co se stane, když vypadne některý prvek nebo nějaká cesta. Vyšší dostupnosti dosáhneme redundancí jednotlivých komponent. Zdvojení všeho by ovšem znamenalo také dvojnásobnou cenu, což se nám určitě nelíbí. Takže musíme najít kompromis mezi cenou a dostupností.

**K**dyž půjdeme od shora dolů, tak nejprve máme nejdůležitější prvek a to jádro. Nejlepší řešení, ale také nejvíce nákladné, je použít dvakrát Catalyst 6500 a všechny linky vést dvakrát po jedné do každého switche. Druhou možností je využít toho, že řada 6500 je modulární a připravena pro vysokou dostupnost. Veškeré prvky můžeme zdvojit (napájení, supervizor i moduly s porty). Tím ale ušetříme pouze vlastní šasi oproti řešení s dvakrát 6500. My jsme se rozhodli počítat s tím, že 6500 je velice kvalitní model a vybudovat pouze levnější náhradní řešení pro případ nouze. Tedy nepoužít srovnatelně výkonný prvek, ale pouze Catalyst 3750, který je pasivní a přebral by roli centrálního prvku pouze v případě výpadku 6500ky.

**J**ako záložní páteř jsme také nevolili optiku s rychlostí 10 Gbps, ale metaliku na rychlosti 1 Gps. Navíc z jednotlivých pater, kde je stack o šesti přepínačích, jsme vyvedli dvakrát primární optiku a dvakrát záložní metaliku.

**U** koncových stanic není dostupnost kritická, takže v případě nějakého výpadku na přístupové vrstvě se provede mechanické přepojení. U serverů je situace důležitější, takže se používá metoda zvaná ***dual-home***, server je připojen přes dvě síťové karty ke dvěma různým switchům. Navíc nemusí být připojení pouze active/passive, ale můžeme použít agregaci linek (v Cisco terminologii Etherchannel) a zvýšit i rychlost.

# Microsoft Network Load Balancing (NLB) a Cisco switche

Pondělí, 09.08.2010 18:26 | Samuraj - Petr Bouška | Microsoft admin

NLB slouží k tomu, co již říká název, vyvažování zátěže na více serverů, jinak můžeme říci, vytvoření clusteru. Celý princip je značně jednoduchý. Výhoda (ale z toho plynou i nevýhody) je, že se jedná o softwarové řešení a obejdeme se bez HW load balanceru. Nebudeme se zde věnovat vlastní konfiguraci NLB (ta je mimochodem docela triviální pomocí průvodce), ale tomu, jak se NLB chová v síti (speciálně s Cisco prvky).



Testování jsem prováděl na MS Windows Server 2008 R2 a switchi Cisco Catalyst 2960G. Zajímavý článek o této problematice je Catalyst Switches for Microsoft Network Load Balancing Configuration Example. NLB je volitelná Feature na serveru 2008, pro správu se používá ***Network Load Balancing Manager*** (z libovolného stroje s dostupným spojením). Při testech jsem pouštěl NLB Manager z členů clusteru a několikrát jsem se setkal s tím, že z jednoho serveru nebyli vidět všichni členové, ale z druhého ano.

NLB zařizuje ***distribuci požadavku*** klienta na skupinu serverů. Odpověď pak přichází od člena clusteru, který požadavek zpracoval. Hlavní roli zde hraje virtuální IP adresa, která se označuje jako **cluster IP address**. To je adresa, kterou pak používáme pro volání aplikace na clusteru. K této IP adrese se automaticky generuje MAC adresa, její tvar záleží na modu clusteru. Členové clusteru by měli (nejsem si jistý, zda je to vždy povinná podmínka) být ve stejném subnetu.

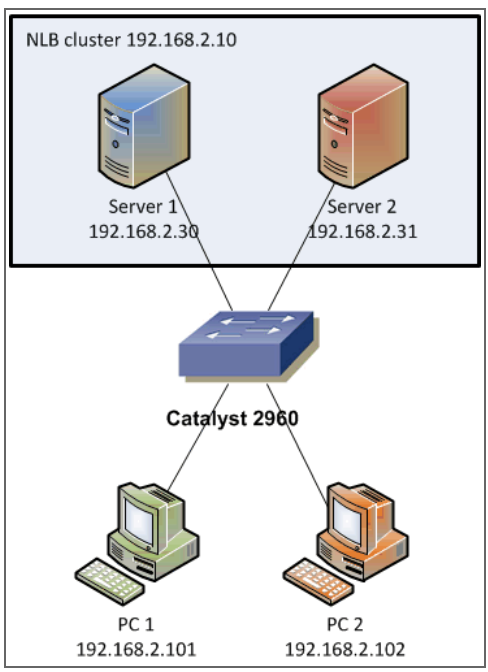
Při konfiguraci NLB se automaticky upraví nastavení adaptéru, který má zvolenou IP adresu (podle ní se přidává do clusteru). A k jeho standardní IP adrese se přidá jako druhá cluster IP. Takže všichni členové clusteru získají stejnou IP adresu (jako druhou). Požadavek klienta pak na L3 (3. vrstva dle OSI modelu) dorazí zároveň na všechny servery a ty si již podle určitých pravidel určí, který z nich bude dotaz zpracovávat. Rozdělení se provádí podle spočítaného hashe (může být založen na zdrojové IP adrese) a podle počtu serverů v clusteru mají rozsah hashe rozdělen mezi sebou.

Členové clusteru musí mít navíc přehled o tom, kolik dalších aktivních členů se v clusteru nachází. Aby si správně rozdělili příchozí komunikaci a zpracovali ji. Proto všichni pravidelně odesílají ***NLB heartbeat*** (součástí je cluster virtual IP a host IP).

NLB je clusterovací technologie MS a je součástí Windows Server 2000, 2003, 2008. Může pracovat v jednom ze tří módů ***unicast***, ***multicast*** a ***IGMP multicast***. Důležité je, že NLB může používat na komunikaci mezi členy clusteru stejný síťový adaptér, který je clusterovaný.

Hodit se může informace, že ***konfigurace NLB*** (včetně IP a MAC adres) se ukládá do registrů do klíče `HKLM\SYSTEM\CurrentControlSet\Services\WLBS`. To je asi z historických důvodů, protože WLBS je zkratka *MS Windows NT Load Balancing Service*, což byl předchůdce NLB.

Pro testy popsané dále bylo použito následující zapojení. Server 1 a Server 2 jsou použity pro vytvoření NLB clusteru, PC1 slouží k posílání dotazů a PC2 je pouze pro monitoring toho, že se některá komunikace šíří i mimo porty kam je směrována.




## NLB Unicast mode

Defaultní mód pro NLB, údajně je s ním nejméně problémů. NLB přidá na adaptér všech členů clusteru ***druhou IP adresu*** (cluster IP) a zároveň ***změní MAC adresy*** těchto adaptérů. Nová MAC adresa je z rozsahu *"běžných NLB MAC adres"* a je společná pro všechny členy clusteru i pro virtuální IP (v mém testu šlo o MAC `02:bf:c0:a8:02:0a`). Díky tomu by měla komunikace na L2 dorazit zároveň všem členům clusteru. Jenže, na jednom switchi nemohou mít dva porty přiřazenu stejnou MAC adresu, takže v praxi toto řešení nefunguje (na switchi by se pořád měnil záznam, který port má tuto MAC a vždy by fungoval pouze jeden = MAC address flapping).

NLB to řeší tak, že defaultně provádí ***maskování MAC adres*** v Ethernet hlavičce odchozích rámců. To jsou ty

adresy, které se switch učí a ukládá do CAM tabulky. NLB vytvoří falešné rozdílné MAC adresy (`02:01:c0:a8:02:0a`, `02:02:c0:a8:02:0a`), které vychází z cluster MAC adresy, ale v druhém oktetu má pořadové číslo člena clusteru. Komunikace pak probíhá tak, že v hlavičkách se používají tyto falešné MAC adresy, ale na ARP dotazy odpovídají servery tou virtuální MAC adresou. V ARP reply pak máme jinou MAC adresu v hlavičce a jinou MAC adresu v datech jako zdrojovou, což mohou některá zařízení vyhodnotit jako podvržení.

Obecně to ale funguje, switch používá MAC adresy z hlaviček rámců, klienti používají údaje z ARP. Klienti tedy odesílají data na virtuální MAC adresu. Tu switch nezná (nepřišel z ní žádný rámec), takže rámec posílá na všechny porty v dané VLANě mimo příchozího, to je stejné jako ***broadcast***.

Zajímavé ještě je, když se přes ARP ptá některý člen clusteru na MAC adresu jiného člena, tak používá opět jinou MAC adresu a podobnou dostane v odpovědi (`03:bf:c0:a8:02:1e`, `03:bf:c0:a8:02:1f`). To je pro přímou komunikaci mezi členy clusteru. Další zajímavostí je, že když provedeme testování pomocí ICMP echo/reply (ping), tak na každý request dostaneme odpověď od všech členů (na této vrstvě se ještě nerozděluje zpracování požadavků).

Pokud použijeme NLB v unicast modu, tak se nám ***MS NLB heartbeat*** šíří pomocí ***broadcastu*** (cílová MAC adresa je `FF:FF:FF:FF:FF:FF`) a odesílá se každou sekundu.

### Vylepšení pomocí konfigurace switche

Výše popsané řešení je sice funkční, ale to, že nám komunikace probíhá vlastně broadcastem, určitě není dobré a pro mnohé je naprosto neakceptovatelné. Naštěstí existuje řešení, jak toto chování změnit. Když máme Cisco switch, tak můžeme ***záznam do CAM tabulky*** provést staticky a dokonce je možné jednu MAC adresu přiřadit více portům. Tím zajistíme, že komunikace odesílaná na cluster (virtuální MAC), dorazí pouze na zadané porty.

```
SWITCH(config)#mac address-table static 02bf.c0a8.020a vlan 112 interface Gi0/3 Gi0/17
SWITCH#show mac address-table | inc Gi0/3
112    0201.c0a8.020a    DYNAMIC     Gi0/3
112    02bf.c0a8.020a    STATIC      Gi0/3 Gi0/17
```

### Aby nedocházelo k maskování MAC adresy v unicast modu

Nyní již vypadá maskování MAC adres jako zbytečné, takže bychom mohli přepnout defaultní chování do režimu, kdy se i na L2 používá clusterová MAC. Toto přepnutí se provádí v registrech na všech členech clusteru, musíme změnit hodnotu `MaskSourceMAC` na 0 (0 je vypnuto, 1 zapnuto) v cestě `HKLM\SYSTEM\CurrentControlSet\Services\WLBS\Parameters\Interface\Adapter-GUID`. Následně je třeba provést restart serveru.

Bohužel, to že zadáme MAC adresu staticky na více portů, nám nevyřeší problém s MAC flappingem. Switch se pořád učí MAC adresy na portech (i když tu stejnou má staticky zadanou) a přestože komunikace funguje, tak se vytěžuje switch.

## NLB IGMP Multicast mode

Multicast mode nám MS nabízí ve dvou typech, jako ***nestandardní multicast***, který používá pro cluster IP Microsoftí NLB multicast MAC adresu. Tomu se zde věnovat nebudeme. Nebo ***standardní IGMP multicast***, který používá standardní multicast MAC adresy (začínají `01:00:5e`) a také *IGMP Report Message* (zprávy o zařazování do multicast skupin).

Při multicast módu zůstává fyzickým adaptérům ***originální MAC adresa***, na adaptér se přiřadí virtuální IP adresa a na ARP dotazy na cluster IP odpovídají servery multicastovou MAC adresou (`01:00:5e:7f:02:0a`). Když klient odesílá na tuto MAC adresu požadavek, tak switch odesílá rámec na ***všechny porty*** mimo příchozího v této VLANě (standardní chování multicastu v rámci subnetu, takže opět jako broadcast). V případě funkčního ***IGMP Snoopingu***, zkoumá switch *IGMP Report Message* a pak posílá data pouze na porty, které jsou přihlášeny k multicast skupině.

POZOR! V tomto případě se může objevit problém, pokud potřebujeme, aby s cluster IP adresou komunikovali i klienti, kteří jsou v jiné VLANě. **Cisco zařízení nepovolí ARP odpovědi, kde je unicast IP a multicast MAC.** Takže cluster IP není dostupná pro klienty z jiných VLAN. Řešením jsou **statické záznamy do ARP tabulky** routeru.

```
SWITCH(config)#arp 172.16.63.241 0300.5e11.1111
```

Pokud použijeme NLB v IGMP multicast modu, tak se nám ***MS NLB heartbeat*** šíří pomocí ***multicastu*** na multicast MAC adresu clusteru (`01:00:5e:7f:02:0a`) a odesílá se každou sekundu. Členové clusteru používají v odpovědi svoji MAC adresu. Zajímavé je, že při zachytávání komunikace na členu clusteru se v rámcích odesílaných na cluster IP a MAC zobrazuje místo multicast MAC adresy MAC adresa daného člena clusteru. Také se v zachycené komunikace vůbec nezobrazují NLB heartbeaty.

### Vylepšení pomocí konfigurace switche

Rozchození IGMP Snoopingu není vůbec tak jednoduché, jak se může zdát (nestačí jej zapnout na aktivních prvcích, což je u Cisca default). Takže i při použití multicastu se nám většinou šíří komunikace jako broadcast. Naštěstí zde funguje stejná věc jako u unicastu. Můžeme na switchi zadat ***statické záznamy do CAM tabulky*** (toto nastavení přebije i IGMP Snooping).

```
SWITCH(config)#mac address-table static 0100.5e7f.020a vlan 112 interface G0/3 G0/17
```

Cisco hovoří ještě o tom, že u Catalystů 6000 a 6500 je třeba doplnit příkaz o klíčové slovo `disable-snooping`.

```
SWITCH(config)#mac address-table static 0100.5e7f.020a vlan 112 interface G0/3 G0/17 disable-snooping
```

Toto řešení je asi to nejlepší, protože se na porty mimo cluster a klienta nedostává žádná komunikace, která tam nepatří (ani NLB heartbeat).

## Tabulky MAC adres v různých režimech

Myslím, že velkou vypovídací hodnotu má tabulka MAC adres, které vidí různá zařízení při nastavení různého módu NLB. V první tabulce jsou zobrazeny výchozí hodnoty.

| | jméno | IP adresa | MAC adresa |
|---|---|---|---|
| **Server 1 (S1)** | PC07 | 192.168.2.30 | 00:22:19:1b:e8:fb |
| **Server 2 (S2)** | PC08 | 192.168.2.31 | 00:22:19:1c:01:e5 |
| **Cluster IP** | | 192.168.2.10 | |

Druhá tabulka ukazuje, jaké MAC adresy používají jednotlivá zařízení pro IP adresy jiného zařízení. Nejprve máme vlastní MAC adresy na adaptérech členů clusteru. Potom jakou MAC adresu vrací pro cluster IP.

| mode | server 1 | server 2 | cluster (PC vidí) |
|---|---|---|---|
| unicast | 02:bf:c0:a8:02:0a | 02:bf:c0:a8:02:0a | 02:bf:c0:a8:02:0a |
| multicast | 00:22:19:1b:e8:fb | 00:22:19:1c:01:e5 | 03:bf:c0:a8:02:0a |
| IGMP multicast | 00:22:19:1b:e8:fb | 00:22:19:1c:01:e5 | 01:00:5e:7f:02:0a |

Následně jaké MAC adresy vidí členové clusteru mezi sebou. Potom jaké MAC adresy vrací členové clusteru klientům.

| mode | S2 vidí pro S1 | S1 vidí pro S2 | PC vidí pro S1 | PC vidí pro S2 |
|---|---|---|---|---|
| unicast | 03:bf:c0:a8:02:1e | 03:bf:c0:a8:02:1f | 02:bf:c0:a8:02:0a | 02:bf:c0:a8:02:0a |
| multicast | 00:22:19:1b:e8:fb | 00:22:19:1c:01:e5 | 00:22:19:1b:e8:fb | 00:22:19:1c:01:e5 |
| IGMP multicast | 00:22:19:1b:e8:fb | 00:22:19:1c:01:e5 | 00:22:19:1b:e8:fb | 00:22:19:1c:01:e5 |

A poslední jsou MAC adresy na portech switche.

| mode | switch vidí S2 (Gi0/3) | switch vidí S1 (Gi0/17) |
|---|---|---|
| unicast | 02:02:c0:a8:02:0a | 02:01:c0:a8:02:0a |
| multicast | 00:22:19:1c:01:e5 | 00:22:19:1b:e8:fb |
| IGMP multicast | 00:22:19:1c:01:e5 | 00:22:19:1b:e8:fb |



Autor: Petr Bouška

**Související články:**

## Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

# Cisco Network Design - Enterprise Architecture

Pátek, 24.09.2010 13:47 | Samuraj - Petr Bouška | sítě

Studoval jsem na zkoušku Cisco Designing for Cisco Internetwork Solutions (pro certifikaci CCDA) a jako hlavní bod, z kterého se vychází, je Cisco Enterprise Architecture. O této architektuře jsem dříve neslyšel, tak zde uvedu alespoň stručnou zmínku.



Pouze pro referenci pár hlavních termínů, z kterých se vychází:

**SONA**
***Service Oriented Network Architecture*** - tento Cisco framework se používá pro návrh sítí, má tři vrstvy - network infrastructure layer, infrastructure services layer, application layer

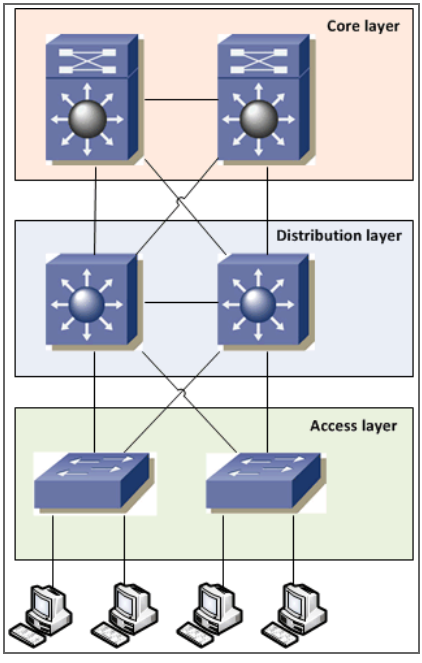
**AVVID**
***Architecture for Voice Video and Integrated Data*** - starší design pro integrované sítě

**IIN**
***Intelligent Information Network*** - nový design sítí, obsahuje tři faze - Integrated transport, Integrated services, Integrated applications

**PPDIO**
***Prepare, Plan, Design, Implement, Operate, Optimize*** - životní cyklus počítačové sítě

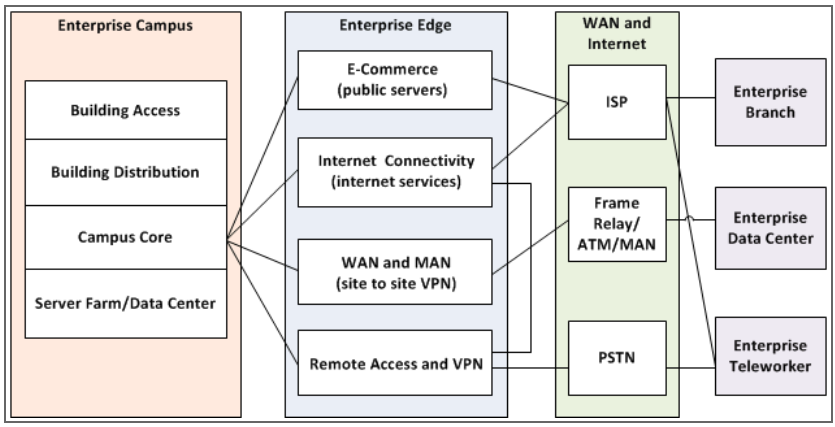
## Three-layer network design

Jedná se o hierarchický síťový model. V tomto případě je síťová hierarchie rozdělena do tří vrstev. Klasický síťový model doporučovaný Ciscem. V praxi je vhodný až pro rozsáhlé sítě, u středních sítí vystačíme s dvojvrstvým designem.



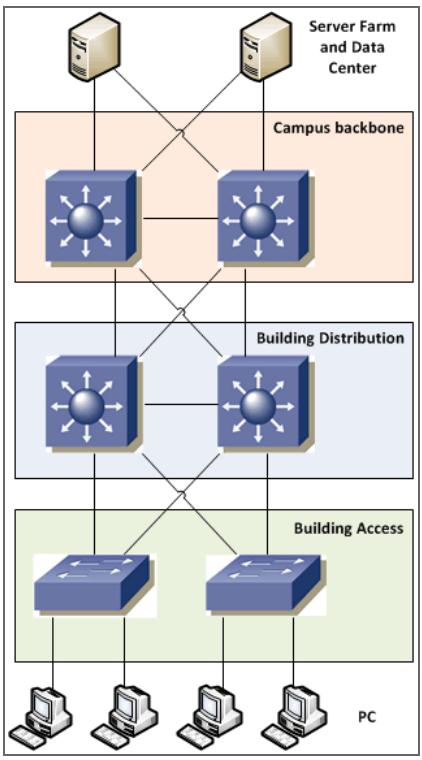

## Cisco Enterprise Architecture

Jedná se o modulární síťový design, který Cisco doporučuje pro rozsáhlé Enterprise sítě. Obrázek ukazuje obecné schématické znázornění. Čáry mezi bloky představují komunikace.




## Enterprise Campus

Podrobnější pohled na ***Enterprise Campus***, v podstatě se jedná o three-layer network design.




## Odkazy na informace ze stránek Cisco

- Cisco WAN and Application Optimization Solution Guide
- Enterprise Branch Architecture Design Overview
- Enterprise Campus 3.0 Architecture: Overview and Framework
- Data Sheet for Borderless Network Campus 1.0 Architecture
- Planning for Complex Networks



Autor: Petr Bouška

**Související články:**

## Počítačové sítě - Computer networks

Tento seriál se věnuje základům počítačových sítí. Jsou zde stručně popsány důležité praktické aspekty, které by měl znát každý, kdo se o sítě zajímá.

- OSI model [05.03.2007 08:09]
- VLAN - Virtual Local Area Network [02.06.2007 15:54]
- Počítačové sítě a jejich typy [09.07.2007 15:32]
- Počítačové sítě - základní topologie [30.07.2007 14:47]
- Ethernet - CSMA/CD, kolizní doména, duplex [03.08.2007 12:40]
- TCP/IP - model, encapsulace, paket vs. rámec [16.08.2007 16:43]
- TCP/IP - metody vysílání dat [02.09.2007 15:05]
- TCP/IP - adresy, masky, subnety a výpočty [05.09.2007 14:53]
- TCP/IP - navázání a ukončení spojení [13.09.2007 16:10]
- TCP/IP a ethernet - cesta v síti, aktivní síťové prvky [17.09.2007 10:22]
- TCP/IP - Routing - směrování [21.09.2007 15:34]
- TCP/IP - nalezení MAC adresy k IP - ARP [25.09.2007 17:35]
- Počítačové sítě - Computer Networks [30.09.2007 13:28]
- DNS (Domain Name System) zaměřeno na Microsoft [26.11.2007 14:26]
- Diagramy a schémata počítačové sítě (Visio) [13.01.2008 18:02]
- Wake on LAN - lokální i vzdálený subnet [10.08.2008 20:42]
- TCP/IP - Internet Protocol Version 6 - IPv6 [05.03.2009 15:41]

# VPN 1 - IPsec VPN a Cisco

Neděle, 10.04.2011 19:59 | Samuraj - Petr Bouška | sítě

Úvodní díl seriálu, který se věnuje technologii VPN. Naleznete zde stručný popis VPN a jejich typů. Dále se více popisuje VPN založená na protokolu IPsec, primárně typ Remote Access VPN. Celý popis je obecný, ale do budoucna se zaměřuje na nasazení na Cisco zařízeních, primárně Cisco ASA. Na konci jsou zmíněny parametry, které podporují klienti Cisco VPN Client, integrovaný klient na Google Android a ve Windows XP/Vista/7.



## Virtual Private Network

**Z**kratka **VPN** je z názvu ***Virtual Private Network***, což je privátní počítačová síť, která dovolí připojit vzdálené uživatele nebo pobočky do LAN organizace přes veřejné telekomunikační služby, převážně přes internet. Pro bezpečné vytvoření sítě na větší vzdálenost je možné využít pronajatých linek, ale to je velice drahé řešení (a pro připojení mobilních pracovníků nepoužitelné). VPN toto umožňuje přes relativně levný internet a bezpečnost řeší pomocí vytvoření ***šifrovaného tunelu*** mezi dvěma body (nebo jedním a několika). VPN řeší důvěrnost, autentizaci, neporušenost komunikace, apod.

**V**PN je velice široký termín a zahrnuje řadu protokolů a technologií. Hlavní typy VPN jsou dva:

- **Site-to-Site VPN** - spojujeme dvě (nebo více) sítě dohromady, většinou centrálu a pobočky, používají se speciální síťová zařízení (VPN koncentrátor, firewall, router, server), která slouží jako VPN gateway a naváží mezi sebou VPN spojení (příchozí komunikaci rozbalí a do sítě posílají standardně, odchozí zapouzdří do VPN tunelu), uživatelské stanice pak nepotřebují VPN klienta, často používané protokoly/typy jsou *IPsec VPN* a *MPLS VPN*
- **Remote Access VPN** - připojujeme individuální klienty do lokální sítě, klienti musí mít speciální SW - VPN klienta, na straně privátní sítě je opět speciální síťové zařízení, často používané protokoly/typy jsou *SSL VPN* a *IPsec VPN*

**D**ruhým nejčastějším dělením VPN je podle primárně použitého protokolu, na kterém je VPN postavena:

- **IPsec VPN** - ***Internet Protocol Security*** je asi nejrozšířenější forma VPN, je to součást IPv6 a hojně se používá v IPv4, často se kombinuje s ***L2TP*** (L2TP over IPsec), má problémy s NATem (řeší to NAT-T)
- **SSL VPN** - moderní metoda VPN, která využívá ***Transport Layer Security*** (TLS) nebo ***Secure Sockets Layer*** (SSL), často se využívá port 443 (HTTPS) a VPN dobře prochází přes FW (pracuje na vyšší vrstvě, takže nemá problémy s NATem), Microsoft vytvořil ***Secure Socket Tunneling Protocol*** (SSTP), což je přenos PPP nebo L2TP skrz SSL
- **MPLS VPN** - ***Multiprotocol Label Switching*** využívají hlavně ISP, kdy vytváří privátní sítě pro zákazníky přes MPLS

**E**xistuje i řada dalších typů VPN (třeba PPTP, SSH), které jsou ale méně rozšířené/používané.

## Typy VPN na Cisco ASA

- **IPsec VPN**
  - **Site-to-Site** - spojují se VPN zařízení
  - **Remote Access** - používá se Cisco VPN Client (nebo jiný SW VPN klient)
- **SSL VPN**
  - **Clientless SSL VPN** - přístup přes webový prohlížeč, dříve se označovala WebVPN
  - **Cisco SSL VPN Client** (Remote Access) - používá Cisco AnyConnect VPN Client

**O**d verze **ASA 8.4** (a ASDM 6.4) přidalo Cisco podporu **IKEv2** a rozděluje **IPsec VPN** na ***IPsec(IKEv1)*** a ***IPsec(IKEv2)*** pro který je potřeba ***Cisco AnyConnect Client verze 3.0*** (a také Anyconnect licence).

## IPsec Remote Access VPN

**IPsec**, plným názvem ***Internet Protocol Security***, je standardizovaná skupina protokolů pro zabezpečení IP komunikace mezi dvěma koncovými systémy. Obsahuje ***obousměrnou autentizaci*** a vyjednání ***kryptografických metod*** a klíčů. IPsec pracuje na IP vrstvě, to znamená na Layer 3 OSI modelu, a doplňuje IPv4 protokol (v IPv6 je povinnou součástí protokolu).

**IPsec** nejprve zařídí to, že se obě strany navzájem ***identifikují*** (autentizují) a následně ***šifruje*** veškerou komunikaci pomocí domluveného algoritmu. Komunikujícím stranám říkáme **peers**. V rámci IPsecu můžeme používat velké množství standardních protokolů a algoritmů. IPsec neurčuje, jaké algoritmy se musí použít pro komunikaci, ale definuje mechanismy vyjednávání a základní množinu algoritmů.

## IPsec režimy funkce

**IPsec** může pracovat ve dvou módech (host-to-host) ***Transport*** a (network) ***Tunnel Mode*** (defaultní), rozdíl je v tom, co se šifruje.

- ***Tunnel mode*** - šifruje celý paket (včetně hlavičky) a doplňuje novou hlavičku. Tento mód se může použít pro IPsec proxy, klient posílá data, router je šifruje a posílá dál. Z komunikace se nedá odhalit adresa klienta.
- ***Transport mode*** - šifruje pouze data, IP hlavička se ponechá a doplní se pouze IPsec hlavička. Někteří klienti podporují pouze transport mode, například L2TP/IPsec klient ve Windows.

## IPsec protokoly

**IPsec** využívá tři hlavní protokoly:

- **Authentication Header** (AH) - zajišťuje integritu a autentizaci zdroje dat, využívá hashovací funkce (jako MD5 či SHA1) a společný klíč (který si na začátku domluví), v hlavičce obsahuje pořadové číslo paketu, pro komunikaci používá ***protokol IP číslo 51***, dnes se téměř nepoužívá
- **Encapsulating Security Payload** (ESP) - zajišťuje důvěrnost, autentizaci zdroje a integritu, využívá šifrovací algoritmy (jako DES či AES), pro komunikaci používá ***protokol IP číslo 50***
- **Security Association** (SA) - skupina algoritmů, které poskytují parametry pro bezpečnou komunikaci pomocí

AH a ESP, používá ***ISAKMP*** Framework (Internet Security Association and Key Management Protocol) a nějaký protokol, většinou **IKE** (Internet Key Exchange), pro vyjednání atributů, atributy obsahují zvolený šifrovací algoritmus, dobu platnosti klíče, kompresi a způsob zapouzdření, komunikace probíhá šifrovaně

## Navázání IPsec VPN

Běžně IPsec VPN komunikace probíhá:

1. pomocí **IKE** se na UDP portu 500 vyjedná **SA**, používá certifikáty (klientský a serverový) nebo PSK
2. z parametrů **SA** se naváže na IP 50 šifrovaná komunikace pomocí **ESP**

## IKE - Internet Key Exchange

Často se na začátku **IPsec** komunikace používá ***IKE*** protokol pro vyjednání ***Security Association***. **IKE** používá pro komunikaci ***UDP port 500*** a pro autentizaci *certifikáty* nebo *Pre-Shared Key* (PSK). Pro výměnu klíčů se používá metoda *Diffie-Hellman* (vytvoří *shared session secret* a z něj se odvodí šifrovací klíče). Takže komunikace je šifrována a IKE může sloužit jako část autentizace (musíme mít certifikát nebo znát PSK). Nověji se používá také IKEv2.

IKE pracuje ve dvou fázích:

- **Phase 1** - autentizují se účastníci a vyjedná se IKE SA (pomocí Diffie-Hellman), díky tomu se vytvoří bezpečný kanál pro vyjednání IPsec SA ve fázi 2, pracuje v režimu *main mode* (chrání identitu komunikujících stran) nebo *aggressive mode*
- **Phase 2** - vyjedná IPsec SA parametry a nastaví odpovídající SA, pouze režim *quick mode*

**Perfect Forward Secrecy (PFC)** - volitelné rozšíření zabezpečení fáze 2. Standardně při IPsec vyjednávání jsou klíče fáze 2 odvozeny od klíčů fáze 1. PFS využije novou výměnu klíčů pomocí Diffie-Hellman.

Rozšíření IKE protokolu **XAUTH** (Extended User Authentication) dovoluje použít autentizaci uživatele vůči ***RADIUS serveru***. Nejde o náhradu, ale o rozšíření. Standardně IKE podporuje ve fázi 1 pouze autentizaci počítače. XAUTH přidává další fázi (po fázi 1 a před fází 2), kdy dojde k autentizaci uživatele (pomocí AAA metod).

## NAT-T - Network Address Translation - Traversal

Pakety posílané s IPsec jsou chráněny ***hashem***, aby nemohl být jejich obsah změněn. Pokud je v cestě ***NAT***, tak ten modifikuje IP hlavičku, potom nesouhlasí hash a takový paket je zahozen. Řešením, aby komunikace fungovala, je využití **NAT-T**. Ten doplní paketu novou ***UDP hlavičku*** (provede zapouzdření paketu do UDP), ta může být cestou modifikována. Na druhé straně se hlavička odstraní a příchozí paket je OK. Vyjednávání během IKE může domluvit UDP zapouzdření. Komunikace používá ***UDP port 4500*** a označuje se jako ***IPsec over UDP*** nebo ***IPsec over NAT-T***.

U ***Cisca*** je možné zapouzdření nejen do **UDP**, ale také do **TCP** a stanovení portu. Cisco také používá různá označení, třeba ***Transparent Tunneling*** či ***NAT Transparent***.

## L2TP over IPsec

**Layer 2 Tunneling Protocol** (L2TP) je tunelovací protokol pro podporu VPN. Sám o sobě neobsahuje žádné šifrování, pouze vytváří tunel. Komunikuje na ***UDP portu 1701***. Často se používá dohromady s IPsec, který zajišťujte důvěrnost (šifrování) a autentizaci. Mluvíme pak o ***L2TP/IPsec***.

Proces navazování spojení může probíhat:

1. pomocí **IKE** se na UDP portu 500 vyjedná **SA**, používá certifikáty (klientský a serverový) nebo PSK
2. z parametrů **SA** se naváže na IP 50 šifrovaná komunikace pomocí **ESP v transportním módu**
3. vyjedná se a naváže **L2TP tunel** mezi koncovými body, komunikace probíhá přes UDP 1701, ale to je zabalené v IPsec, takže v paketu jde o IP 50

## Další vlastnosti

**Split Tunneling** - pouze část komunikace od klienta jde do VPN (firemní rozsah adres) a zbytek jde přímo do internetu. Běžný způsob je, že veškerá komunikace je směrovaná do VPN tunelu.

# Parametry pro některé IPsec VPN klienty

Když vytváříme VPN, tak potřebujeme znát, jaké algoritmy a protokoly podporuje určitý klient, aby nám výsledná komunikace fungovala.

## Cisco VPN client

***Cisco VPN client*** podporuje většinu možností a téměř vše, co můžeme nastavit na Cisco ASA. IPsec musí být v ***tunnel mode*** a není podporován L2TP. Můžeme použít ***IPsec ESP*** s ***NAT-T*** jako IPsec/UDP či IPsec/TCP. Zabezpečení ***ESP*** šifrování **DES**, **3DES** (56/168-bit) a **AES** (128/256-bit), pro integritu hashovací funkce **MD5** a **SHA**, IKE s digitálními certifikáty, Diffie-Hellman (DH) Groups 1,2 a 5 s PFS. Kompresi dat LZS. Pro autentizaci RADIUS, ale nepodařilo se mi rozchodit autentizaci uživatelským certifikátem.

## Windows XP či Windows 7

***Microsoft*** má v řadě svých klientů zabudovaného IPsec VPN klienta (ve Windows 7 je také podpora SSTP a IKEv2, nezkoušel jsem). Podporuje i NAT-T, ale pokud je server za NATem, tak se na klientovi musí povolit v registrech.

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\RasMan\Parameters\"NegotiateDh2048"=dword:1
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\IPSec\"AssumeUDPEncapsulationContextOnSendRule"=dword:2
```

Ve Windows 7 se druhá hodnota nachází na jiném místě.

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\PolicyAgent\"AssumeUDPEncapsulationContextOnSendRule"=dword:2
```

Standardní klient je ***L2TP/IPsec***, takže L2TP je potřeba, a musí jít o ***transport mode***. Zabezpečení je podporováno ***ESP*** šifrování **DES** a **3DES**, integritu **MD5** nebo **SHA** a D-F group 1, 2, 2048. Není zde problém s využitím EAP smartcard, tedy autentizace uživatelským certifikátem.

## Google Android

V OS pro mobilní telefony ***Google Android*** je integrovaný VPN klient (údajně) již od verze 1.6. Volíme mezi ***L2TP/IPsec PSK VPN*** (pro IKE se použije PSK) a ***L2TP/IPsec CRT VPN*** (pro IKE máme certifikát). Nikde jsem nenašel popis vlastností, ale z praktických testů vyplívá, že podpora je podobná jako ve Windows XP. Tedy ***L2TP/IPsec*** v ***transportním módu***, ***ESP*** s **DES** a **3DES**, **MD5** a **SHA1**.

# Začínáme s monitoringem sítě

Úterý, 01.09.2009 12:21 | Samuraj - Petr Bouška | administrace

Monitoring serverů, síťových prvků a dalších zařízení je dnes nezbytností, pokud chceme zvládat správu sítě. Přitom možností a monitorovaných oblastí je velké množství. Pokud známe odpovídající technologie, tak můžeme postavit kvalitní monitoring i zdarma. Pojďme se tedy podívat na možnosti spolu s technologiemi, které jsou k dispozici.



*Tento článek jsem napsal pro časopis Connect a vyšel v čísle Connect 09/09, zde jej publikuji s laskavým svolením redakce.*

## Co monitoring dává a jaké technologie jsou k dispozici

Monitoring má mnoho podob a můžeme využít řadu technologií a protokolů. Celý monitorovací systém můžeme postavit na vlastních skriptech či na bezplatných řešeních, kdy investujeme pouze svůj čas a znalosti. Nebo využít některý z rozsáhlé nabídky komerčních produktů.

Pokud nebudeme spoléhat na cizí aplikace, ale vytvoříme vlastní skripty či programy, tak budeme mít detailní přehled o tom, jak monitoring probíhá, a naše řešení bude přesně odpovídat našim potřebám. Musíme však mít hlubší znalosti skriptování či programování, spolu se znalostí síťových protokolů a technologií. Navíc je tvorba takového řešení časově náročná.

Produkty zdarma jsou často značně univerzální se širokými možnostmi konfigurace. Vyžadují však také hlubší znalosti pro nastavení, protože některé konfigurace znamenají psaní vlastních skriptů. Výhoda je, že máme komplexní prostředí (které zahrnuje třeba konfiguraci, dashboard, zpracování grafů) a na míru nastavujeme pouze určité šablony pro získávání dat.

Oproti tomu komerční produkty většinou nainstalujeme na pár kliknutí a monitorování rozchodíme během pár minut. Stačí znát adresy zařízení, a jaké údaje na nich chceme monitorovat. Většinou jsou zde připraveny šablony pro jednotlivé oblasti. To nám ovšem zároveň omezuje možnosti použití.

Samozřejmě nic není pouze bílé nebo černé, takže i do komerčních aplikací můžeme dopisovat vlastní skripty. A nalezneme i volně šiřitelné systémy, které jsou připravené na nejběžnější nasazení.

***Některé monitorovací systémy***
***Zdarma*** – Nagios, Cacti, Zabbix
***Placené*** – Zenoss, PacketTrap pt360, WhatsUp Gold
***Od velkých firem*** – Microsoft System Center Operations Manager, HP OpenView, IBM Tivoli Netcool, CiscoWorks LAN Management Solution

## Co chceme sledovat?

Než začneme vlastní monitoring plánovat, je třeba si uvědomit, co přesně chceme sledovat. A jaké výstupy z monitoringu jsou pro nás prioritní. Podle toho je třeba zvolit použité technologie. I když existují rozsáhlé systémy s řadou komponent, tak asi nenalezneme monitorovací nástroj, který by obsáhl všechny oblasti, které potřebujeme sledovat ve větší firmě. Musíme tedy kombinovat více produktů.

Na monitoring se můžeme dívat ze dvou pohledů. Chceme se dozvědět, že někde došlo k problému. Tedy, že něco přestalo fungovat či byl překročen nějaký kritický limit. Nebo chceme získávat aktuální (ale i historické) informace o určitém systému. To může být pohled na vytížení serveru, abychom plánovali jeho další využití. Přehled, kde v síti (do jakého portu jakého switche) je připojen klient s jakou IP a MAC adresou. Či sledování vytížení datových linek.

## Oblasti pro monitoring

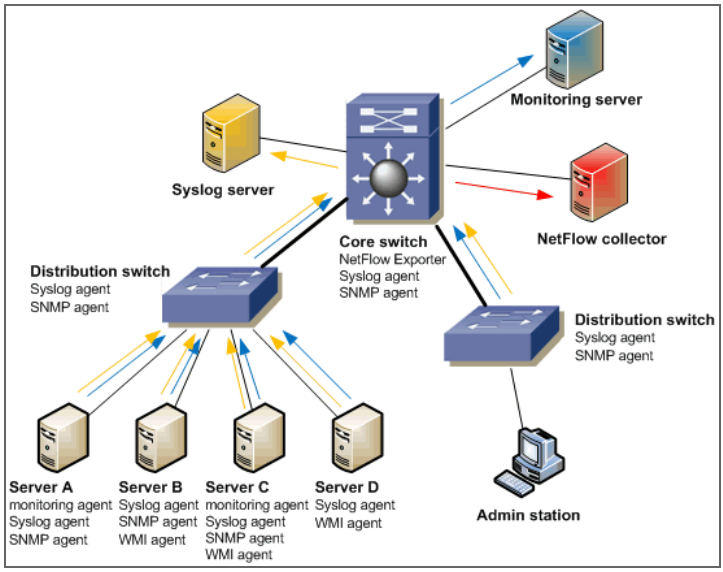
Pokusíme se nyní zamyslet, co vše je možné v počítačové síti monitorovat. Nejprve z obecnějšího hlediska můžeme uvést monitoring:

- serverů a jejich služeb
- aktivních síťových prvků
- síťové komunikace/provozu
- bezpečnosti

K tomu se přidávají určité specifické oblasti, kdy si v základu vystačíme se stejným monitoringem, jako pro výše uvedené skupiny, ale můžeme získat více použitím specializovaného nástroje. To jsou oblasti jako IP telefonie, bezdrátové sítě a virtuální prostředí.

Když půjdeme hlouběji, tak nás může zajímat:

- dostupnost serverů
- dostupnost služeb/aplikací (spolu s latencí – reakční dobou)
- události na serverech
- vytížení zdrojů (procesor, paměť, disk)
- vytížení linek – měření přenosu dat
- statistiky síťového provozu
- analýzy nestandardního chování v síti
- informace o portech switchů
- monitoring speciálních oblastí jako je WiFi či IP telefonie
- bezpečnostní incidenty




## Technologie pro monitoring

Pokud použijeme nějaký komplexní monitorovací systém pro dohled serverů, tak máme většinou dvě základní možnosti pro přístup k informacím.

***Monitorování s agentem***, kdy na server instalujeme speciálního klienta. Musíme tedy mít k dispozici agenta pro daný operační systém, musíme mít možnost na sever instalovat a přibývá další aplikace, která může způsobovat problémy. Na druhou stranu dostaneme širokou škálu údajů, které můžeme ze serveru získávat.

Druhou možností je ***monitorování bez agenta***, kdy se testují vlastní služby serveru nebo se data získávají pomocí určitých standardních protokolů (jako SNMP, WMI, IPMI).

Výše jsme si popsali oblasti, které bychom pomocí monitoringu chtěli sledovat. Nyní uvedeme stručný seznam technologií, které můžeme pro tento monitoring použít, ať již samostatně nebo uvnitř monitorovacího systému.

- dostupnost serveru pomocí ping testu
- dostupnost služby pomocí navázání TCP spojení nebo na aplikační úrovni
- události ze serverů - Syslog
- získávání údajů pomocí klienta
- získávání údajů pomocí monitorovacích protokolů WMI, SNMP, IPMI
- sledování síťových toků - NetFlow
- analýza síťových protokolů - network protocol analyzer
- bezpečnost v síti - IDS/IPS

## Výstupy monitoringu – reporty a alerty

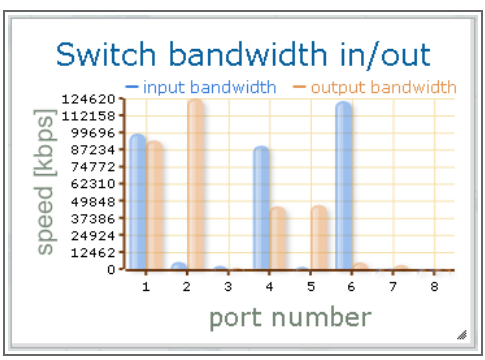
Můžeme si nastavit perfektní monitoring, který bude hlídat a zaznamenávat vše v naší síti, ale pokud nebudeme mít přehledné, dostupné a často i inteligentní výstupy, tak ničeho nedosáhneme. Proto je důležité již od začátku plánovat, jaký výstup pro jakou oblast je pro nás nejvhodnější.

Z pohledu typů dat máme dvě základní oblasti. Jedna jsou ***události***, které získáme pomocí Syslogu, WMI či SNMP trapů ze serverů, switchů a dalších zařízení. Druhá jsou ***hodnoty***, často číselné, ukazující okamžitý stav nějaké vlastnosti. K tomu se ještě přidává otázka, zda nás zajímá okamžitý stav nebo potřebujeme ukládat historii, jak se hodnoty mění v čase.

Pro různé sledované údaje se samozřejmě hodí různá reprezentace. Vytížení procesoru nás zajímá v určité časové periodě a vhodné zobrazení je grafem. Stavy portů switche naopak chceme vidět aktuální a přehledná je tabulka. Dostupnost serveru za období se může zobrazit jednou procentuální hodnotou.

Pro globální pohled na síť je výhodná grafická reprezentace, kdy vidíme pohled na schéma sítě nebo její části. Pokud dojde někde k problému, tak se daný prvek zvýrazní a po rozkliknutí dostaneme detailní informace. Vedle toho můžeme mít zobrazeny kategorie událostí podle závažnosti zobrazující počet nevyřešených incidentů.

Výše popsané reprezentace jsou zajímavé a v určitých situacích užitečné. Pokud se však jedná o bezpečnostní nebo havarijní události, a my nemáme dohledový tým, který neustále sleduje dashboard monitorovacího systému, tak je mnohem užitečnější vygenerování a odeslání emailové nebo SMS zprávy (třeba pomocí IP GSM brány). Můžeme takto zasílat důležité zprávy pro informaci a kritické zprávy, abychom provedli rychlou reakci.




Nad událostmi ze serverů a aktivních prvků, či také nad datovými toky z NetFlow, může běžet inteligentní systém, který vyhodnocuje množství dat z různých systémů, hledá mezi nimi souvislosti, a generuje bezpečností varování. Případně rovnou provádí určitou reakci, jako je zablokování portu switche, ze kterého se šíří útoky do sítě.

## Dostupnost zařízení/služby

Podíváme se nyní trochu blíže na základní technologie pro monitoring.

Asi první věcí, kterou začneme monitorovat, je dostupnost serveru či aktivního síťového prvku. V malé firmě, kde máme jeden dva servery, se nedostupnost projeví patrně velice záhy. Ve větším prostředí se však mohou začít kupit problémy navazujících služeb, a může trvat delší dobu, než někdo zaregistruje nedostupnost nějakého serveru či síťové cesty (které bývají často redundantní).

Běžně se dostupnost zařízení zjišťuje pomocí jednoduché metody ***ICMP echo request/response*** (tedy ping). Případně se používá tzv. „SNMP ping", což je SNMP dotaz na běžné OID. Takto můžeme zjistit, zda je dané zařízení „živé" a měřit dobu odezvy (latency). Nedozvíme se však, že třeba webový server Apache neběží. Proto je dalším krokem monitoring dostupnosti aplikační služby.

Většina běžných síťových služeb využívá protokol TCP a poslouchá na nějakém portu. Takže můžeme testovat, zda se nám na server podaří ***navázat TCP spojení*** na daný port. To znamená, že daná služba běží a poslouchá. Lepším ověřením je ***aplikační test***, kdy ověříme, že se služba chová jak má. Takže například pro webserver se připojíme ke stránce a ověříme, zda vrací v hlavičce kód 200. Pro mail server zavoláme základní SMTP příkazy pro navázání spojení se serverem apod.

## Syslog – události ze serverů

Syslog je standard pro přeposílání zpráv z logů po síti. Slouží k tomu, abychom koncentrovali logy z různých zařízení a jejich aplikací na jedno místo a mohli na ně reagovat. Na klientovi potřebujeme aplikaci, která odesílá zprávy z logu, tak jak přibývají, pomocí protokolu **Syslog**. A potom potřebujeme ***Syslog server***, který tyto zprávy přijímá a zpracovává.

V oblasti Linuxu se jedná o běžnou záležitost. Trochu složitější je situace v Microsoftím světě. Windows si sami vytváří řadu logů označovaných jako ***Event Log*** a různé servery přidávají další logy. Tyto logy jsou v MS formátu a nativně nemáme žádnou podporu pro Syslog. Na internetu však nalezneme zdarma aplikace, které fungují jako Syslog server, například **Kiwi Syslog Server**. A také klienty, kteří přeposílají vybrané události z Event Logu, příkladem je **Snare Agent for Windows** nebo **SaberNet NTsyslog**.

Syslog je velice užitečný, protože do logů přibývají běžně stovky zpráv za minutu a pro desítky zařízení nemáme šanci tyto informace procházet. Na Syslogu můžeme vytvořit skripty, které analyzují příchozí zprávy a upozorní nás na problémy. Například ze Security logu Windows můžeme číst špatné pokusy o přihlášení a když se jeden účet snaží v určitém časovém intervalu přihlásit mnohokrát, tak odeslat email správci, že se může jednat o útok.

Druhou výhodou je, že můžeme uchovávat velké množství zpráv, tedy dlouhou historii. Řada zařízení dokáže lokálně uložit pouze omezené množství logů. Navíc máme tyto logy dostupné, i když není dostupný server. Pokud došlo na serveru k poruše nebo byl napaden útočníkem a my se nemůžeme přihlásit, abychom si přečetli lokální log. Tak na Syslogu nalezneme zprávy, které se uložily těsně před tím, než přestal server odpovídat.

Monitoring je oblast kreativní a nikde není dáno, jak jej musíme provádět. Vše záleží na našich potřebách, schopnostech a přáních. Zde jsme si pouze uvedli možnosti, které můžeme použít. Příště se podíváme podrobněji na základní protokol pro síťovou správu, což je SNMP. A také na protokol pro sledování toků v síti, tedy NetFlow.

***Použité zkratky***
***ICMP*** – *Internet Control Message Protocol*
***SNMP*** – *Simple Network Management Protocol*
***OID*** – *Object Identifier*
***WMI*** – *Windows Management Instrumentation*
***IPMI*** – *Intelligent Platform Management Interface*
***IPS*** – *Intrusion Prevention System*
***IDS*** – *Intrusion Detection System*
***IP*** – *Internet Protocol*
***TCP*** – *Transmission Control Protocol*
***MAC*** – *Media Access Control*



Autor: Petr Bouška

**Související články:**

## Monitoring sítě

Úvod do monitoringu počítačových sítí a popis základních protokolů.

- Začínáme s monitoringem sítě [01.09.2009 12:21] ..... **právě čtete**
- Zařízení v síti pod kontrolou [21.09.2009 17:08]
- Automatizovaná kontrola prostředí pomocí PowerShellu [13.06.2011 17:35]

# Zařízení v síti pod kontrolou

Pondělí, 21.09.2009 17:08 | Samuraj - Petr Bouška | administrace

Samostatnou kapitolou monitoringu je sledování a analýza datových toků v síti. K tomu nám slouží protokol NetFlow. Druhou oblastí je dohled a správa zařízení v síti. Pokud porozumíme nejrozšířenějšímu protokolu SNMP, můžeme sami nebo pomocí monitorovacích systémů, dosáhnout úžasných výsledků.



*Tento článek jsem napsal pro časopis Connect a vyšel v čísle Connect 10/09, zde jej publikuji s laskavým svolením redakce.*

## Jak na monitoring sítě pomocí SNMP a NetFlow

V minulém čísle časopisu Connect! jsme rozebírali základní oblasti a principy monitoringu sítě. Dnes si podrobněji popíšeme další dva důležité protokoly NetFlow a SNMP.

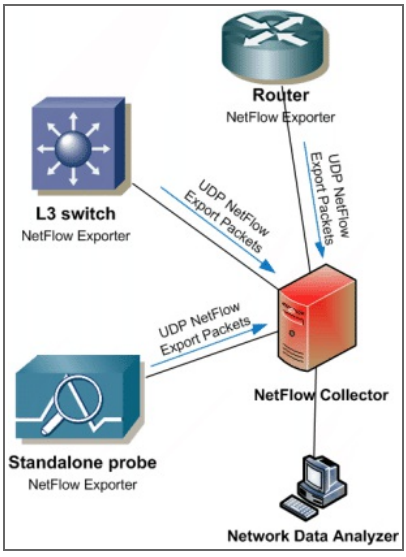
## NetFlow – datové toky v síti

NetFlow, sFlow, IPFIX (IP Flow Information Export) a další jsou protokoly a mechanismy určené pro sbírání datových toků a jejich odeslání na jiné zařízení pro shromažďování a analýzu. Různé protokoly byly vytvořeny různými výrobci aktivních síťových prvků a jsou většinou proprietární. **NetFlow**,v současnosti asi nejpoužívanější, byl vytvořen firmou Cisco, později na něm byl založen IETF standardem IPFIX.

Princip funkce je takový, že v síti máme jeden nebo několik **Observation Point**, tedy míst, kde se sleduje provoz (většinou jeden nebo více interfaců nějakého zařízení). Na tomto místě sledujeme provoz, provádíme filtrování a agregaci a vytváříme záznamy o tocích **Flow Record**. Tyto záznamy obsahují informace o jednotlivých tocích **IP Flow** (včetně času začátku a konce, vstupní a výstupní interface, počet bytů a paketů v toku, údaje z hlavičky paketu).

**Síťový tok** je jednosměrná sekvence paketů, které projdou skrze Observation Point v určitém časovém intervalu, a které mají shodné základní vlastnosti paketu (jako IP adresa, port, protokol).

Zařízení, kde běží NetFlow služba se označuje jako **Exporter**(může se jednat o router, switch nebo samostatnou sondu). Ten monitoruje pakety, vytváří toky těchto paketů a tato data odesílá ve formě Flow Records v pravidelném intervalu na **NetFlow Collector**. Zde se zpracovávají a ukládají (do textových souborů nebo databáze). Collector může přijímat data z více Exporterů. Nad databází záznamů nám pak může běžet nějaká aplikace, která provádí analýzu a vizualizaci NetFlow dat.




Podpora NetFlow je primárně u zařízení firmy Cisco a to u routerů, již od nízkých modelů, jako je řada 800. Kdežto u switchů až vyšších řad Catalyst 4500 a výše.

Standardně se ukládají údaje o routovaném provozu, jaké přesně údaje shromáždíme, záleží na verzi NetFlow. První verze měla číslo 1. Dnes je stále rozšířena starší ***verze 5 a 7*** (obdoba verze 5, ale speciálně pro Catalyst 6500 a 7600). ***Verze 8*** přidala různá agregační schémata a snížila využití zdrojů. Zatím poslední ***verze 9*** se označuje jako Flexible NetFlow. Má rozšiřitelný exportní formát, takže můžeme doplňovat další sledovaná pole (používá se třeba pro MPLS, Multicast, BGP).

Používání NetFlow je vhodné z řady důvodů. Můžeme sledovat zatěžování sítě v čase a plánovat rozšiřování kapacity nebo lepší využití stávajících prostředků. Zároveň můžeme údaje z NetFlow použít pro analýzu bezpečnostních hrozeb, například detekovat Denial of Service (DoS) útoky.

Z volně dostupných nástrojů pro NetFlow je nejrozšířenější collector **Nfdump**, který sbírá a zpracovává data z příkazové řádky. Nad těmito daty můžeme postavit vlastní skripty, které reagují na určité situace. Nebo můžeme použít grafickou webovou nadstavbu **NfSen**, která dokáže zobrazovat grafy se základními údaji.

Abychom z NetFlow dostali maximum, tak potřebujeme chytrý analyzátor, protože jinak se jedná pouze o velké množství dat. Jako příklad specializovaného komerčního řešení je **Cisco MARS** či **NetFlow Tracker** od *FLUKE networks*.

## SNMP – protokol pro správu

Určitě můžeme říci, že základem pro správu (monitoring i konfiguraci) síťových zařízení, je protokol **SNMP** (Simple Network Management Protocol). Jedná se o sérii standardů (popsaných v RFC od IETF), která se poprvé objevila již v roce 1988 a v dnešní době je masově rozšířena. Podpora SNMP je nejen u serverů a aktivních síťových prvků (jako jsou switche a routery), ale téměř u každého zařízení, které komunikuje po síti (tiskárny, UPS, WiFi AP, FW, různá síťová čidla, jako třeba teploměr).

Pomocí SNMP můžeme číst nebo nastavovat hodnoty na zařízení, anebo obdržet upozornění na nějakou definovanou událost. Hodnoty můžeme načítat jednorázově v případě potřeby nebo v pravidelném intervalu. Další zpracování již záleží na nás (potažmo na použité aplikaci), zda pouze zobrazíme danou hodnotu nebo budeme vykreslovat graf (třeba průběh teploty či vytížení procesoru v čase) či provedeme definovanou reakci.

Protokol SNMP vyžaduje pro komunikaci dvě strany. Jednou entitou je **správce** (manager) a druhou **agent**. SNMP pracuje ve dvou režimech činnosti:

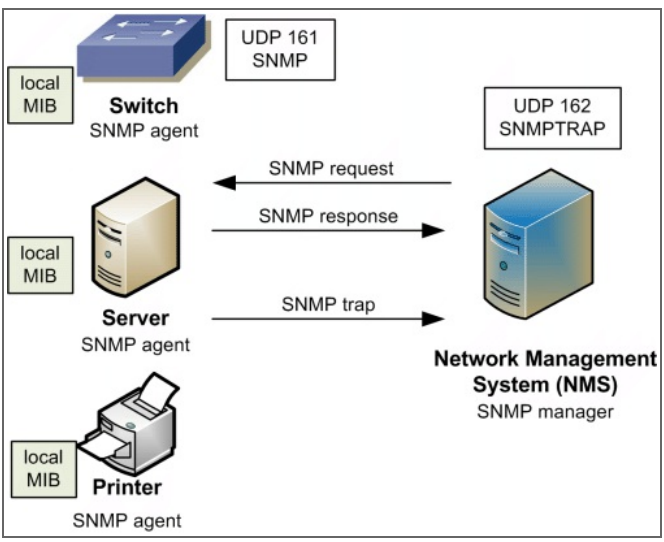
- **Správce posílá dotazy agentovi a přijímá odpovědi** – hodnoty tedy může získávat i více správců a mohou se ptát kdykoliv
- **Agent zasílá oznámení (trapy) na adresu správce** – v nějakých definovaných situacích (překročení nějaké hodnoty nebo i v pravidelném intervalu) odesílá agent jednomu správci hodnoty

### Různé verze SNMP

V současnosti existuje protokol SNMP ve třech verzích. Prvotní verze **SNMPv1**, která má řadu nedostatků, takže je dobré se jí pokud možno vyhnout. Potom přišla SNMPv2, tato verze existovala v několika variantách, ale dnes se používá pouze verze **SNMPv2c** (a jedná se asi o nejrozšířenější verzi). Poslední je **SNMPv3**, která je doporučována z bezpečnostního hlediska, ale její použití je o něco obtížnější.

Informace odesílané v SNMP paketech jsou standardně uloženy v čistém textu. Aby data ze zařízení mohl číst nebo nastavovat pouze oprávněný uživatel, tak se používá určitá forma autentizace. Pro SNMPv1 a SNMPv2c se používá **community string**, tedy textový řetězec, a ten se přenáší v každém paketu jako čistý text. Přesněji se používají dvě tato hesla *read community string*, který povoluje přístup pouze pro čtení a *write community string*, který povoluje i zápis. SNMPv3 přináší několik vylepšení v oblasti bezpečnosti a to je autentizace pomocí jména a hesla (User-based Security Model), ochranu proti změně či odhalení (kontrolní součty MD5 a SHA a šifrování komunikace pomocí DES) a řízení přístupu k jednotlivým objektům (View Access Control Model).

Protokol SNMP může běžet nejen nad IP, ale také dalšími protokoly, jako OSI CLNS, AppleTalk DDP či Novel IPX. V praxi se ale sekáme téměř výlučně s IP sítěmi, takže pro zjednodušení budeme uvažovat pouze TCP/IP verzi SNMP.




SNMP využívá pro komunikaci protokol **UDP**, který je rychlý, ale neobsahuje mechanismus pro ověření doručení (který je třeba v TCP). SNMP od verze 2 obsahuje vlastní mechanismus pro kontrolu doručení. Každá SNMP zpráva se zapouzdří do jednoho UDP paketu. Když správce posílá dotaz klientovi, tak zdrojový port je dynamicky volený a cílový je defaultně **port 161** (na něm poslouchá SNMP agent). Agent odpovídá z portu 161 na dynamický port správce. Trapy odesílá agent z dynamického portu na adresu správce **port 162**.

### Do hlubin komunikace

Když se podíváme více do hloubky, jak funguje SNMP komunikace, tak je to velmi jednoduché. SNMP má přesně daný formát paketu, jehož aplikační část (4. vrstva dle TCP/IP modelu) je schematicky zobrazena na obrázku. Je zde vidět rozdělení na hlavičku, která obsahuje určení použité *verze SNMP* a *community string*, a *uživatelská data*, která se označují jako SNMP PDU (Protocol Data Unit). Datová část je složena z určení *SNMP operace*, *čísla dotazu* pro svázání mezi dotazem a odpovědí, případného *chybového kódu* a *ukazatele na objekt*, kde k chybě došlo, a variable bindings, což je svázání OID a odpovídající hodnoty. Protože se můžeme najednou ptát na více hodnot, tak *variable bindings* se může vyskytovat vícekrát za sebou pro různé OID.

Uvnitř SNMP paketu se data ukládají pomocí Basic Encoding Rules (BER), což zjednodušeně znamená, že každé pole je uloženo jako určení datového typu, délka a vlastní data. Typů je definováno několik, příkladem je sekvence (kód 0x30), Integer (2), OctetString (4), Null (5).

| verze SNMP | community string | SNMP PDU | | | | | |
|---|---|---|---|---|---|---|---|
| verze SNMP | community string | SNMP operace | ID dotazu | kód chyby | index chyby | variable bindings (jedno nebo více) | |
| verze SNMP | community string | SNMP operace | ID dotazu | kód chyby | index chyby | OID | hodnota |
| Příklad dotazu | | | | | | | |
| v2c (1) | public | get-request (0) | 29854 | noError (0) | 0 | sysUpTime (1.3.6.1.2.1.1.3.0) | 0 |
| Příklad odpovědi | | | | | | | |
| v2c (1) | public | get-response (2) | 29854 | noError (0) | 0 | sysUpTime (1.3.6.1.2.1.1.3.0) | 1499579826 |

Příklad komunikace si popíšeme na operaci `SNMP GET`. Standardní komunikace probíhá tak, že se do SNMP paketu nastaví typ na get-request, ID dotazu, hodnota OID, které chceme přečíst a vlastní hodnota se nastaví na NULL. Takto vytvořený UDP paket se odešle na agenta.

Ten provede zpracování a odešle odpověď, kde nastaví typ na get-response, zachová ID dotazu, a pro OID nastaví načtenou hodnotu.

SNMP má definováno několik operací:

- **Get** – pro získání jedné nebo více instancí objektu (v SNMPv1 pokud dojde k chybě u jedné hodnoty, nevrátí nic, u SNMPv2 již vrací to, co je načteno bez chyby)
- **GetNext** – vrátí instanci dalšího objektu v seznamu nebo tabulce
- **Set** – nastaví hodnotu instanci objektu
- **Trap** – asynchronně informuje o události
- **GetBulk** – nově od SNMPv2, vrací větší blok dat (jako několik řádků z tabulky)
- **Inform** – nově od SNMPv2, slouží pro komunikaci mezi dvěma správci

V SNMPv1 je jiný formát paketu pro Trap než pro ostatní operace. Ve verzi SNMPv2 došlo ke zjednodušení a formát byl sjednocen na stejný.

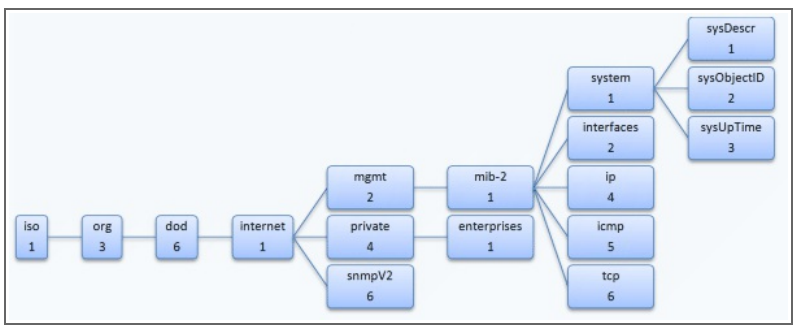
Spravované objekty, tedy hodnoty či vlastnosti, které můžeme pomocí SNMP číst či nastavovat, jsou uloženy ve virtuálním informačním úložišti, které se nazývá **Management Information Base - MIB**. Struktura MIB je hierarchická a reprezentuje strom. Začíná nepojmenovaným kořenem a pod ním jsou uzly, každý uzel může být kořenem podstromu. Uzly mají návěští, které se skládá ze stručného textového popisu a čísla.

Pro jednoznačnou identifikaci objektu se používá **Object Identifier – OID**, což je sekvence celých čísel oddělených tečkou. OID odráží hierarchickou strukturu MIB, kdy bere OID nadřazeného uzlu a doplní za tečku své číslo. Příkladem OID pro podstrom `MGMT` je `.1.3.6.1.2`. OID můžeme zapisovat i pomocí popisků z návěští, takže MGMT můžeme zapsat jako `.iso.org.dod.internet.mgmt`.

Související údaje se ukládají do **MIB modulů**, které dohromady tvoří MIB databázi. MIB moduly se ukládají jako textové soubory. Správce nepotřebuje mít k dispozici MIB DB, ta mu ale může pomoci třeba pro reprezentaci hodnot. Agent MIB databázi obsahuje, v ní jsou definovány objekty, které zpracovává.

MIB používá pro zápis notaci dle **SMI** (Structure of Management Information) což je podmnožina standardu **ASN.1** (Abstract Syntax Notation One). Se SNMPv2 přišlo rozšířené SMI, které přidává několik hodnot. Pomocí SMI jsou definovány jednotlivé typy objektů, které můžeme v SNMP použít.

MIB se dělí na dvě hlavní části. **Standard MIB**, která je definována organizací IETF a popisuje univerzální vlastnosti, které většina výrobců implementuje. A **Enterprise MIB**, kde organizace IANA přiděluje unikátní podstromy (OID) jednotlivým společnostem, které si následně sami spravují.




SNMP je jednoduché a z toho pramení i jeho síla, i když se dnes často probírá otázka bezpečnosti. Pro SNMP existuje velké množství aplikací a můžeme jej použít na mnoha místech řadou různých způsobů. Společné je to, že základní použití je velmi jednoduché.

Hodnoty můžeme načítat či nastavovat pomocí skriptů v prostředí Windows, Linux i dalších. Asi nejrozšířenější pro tento účel je skupina open source aplikací pro příkazovou řádku po názvem **Net-SNMP**. Ta se skládá z řady malých aplikací, které v základu reprezentují jednotlivé operace SNMP. Příklad, kdy ze síťového zařízení načítáme jeho `UpTime`, je uveden níže. Návratová hodnota je formátu `Timeticks`, takže ji Net-SNMP převádí i do čitelnějšího formátu.

```
c:\NetSNMP\bin>snmpget -v 2c -c public 192.168.100.21 .1.3.6.1.2.1.1.3.0

DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (1500335577) 173 days, 15:35:55.77
```

Další možností je vytvoření jednoduchého i složitějšího programu. Vystačíme i se základními znalostmi programování a můžeme vytvořit vlastní aplikaci například v jazyce PHP či Java. Jednoduchý příklad v jazyce PHP a jeho výstup.

```
<?php echo snmpget("192.168.100.21", "okro", ".1.3.6.1.2.1.1.3.0"); ?>
```

Po spuštění dá výstup:

```
Timeticks: (1500372420) 173 days, 15:42:04.20
```

Většina monitorovacích řešení využívá jako jednu z možností monitoringu SNMP. Také existují jednoúčelové aplikace, které načítají údaje ze zadaného OID a zobrazují je do grafu či tabulky. Pro testování SNMP existuje

řada jednoduchých nástrojů, jako je **KS-Soft MIB Browser** či **Paessler SNMP Tester**.



Autor: **Petr Bouška**

**Související články:**

## Monitoring sítě

Úvod do monitoringu počítačových sítí a popis základních protokolů.

- **Začínáme s monitoringem sítě** [01.09.2009 12:21]
- **Zařízení v síti pod kontrolou** [21.09.2009 17:08] ..... **právě čtete**
- **Automatizovaná kontrola prostředí pomocí PowerShellu** [13.06.2011 17:35]

# Automatizovaná kontrola prostředí pomocí PowerShellu

Pondělí, 13.06.2011 17:35 | Samuraj - Petr Bouška | Microsoft admin

aneb Pár tipů jak využít PowerShell. Každá síť potřebuje dohled (monitoring, kontrolu) a čím je větší, tím je složitější provádět různé kontroly ručně. Existuje řada systémů, které tyto kontroly budou provádět za nás, ale ty komplexní většinou stojí nemalé peníze. Přitom se dá vše postavit zdarma s menším či větším úsilím. Důležité je si uvědomit, co nás v naší síti zajímá, a na co chceme být upozorněni. Pak můžeme vytvářet jednoúčelové skripty, které nám zajistí potřebný přehled.



*Tento článek jsem napsal pro Microsoft TechNet Blog CZ/SK, kde vyšel jako Automatizovaná kontrola prostředí pomocí PowerShellu a zároveň byl publikován v Microsoft sekci na Živě Automatizovaná kontrola prostředí pomocí PowerShellu.*

Základem monitoringu, ale stále zůstává ***odesílání důležitých událostí*** ze serverů na jedno centrální místo, kde se budou kontrolovat a vybrané události odesílat třeba na email. Tak můžeme kontrolovat chybné pokusy o přihlášení, zamčení účtu, vytvoření nějakého objektu v Active Directory, restart serveru, apod.

Druhou nezbytnou oblastí je nějaký ***standardní monitoring***, který pravidelně kontroluje dostupnost serverů a vybraných služeb. Tak zjistíme kompletní pád serveru, nedostupnost služby (jako například smtp protokol na poštovním serveru), nedostatek místa na disku, dlouhodobé vytížení procesoru, apod.

Dnes se ale podíváme na něco jiného. Můžeme vytvořit jednoduché (samozřejmě i složité) ***skripty v PowerShellu***, které se automaticky spustí každý den (či jinou definovanou periodu) a odešlou nám určitý ***report***. Potom krátkým pohledem zjistíme stav v určité oblasti. Podobně můžeme vytvořit řadu ***pravidelných kontrol***, které nám pošlou zprávu, pokud je něco v nestandardním stavu. V dalších řádcích si ukážeme pár praktických příkladů, jak toto využít na Microsoftím prostředí.

## Úvod ke skriptům v PowerShellu

Nebudeme se zde věnovat základům PowerShellu, ale pokud je neznáte, tak se nemusíte obávat. Použití není nikterak složité a možná zjistíte, že je to opravdu užitečný nástroj. Ani dále v článku nebudeme rozebírat jednotlivé příkazy PowerShellu, ale ukážeme si pár jednoduchých praktických příkladů. Ty budou vždy provádět jednu činnost a z jejich zápisu určitě pochopíme význam jednotlivých částí.

### Spouštění PowerShell skriptů

Standardně v operačních systémech není povoleno ***spouštět PowerShell skripty***, jde o bezpečnostní nastavení zvané Execution Policy. Toto nastavení můžeme v systému změnit pomocí *PowerShell příkazu* nebo pomocí *Group Policy* (toto nastavení znemožní změnu příkazem). Aktuální nastavení zjistíme příkazem:

```
Get-ExecutionPolicy
```

Nemáme zde prostor vysvětlovat možnosti a bezpečnostní dopady. Takže jednoduše, abychom mohli spustit lokální nepodepsaný skript, tak musíme zadat příkaz:

```
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned
```

Pokud chceme spouštění povolit pro celou skupinu serverů či na celou firmu, tak je výhodnější použít ***Group Policy***. Nastavení se nachází v položce:

```
Computer Configuration/Policies/Administrative Templates/Windows Components/Windows PowerShell/Turn on Script Execution
```

Abychom tuto položku měli k dispozici, tak musíme mít v systému (nebo v Central Store) nahraný soubor `PowerShellExecutionPolicy.admx`, který obsahuje tuto šablonu. Ten se nachází pouze na Windows Server 2008 R2 a ne třeba ve Windows 7 (ale je možno jej zkopírovat ze serveru nebo stáhnout z webu MS).

### Načítání modulů a snap-in

V následujících příkladech budeme pracovat s ***ActiveDirectory*** a k tomu využijeme ***PowerShell modul*** od Microsoftu, který je součástí (mimo jiné) Windows Server 2008 R2. A také s ***Exchange Server 2007***, který obsahuje ***Snap-In pro PowerShell***. Abychom mohli použít ***cmdlety*** z modulů a snapinů v čistém PowerShellu (powershell.exe), tak je musíme nejprve zavést (načíst). Informace o načtených modulech a snapinech získáme pomocí příkazů:

```
Get-Module
Get-PSSnapin
```

Načtení ***ActiveDirectory modulu*** provedeme příkazem:

```
Import-Module ActiveDirectory
```

Načtení ***Exchange Server 2007 Snap-Inu*** provedeme příkazem:

```
Add-PSSnapin Microsoft.Exchange.Management.PowerShell.Admin
```

### Plánování skriptů

Standardně se ***PowerShell Script*** ukládá do souboru s příponou **ps1**. Spustit jej můžeme pomocí příkazu (důležité je zadání cesty, i když to lze samozřejmě i jinak):

```
powershell c:\Skripty\skript.ps1
```

Pro automatické spouštění využijeme ***plánovač úloh*** z Windows (Task Scheduler). Ve ***Windows Server 2008*** přidáme na záložce *Action* událost *Start a program* a jako script dáme `powershell`, jako argument dáme `c:\Skripty\skript.ps1`. Na záložce *Triggers* přidáme plánovaný úkol a zadáme požadovanou periodu spouštění. Na záložce *General* nastavíme, pod jakým účtem skript poběží. Na ***Windows Server 2003*** je situace obdobná, pouze do políčka *Run* zadáme celý příkaz `powershell c:\Skripty\skript.ps1`.

## Kontrola účtů v Active Directory

Nejen z bezpečnostního hlediska je dobré mít v Active Directory pořádek. Přitom některé informace se dozvíme pouze položením určitého dotazu. To můžeme automatizovat skriptem, spouštět pravidelně každou noc a na email si odeslat výsledek.

> ***Pozn.:*** *V prvním příkladu popíšeme celý skript, v dalších již budeme vynechávat načtení modulu na začátku a odeslání emailu na konci. Skript vždy uložíme do nějakého souboru s příponou ps1 na doménovém řadiči a budeme automatizovaně spouštět pomocí Task Scheduler pod účtem s dostatečným oprávněním (stačí čtení AD). Samozřejmě můžeme použít více skriptů najednou a odeslat jeden společný email.*

### Neaktivní uživatelé

Například nás může zajímat seznam uživatelů, kteří se nepřihlásili nějakou definovanou dobu (zde 30 dní) do domény. Tím můžeme odhalit, že jsme nějaký účet zapomněli smazat nebo již přestal být potřeba.

Ve skriptu nejprve načteme modul. Druhý příkaz získá seznam uživatelských účtů, které se za posledních 30 dní nepřihlásili, ten seřadí podle abecedy, vrátí pouze vybrané sloupce a uloží do souboru. Poslední příkaz odešle informační email a do přílohy vloží vygenerovaný soubor.

```
Import-Module ActiveDirectory
Search-ADAccount -AccountInactive -UsersOnly -TimeSpan "30" | Sort-Object Name | FT Name,LastLogonDate,Enabled,LockedOut -Aut
Send-MailMessage -From ad@firma.cz -To monitoring@firma.cz -Subject "Kontrola uživatelů v AD" -SmtpServer mail.firma.local -A
```

### Neaktivní počítače

Stejným způsobem můžeme nalézt počítače, které se 30 dní nepřihlásili.

```
Search-ADAccount -AccountInactive -ComputersOnly -TimeSpan "30" | Sort-Object Name | FT Name,LastLogonDate,Enabled,LockedOut
```

### Zakázané účty uživatelů a počítačů

Můžeme vytvářet seznam počítačových a uživatelských účtů, které jsou zakázané (disabled). Tuto informaci asi nepotřebujeme každý den, ale můžeme ji pro přehled přidat k předchozím.

```
Search-ADAccount -AccountDisabled -UsersOnly | Sort-Object Name | FT Name -AutoSize | Out-File -FilePath "c:\Skripty\disabled

Search-ADAccount -AccountDisabled -ComputersOnly | Sort-Object Name | FT Name -AutoSize | Out-File -FilePath "c:\Skripty\disa
```

### Zamčené uživatelské účty

Pokud máme politiku na zamykání účtů, tak nás může zajímat seznam zamčených účtů. Uživatelé by se asi ozvali, ale například se nemusíme dozvědět, že se zamknul nějaký servisní účet.

```
Search-ADAccount -LockedOut -UsersOnly | Sort-Object Name | FT Name -AutoSize | Out-File -FilePath "c:\Skripty\locked-users.t
```

### Nové počítačové účty

Můžeme vytvářet report, který bude obsahovat seznam nových, smazaných nebo přejmenovaných počítačových účtů. Tyto informace můžeme získat i z auditování AD DS, ale takto dostaneme přehledný report.

Nejprve načteme předchozí stav (takže při prvním spuštění nám celý skript nezafunguje). Potom získáme seznam všech počítačů a seřazený jej uložíme do souboru. Tento seznam znovu načteme do jiné proměnné (kvůli typu a následnému porovnání). Porovnáme tyto dva seznamy a jejich rozdíl uložíme do souboru (tento soubor pak můžeme odeslat jako email). Ve výsledku vidíme nové záznamy (indicator =>) a chybějící záznamy (indicator <=), pokud došlo ke změně jména, tak bude zobrazeno jako odstraněný a nově vložený.

```
$yesterday = Get-Content c:\skripty\computers.txt
Get-ADComputer -Filter * | Sort-Object Name | FT Name -AutoSize | Out-File -FilePath c:\skripty\computers.txt
$today = Get-Content c:\skripty\computers.txt
Compare-Object $yesterday $today | FT -AutoSize | Out-File -FilePath c:\skripty\computers-changes.txt
```

## Kontrola poštovních schránek na Exchange serveru

Tak jako informace z domény nás může zajímat řada informací na poštovním serveru, hlavní se týkají poštovních schránek (mailboxů).

> ***Pozn.:*** *Opět první skript bude kompletní a dále jen hlavní část. Skript bude v souboru s příponou ps1 a plánovat jej budeme přímo na Exchange serveru (nyní musí mít účet práva na Exchange).*

### Poštovní schránky s překročeným limitem

Na velikost poštovní schránky se může nastavit několik limitů, při prvním se uživateli posílá upozornění, ale jinak může standardně pracovat. Pokud nás zajímá seznam schránek, které překročili tento limit, tak můžeme použít následující skript.

Nejprve načteme snapin. Potom vezmeme seznam mailboxů, odstraníme z něj ty, které mají velikost pod limitem, seřadíme podle velikosti a vybrané sloupce uložíme do souboru. Ten pak odešleme emailem.

```
Add-PSSnapin Microsoft.Exchange.Management.PowerShell.Admin
Get-MailboxStatistics | sort TotalItemSize -Descending | where { $_.StorageLimitStatus -ne "BelowLimit" } | FT DisplayName,To
Send-MailMessage -From exchange@firma.cz -To monitoring@firma.cz -Subject "Kontrola velikosti poštovních schránek" -SmtpServe
```

## Veřejné složky s překročeným limitem

Obdobně můžeme kontrolovat velikost veřejných složek (Public Folders) a vytvořit seznam těch, které jsou větší než zadaná hodnota (zde cca 500MB). Jen je třeba si uvědomit, že podsložky ve veřejných složkách se počítají samostatně.

```
Get-PublicFolderStatistics | Sort TotalItemSize -Descending | where { $_.TotalItemSize -gt 500000000 } | FT Name,FolderPath,I
```

## Přehled mobilních zařízení

Pokud používáme Exchange ActiveSync, tak se mobilní zařízení, která se serverem synchronizujeme, spárují s účtem a na serveru se o nich vytvoří záznam. K dispozici máme příkaz, který nám napíše několik informací. Uživatelé si mohou připojit různé přístroje a nás může zajímat, že přibylo nové zařízení či obecně přehled o nich.

Můžeme opět naplánovat úlohu, která zjistí seznam všech zařízení, a vytvoří rozdíl oproti předchozímu dni, takže vidíme změny.

```
$yesterday = Get-Content c:\skripty\ActiveSync.txt
Get-Mailbox | ForEach { Get-ActiveSyncDeviceStatistics -Mailbox:$_.Identity } | FT Identity,DeviceType,DeviceUserAgent,Device
$today = Get-Content c:\skripty\ActiveSync.txt
Compare-Object $yesterday $today | Out-File -FilePath c:\skripty\ActiveSync-differences.txt -Width 400
```

## Kontrola komunikace

V řadě situací můžeme využít test, jestli přišel nějaký určitý email. Například máme vzdálený systém, který není přímo dostupný z internetu, ale může odesílat emaily. Necháme tedy jednou za hodinu posílat emailový keepalive a máme hrubou kontrolu, že systém žije.

```
$mails = Get-MessageTrackingLog -EventID "RECEIVE" -Sender "server@nekde.cz" -Recipients "monitor@firma.cz" -Start (Get-Date)
if($mails -eq $null) { Send-MailMessage -From dohled@firma.cz -To monitoring@firma.cz -Subject "Kontrolní email ze serveru" -
```

# Systémové kontroly

Ještě se podíváme na dva příklady, které nespadají do předchozích kategorií. Navíc se jedná o malinko složitější skripty.

## Kontrola zálohy

Pokud provádíme nějaké jednoduché zálohy, například dump z linuxového serveru, který v taru kopírujeme na souborový server s Windows, tak můžeme testovat, jestli se v danou dobu soubor vytvořil a jestli není jeho velikost nulová.

Do proměnné `$files` si uložíme seznam souborů (bez adresářů), které byly změněny (tedy i nově vznikly) za poslední den v adresáři `c:\backup` (mohli bychom hledat pouze soubory určitého jména). Potom porovnáme, jestli je v seznamu alespoň jeden soubor (nemůžeme použít jednodušší `$files.Count`, protože nefunguje na prázdný seznam). Pokud nějaké soubory existují, tak ověříme, jestli nemají nulovou velikost (můžeme testovat zadanou minimální velikost). Pokud došlo k problému, tak odešleme informaci emailem.

```
$message = ""
$files = Get-ChildItem c:\backup | Where-Object { $_.LastWriteTime -gt (Get-Date).AddDays(-1) -and ! $_.PSIsContainer }
if(($files | Measure-Object).Count -eq 0) {
   $message = "Za poslední den nevznikl žádný soubor."
} else {
   $files | ForEach-Object { if($_.Length -eq 0) { $message += "Soubor " + $_.name + " má nulovou velikost. " } }
}
if($message -ne "") { Send-MailMessage -From zalohovani@firma.cz -To monitoring@firma.cz -Subject "Zálohování selhalo" -SmtpS
```

## Změny na DHCP serveru

Sice je řada možností, jak zabezpečit připojení "cizího" zařízení do sítě, ale dokonalé zabezpečení je většinou v praxi nemožné. Pro zvýšení našeho přehledu o zařízeních v síti si můžeme posílat report zařízení, která dostala IP adresu od DHCP. Není to myšleno jako metoda odhalení nepovolených zařízení v síti, také bychom mohli využít auditování DHCP serveru, je to pouze ukázka možností.

Ani v poslední verzi Windows serveru neobsahuje PowerShell cmdlety pro ovládání DHCP serveru. Nic nám ale nebrání využít z PowerShellu klasický příkaz `netsh` (v neinteraktivním režimu) a zpracovat jeho výsledky. Takže si vypíšeme seznam aktivních DHCP Scope, a z nich vybereme prvních 15 znaků (což je síťová IP adresa) a uložíme do proměnné. Potom pro každou adresu scopu vypíšeme seznam pronajatých adres (lease). Výsledek uložíme do souboru. Porovnáváme stav z předchozího dne s aktuálním a výsledkem jsou změny.

```
$leases_yesterday = Get-Content c:\skripty\dhcp.txt
$scopes = netsh dhcp server \\pdc show scope | Where-Object { $_.Contains("Active")  } | ForEach-Object { $_.Substring(0, 15)
$leases = $scopes | ForEach-Object { netsh dhcp server \\pdc scope $_.Trim() show clients 1 } | Where-Object { $_.Contains("-
$leases | Out-File -FilePath c:\skripty\dhcp.txt
Compare-Object $leases_yesterday $leases | FT -AutoSize | Out-File -FilePath d:\dhcp-changes.txt
```

# Závěr

Pokud jste dočetli až sem, tak vám děkuji a doufám, že pro vás měl článek nějaký přínos. Jeho cílem bylo ukázat, že můžeme jednoduše získat přehled o tom, co se děje v našem prostředí. A nebude nás to stát příliš času ani peněz. Potom také, že PowerShell není špatná ani složitá technologie. Hlavní je rozmyslet, o čem v síti chceme být informováni. Sestavení skriptu již nemusí být složité a hodně nám pomůže Google. Samozřejmě příklady v článku jsou docela triviální, daly by se vylepšit, aby měly větší inteligenci a to by obnášelo trochu více kódu a složitosti.



Autor: Petr Bouš

# SNMP - Simple Network Management Protocol

Středa, 20.12.2006 15:09 | Samuraj - Petr Bouška | sítě

Téměř každý slyšel o protokolu SNMP, o tom jak je výhodný a jednoduchý a jak je dobré jej využívat pro monitoring sítě. Ale již ne každý pořádně ví, o co se jedná a jak to funguje. Když jsem se podíval po internetu, tak jsem nenalezl žádný článek, který by stručně a jasně vysvětlil, oč se jedná. Proto jsem se o to pokusil já.



**SNMP** je jednoduchý, široce rozšířený a užitečný standardizovaný protokol, který slouží k získávání nebo nastavování hodnot na určitém zařízení. Obdobou je například **WMI** od firmy *Microsoft*. Podporu **SNMP** má velká řada zařízení, například aktivní síťové prvky, počítačová čidla, tiskárny, přístupové body nebo pomocí softwaru a ovladačů ji mohou získat osobní počítače a servery. Hodnoty můžeme získávat v pravidelném intervalu a ty pak jednoduše ukládat do databáze spolu s časem a následně vykreslit do grafu. Přehledně tak můžeme zobrazit třeba vytížení procesoru, průběh teploty nebo datový tok na portu přepínače.

## Jak SNMP funguje

Protokol **SNMP** vyžaduje pro komunikaci dvě strany. Jednou entitou je **správce** (manager) a druhou **agent**. SNMP pracuje ve dvou režimech činnosti:

- ***Správce posílá dotazy agentovi a přijímá odpovědi.*** Hodnoty tedy může získávat i více správců a mohou se ptát kdykoliv.
- ***Agent zasílá oznámení (trapy) na adresu správce.*** V nějakých definovaných situacích (překročení nějaké hodnoty nebo i v pravidelném intervalu) odesílá agent jednomu správci hodnoty.

Protokol SNMP nyní existuje ve třech verzích. **SNMPv1** a **SNMPv2c** používají pro autentizaci ***community string***, v podstatě textové heslo. V **SNMPv3** je možno využít autentizaci pomocí *jména* a *hesla* a *šifrování*.

**SNMP** používá pro komunikaci **UDP** protokol, díky čemuž je velmi rychlé, ale může dojít ke ztrátě (nedoručení) zasílané informace (paketu). Od ***verze 2*** je implementována kontrola doručení, takže ke ztrátě by nemělo dojít. Standardně se používá **port 161** (SNMP) na straně ***agenta*** (pro dotazy) a **port 162** (SNMPTRAP) na straně ***serveru*** (pro trapy). Klient, který posílá dotaz, zvolí dynamický port, z kterého posílá dotaz na port 161. Agent odpovídá z portu 161 na dynamický port klienta. V praxi je pro každý dotaz použit jiný dynamický port.

## SNMP paket

**Dotaz a odpověď**

| verze | community string | PDU typ | ID dotazu | error status | error ID | OID | hodnota |
|---|---|---|---|---|---|---|---|

***Příklad***

| 1 | public | GET (0) | 8 | no error (0) | 0 | 1.3.6.1.4.1.311.1.1.3.1.1.1 | NULL |
|---|---|---|---|---|---|---|---|

**Pro trapy**

| verze | community | PDU typ | enterprise | agent IP | gen trap | spec trap | čas | objekt 1 hodnota 1 | ... |
|---|---|---|---|---|---|---|---|---|---|

***Community string*** je heslo pro SNMPv1 a SNMPv2c, ***PDU typ*** je typ SNMP dotazu.

## SNMP dotazy

Klasická komunikace, kdy posílám dotaz na jednu hodnotu a následně přijímám odpověď, probíhá následovně:

- **odešle se dotaz** – nastaví se typ GET, zadá se OID pro zjišťovanou hodnotu, vlastní hodnota se nastaví na NULL
- **vrátí se odpověď** – typ je nastaven na RESPONSE (2), OID na dotazovanou hodnotu a je vyplněna hodnota

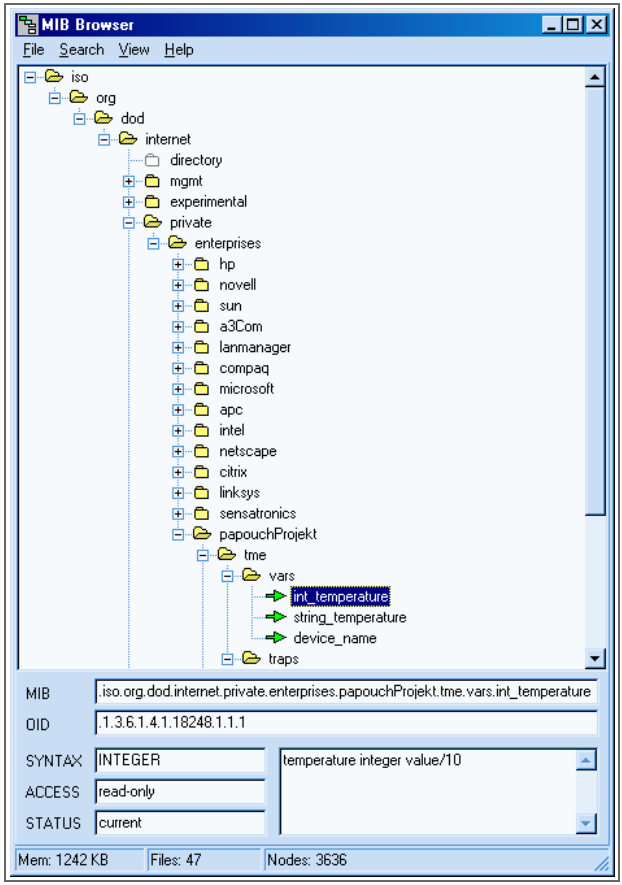
Další možné typy dotazů jsou **SNMP GET** (0) – vrátí jednu hodnotu, **GET-NEXT** (1) – vrátí další hodnotu (vezme následující OID za zadaným). Od verze 2 **GET-BULK**, který vrací více hodnot najednou (například pro všechny porty přepínače). Pomocí SNMP můžeme hodnoty nejen číst, ale také ***zapisovat***, k tomu slouží typ **SET**. ***Trapy*** mají trochu jiný formát paketu, jako typ je **TRAP**. Ještě jsou speciální typy ***NOTIFICATION***, ***INFORM*** a ***REPORT***.

## MIB databáze – Management Information Base

Každá **hodnota** v SNMP je jednoznačně identifikována pomocí číselného identifikátoru **OID** - ***Object Identifier***. OID je tvořeno posloupností čísel oddělených tečkou, tato hodnota vznikne tak, že se vezme OID nadřazeného prvku a doplní se tečka a aktuální číslo. Celá tato stromová struktura je uložena v **MIB databázi**. Navíc ***MIB databáze*** obsahuje *jména* a *popisy* jednotlivých hodnot (OID). ***MIB databáze*** může být doplněna o další hodnoty pomocí části struktury uložené v ***MIB souboru***.

***Příkladem OID*** může být třeba hodnota **1.3.6.1.2.1.2.2.1.6.1**, které odpovídá textová verze z *MIB databáze* **iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifPhysAddress**.

Pro vlastní práci se **SNMP** nepotřebujeme ***MIB databázi***, v *SNMP paketu* figuruje pouze **OID**, ale pokud neznáme správné OID, tak nám databáze může pomoci k jeho dohledání.




## Software pro SNMP

**MIB Browser** je velice dobrý a užitečný program pro prohlížení *MIB databáze* s možností přidávání *MIB souborů* a dotazů na zařízení. Druhým užitečným programem je **Paessler SNMP Tester**, který sice neobsahuje MIB databázi, ale má zase několik funkcí navíc. Dovede provádět dotazy v SNMPv1, SNMPv2c i SNMPv3 a také má užitečnou funkci ***SNMP WALK*** pro výpis všech OID daného zařízení.

- **nástroje pro SNMP**
  - **MIB Browser**
  - **Paessler SNMP Tester**
  - **Net-SNMP** - SNMP příkazy do command line
  - **Getif** - různé nástroje
- **popis SNMP**
  - **SNMP tutorial**
  - **Essential SNMP**
  - **SNMP MIB**
- **MIB soubory různých výrobců**
  - **ByteSphere's MIB Download Area**
  - **mibDepot**
- **MIB validátor**
  - **Online MIB Validator**



Autor: **Petr Bouška**

**Související články:**

### SNMP

Protokol SNMP (Simple Network Management Protocol) je velice užitečný pro správu počítačové sítě.

- **SNMP - Simple Network Management Protocol** [20.12.2006 15:09] ..... **právě čtete**
- **CACTI - SNMP monitoring a grafy** [07.01.2007 12:51]
- **Informace o portech switche pomocí SNMP a PHP** [17.03.2007 19:37]
- **Informace o zařazení portu do VLANy u Cisco Switchů pomocí SNMP a PHP** [25.03.2007 19:02]
- **Zjištění MAC adresy zařízení na portu switche pomocí SNMP a PHP** [20.03.2008 11:21]

# Informace o portech switche pomocí SNMP a PHP

Sobota, 17.03.2007 19:37 | Samuraj - Petr Bouška | administrace

V článku se snažím přiblížit možnosti praktického využití SNMP ve spojení s programovacím jazykem PHP. Příklad, který jsem zvolil je získání informací o portech/interfacech switche. Nejsou potřeba hluboké znalosti ani SNMP ani PHP k tomu abychom dosáhli zajímavé funkčnosti. Myslím, že hlavní je nápad (nebo informace) co vše můžeme dokázat.



Popisovaný příklad vychází z praktické situace, kterou řeším. Využívám description portu, kam zapisuji co je do daného portu zapojeno - označení zásuvky (když je připojena zásuvka) nebo jméno počítače (pokud je přímo připojen do portu). Potom je jednoduché zjistit, kam port vede (například, když sleduji nějakou podezřelou MAC adresu v síti), aniž bych musel procházet kabely. Občas, ale potřebuji opačnou věc, pro určitou zásuvku nalézt switch a port, kam je zapojena. Proto potřebuji seznam všech portů a switchů, který se však ručně složitě udržuje. A tady nastupuje myšlenka získávat tyto informace dynamicky pomocí SNMP, na což mě přivedlo Cacti, kde v podstatě tato možnost již je.

## PHP a podpora SNMP

> ***Pozn.:*** *Používám PHP (a Apache) pod Windows, takže můj popis tomu odpovídá. Pod Linuxem jsou některé věci odlišné.*

V programovacím jazyce **PHP** máme k dispozici extension **php_snmp**, které nabízí základní SNMP komunikační příkazy jako `snmpget`, `snmpset` a `snmpwalk`. Některé další příkazy jsou dostupné až od verze PHP 5. Podrobný popis naleznete v dokumentaci http://www.php.net/manual/en/ref.snmp.php .

Omezením je, že tyto příkazy používají pouze ***SNMP verze 1***. Našel jsem různé odkazy na nedokumentované funkce `snmp2_get`, `snmp3_get`, apod., které by měly používat SNMPv2c a SNMPv3. Ale tyto funkce v mé verzi PHP nefungují.

PHP má ve svém adresáři podadresář **mibs**, který obsahuje běžné ***MIB tabulky***. Může však nastat problém, že PHP tento adresář nenajde a při použití SNMP vypisuje varování (a samozřejmě pak MIB databázi nevyužívá). Řešením je vytvořit ***systémovou proměnnou*** (Control Panel - System - Advanced - Environment Variables - System variables) se jménem **MIBDIRS** a hodnotou je **cesta** k adresáři s MIB soubory.

> ***Pozn.:*** *Například v EasyPHP není adresář s MIB soubory obsažen, a proto je třeba jej vykopírovat z běžné instalace PHP.*

Výhodou, když používáme ***MIB tabulky*** je, že návratové hodnoty jsou podle nich upraveny a například výčtový typ má doplněno jméno (to je například u typu interfacu).

V PHP pro SNMP existuje také řada funkcí, které nastavují chování hodnot, například, že se OID zobrazují číselně nebo výsledná hodnota neobsahuje doplňující informace (jako třeba typ - Gauge32: 100). Bohužel ne vše vždy správně funguje.

Druhou možností pro použití SNMP je mít nainstalováno **Net-SNMP** a volat jeho příkazy z PHP pomocí funkce `exec` a přebírat výstup. V tomto případě můžeme využít všechny verze SNMP a nemáme problém s verzemi PHP (které obsahují pouze některé funkce).

## SNMP pro interfacy

SNMP hodnoty pro interfacy jsou součástí standardní MIB tabulky **IF-MIB**. Strom hodnot začíná na **OID = 1.3.6.1.2.1.2**, zápis pomocí jmen uzlů je `iso.org.dod.internet.mgmt.mib-2.interfaces`. První hodnotou je počet interfaců.

| jméno | OID | popis |
|---|---|---|
| ifNumber | .1.3.6.1.2.1.2.1.0 | počet interfaců daného zařízení |

Dále je tabulka, která obsahuje určité vlastnosti pro každý interface. Zajímavé jsou například

| jméno | OID | popis |
|---|---|---|
| ifIndex | .1.3.6.1.2.1.2.2.1.1 | indexové číslo interfacu |
| ifDescr | .1.3.6.1.2.1.2.2.1.2 | pojmenování portu |
| ifType | .1.3.6.1.2.1.2.2.1.3 | typ portu |
| ifSpeed | .1.3.6.1.2.1.2.2.1.5 | rychlost portu |
| ifOperStatus | .1.3.6.1.2.1.2.2.1.8 | stav portu (up, down, ...) |
| ifInUcastPkts | .1.3.6.1.2.1.2.2.1.11 | příchozí unikástové pakety |

Některé další hodnoty nalezneme na trochu jiném místě, pod **OID = 1.3.6.1.2.1.31**, `iso.org.dod.internet.mgmt.mib-2.ifMIB`

| jméno | OID | popis |
|---|---|---|
| ifName | .1.3.6.1.2.1.31.1.1.1.1 | zkrácené jméno portu |
| ifAlias | .1.3.6.1.2.1.31.1.1.1.18 | popis (alias) portu |

Informace o IP adresách (ty mohou být přiřazeny samozřejmě jen někde, např. pro VLANy) jsou pod **OID = 1.3.6.1.2.1.4**, `iso.org.dod.internet.mgmt.mib-2.ip`

| jméno | OID | popis |
|---|---|---|

| | | |
|---|---|---|
| ipAdEntAddr | .1.3.6.1.2.1.4.20.1.2 | IP adresy |
| ipAdEntIfIndex | .1.3.6.1.2.1.4.20.1.1 | indexy interfaců pro IP adresy |

Pro switch jsou také zajímavé globální informace jako jméno a umístění. To jsou hodnoty, které můžeme zadat téměř pro každé SNMP zařízení při instalaci (nebo v nastavení). **OID = .1.3.6.1.2.1.1,** `iso.org.dod.internet.mgmt.mib-2.system`

| jméno | OID | popis |
|---|---|---|
| sysName | .1.3.6.1.2.1.1.5.0 | jméno zařízení |
| sysLocation | .1.3.6.1.2.1.1.6.0 | umístění |

Některé údaje, jako třeba ***počet interfacú*** nebo jméno zařízení jsou reprezentovány ***jednou hodnotou*** a můžeme proto využít funkci `snmpget`. Ostatní hodnoty, které jsou specifické pro každý interface, jsou uloženy tak, že za dané **OID** je připojen ***index*** interfacu a teprve zde je uložena hodnota. Protože tyto interfacy většinou neznáme a také proto, že je to efektivnější, můžeme využít funkci `snmpwalk`, která projde všechny hodnoty pod zadaným OID a vrátí nám ***pole hodnot***. Vhodnější je ještě funkce `snmprealwalk`, která vrací také OID pro danou hodnotu.

Můžeme také využít funkci `snmpwalk` tak, že ji zavoláme na OID pro interface a získáme všechny hodnoty pro všechny interfacy.

## Ralizace v PHP

```
// funkce, která naplní pole $interfaces hodnotami ze zařízení s adresou $ip a pro přístup se použije community string $comm
function getInterface(&$interfaces, $ip, $comm) {
  getInterPart($interfaces, $ip, $comm, ".1.3.6.1.2.1.2.2.1.1", "Index");
  getInterPart($interfaces, $ip, $comm, ".1.3.6.1.2.1.2.2.1.8", "Status");
  getInterPart($interfaces, $ip, $comm, ".1.3.6.1.2.1.2.2.1.2", "Port_name");
  getInterPart($interfaces, $ip, $comm, ".1.3.6.1.2.1.31.1.1.1.18", "Descr");
  getInterPart($interfaces, $ip, $comm, ".1.3.6.1.2.1.2.2.1.3", "Type");
  $name = format_snmp_string(snmpget($ip, $comm, ".1.3.6.1.2.1.1.5.0"));
  addInterSwitcheVal($interfaces, "Switch", $name);
}
// do pole přidá položku, která je stejná pro všechny indexy
function addInterSwitcheVal(&$interfaces, $valname, $value) {
  foreach($interfaces as $val)
    $interfaces[$val["Index"]][$valname] = $value;
}
// načte jednu vlastnost pro všechny interfacy
function getInterPart(&$interfaces, $ip, $comm, $oid, $name) {
  $array = snmprealwalk($ip, $comm, $oid);
  foreach($array as $key => $val) {
    $index = ereg_replace('.*\.([0-9]+)$', "\\1", $key);  //index je posledni cislo z OID
    $val = format_snmp_string($val);
    $interfaces[$index][$name] = $val;
  }
}
// vytiskne pole interfaců jako tabulku
function printInterfaces($interfaces) {
  echo "<table>";
  echo "<tr><th>Switch</th><th>Index</th><th>Status</th><th>Port_name</th><th>descr</th><th>Type</th> tr>";
  foreach($interfaces as $val) {
    echo "<tr>";
    echo "<td>".$val["Switch"]."</td>";
    echo "<td>".$val["Index"]."</td>";
    echo "<td>".$val["Status"]."</td>";
    echo "<td>".$val["Port_name"]."</td>";
    echo "<td>".$val["Descr"]."</td>";
    echo "<td>".$val["Type"]."</td>";
    echo "</tr>";
  }
  echo "</table>";
}
// proměnná kam se ukládají výsledné hodnoty, jendá se o dvourozměrné pole - všechny interfacy (podle indexu) a načtené hodno
$interfaces = array();
$ip = "10.0.0.1"; $co = "public";
// IP adresa zařízení, community string
getInterface($interfaces, $ip, $co);
printInterface($interfaces);
```

Uvedený příklad je pouze nástin řešení. Je třeba jej obalit do ***HTML tagů***, aby se mohl použít jako webová stránka. Také by si zasloužil řadu vylepšení nebo se dá řešit úplně jiným způsobem. Výsledné hodnoty můžeme například ukládat do databáze, což stačí jednou za delší dobu, protože asi nedochází k tolik častým změnám. Standardně jsou návratové hodnoty obaleny do řady zbytečných informací (číselné hodnoty, MAC adresa ..), takže je vhodné je odstranit. Například Cisco switche vrací 3 typy interfaců - vlastní porty, VLANy a null interface. Takže zobrazovat třeba chceme pouze porty, což můžeme filtrovat podle `ifType`.

Můj příklad sloužil hlavně k tomu uvědomit si, jaké jsou možnosti a ukázat jak je jednoduché využívat SNMP z PHP.



Autor: Petr Bouška

**Související články:**

### SNMP

Protokol SNMP (Simple Network Management Protocol) je velice užitečný pro správu počítačové sítě.

- SNMP - Simple Network Management Protocol [20.12.2006 15:09]
- CACTI - SNMP monitoring a grafy [07.01.2007 12:51]
- Informace o portech switche pomocí SNMP a PHP [17.03.2007 19:37] ..... **právě čtete**
- Informace o zařazení portu do VLANy u Cisco Switchů pomocí SNMP a PHP [25.03.2007 19:02]
- Zjištění MAC adresy zařízení na portu switche pomocí SNMP a PHP [30.03.2008 11:21]

# TCP/IP - Routing - směrování

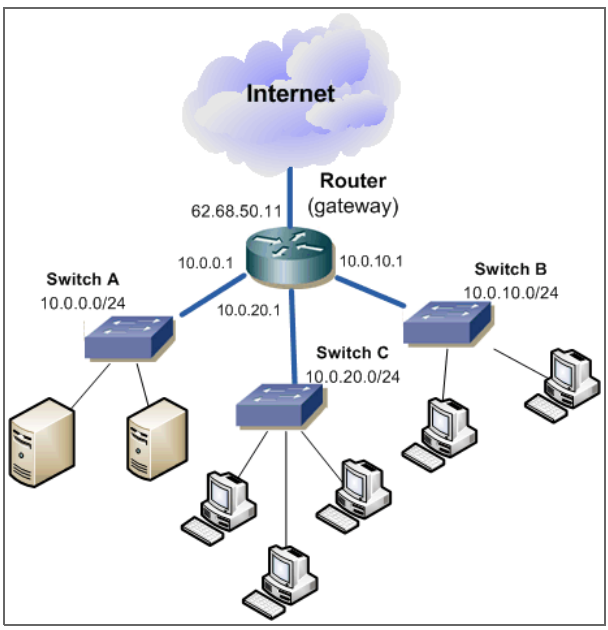
Pátek, 21.09.2007 15:34 | Samuraj - Petr Bouška | sítě

V jedenácté části seriálu o počítačových sítích se věnuji routingu, tedy směrování v sítích. Je zde stručný popis, vysvětlení termínů a pak jsou velmi stručně popsány některé běžnější routovací metody (RIP, IGRP, EIGRP a OSPF) včetně dělení těchto metod. U metod je ukázka základní konfigurace na Ciscu. Článek není zdaleka vyčerpávající a popis je často bodový. V závěru je zmíněno routování na Windows.



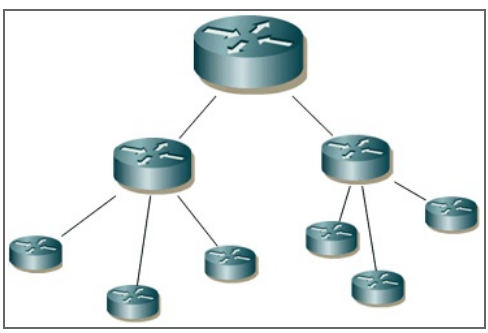
**Routing**, česky řečeno ***směrování***, ale častěji se používá slovo ***routování*** (alespoň v mém okolí). Jedná se o techniku, která slouží k propojení jednotlivých sítí (přesněji subnetů). Původním zařízením, určeným pro routování byl **router**, ale v dnešní době se velmi využívají ***L3 switche***, ***firewally*** nebo pouze ***servery***/počítače. Router přeposílá komunikaci z jedné sítě do jiné.

**N**ásledující obrázek ukazuje jednoduchý příklad sítě, kde máme subnety A, B a C. Tyto subnety jsou propojeny přes router mezi sebou a také do internetu. Takže pokud chce například stanice ze subnetu B komunikovat se serverem v subnetu A, tak pošle data na router a ten zařídí doručení do subnetu A. Pokud by stanice chtěla komunikovat do internetu, tak router naopak zašle data na jiný interface.




**D**ělení sítě na subnety je hierarchické a ve všech propojích musí být router. Při komunikaci pak postupujeme směrem nahoru ve stromu na nejbližší vrstvu, která propojuje dané subnety, a pak opět dolů. Délka cesty se počítá podle počtu **hopů** (skok), což je každý přechod ze zařízení na zařízení, je to tedy počet routrů v cestě + 1. Přímé spojení dvou počítačů je dlouhé 1 hop. Používá se také termín ***next hop*** (další skok) a označuje se jím adresa dalšího routeru v cestě.

**N**a dalším obrázku jsem se pokusil zachytit tuto situaci. Je zde malý (a pouze schematický) výřez větší sítě (či internetu). Na listech stromu jsou malé routery, ke kterým jsou připojeny switche a počítače. Tyto routery se sdružují do dalších (větších) a tak dále na několika úrovních. Samozřejmě v praxi jsou větší routery vždy redundantně, aby byla síť odolná vůči výpadku či kvůli vyvažování zátěže.



## Důležité pojmy pro routování

**Router**
zařízení, které provádí routování

**Routing**
routování, přeposílání (forwarding) dat mezi sítěmi

**Route**

cesta, která se použije, zapsaná v routovací tabulce

**Routing table**
routovací tabulka obsahuje záznamy o jednotlivých cestách

**Routing protocol**
routovací protokol slouží ke směrování routovaného protokolu, určuje nejlepší cestu k cíli a posílá routovací informace dalším routrům

**Routed protocol**
routovaný protokol je IP, IPX nebo Apple Talk

Ještě jeden občas používaný termín, který je dobré znát.

**Router on stick**
je router, který je připojený do switche pomocí jednoho trunk portu - tzn. máme pouze jeden router a pouze jednu linku, což přináší velkou zátěž na router i linku a problémy při výpadku

## Dělení routovacích protokolů

Do routovací tabulky se vytváří několik typů záznamů cest (route), záleží na tom, jakým způsobem vznikly. Pakety jsou podle toho směrovány jedním ze základních způsobů routování:

- **statické routování** – ručně zadané cesty (záznamy v routovací tabulce), bezpečné a dobré, ale nereflektuje změny v topologii sítě
- **dynamické routování** – síť se automaticky přizpůsobuje změnám v topologii a dopravě, automaticky se vypočítávají cesty pomocí routovacího protokolu
- **defaultní routování** – pokud neexistuje jiná cesta, tak se použije defaultní

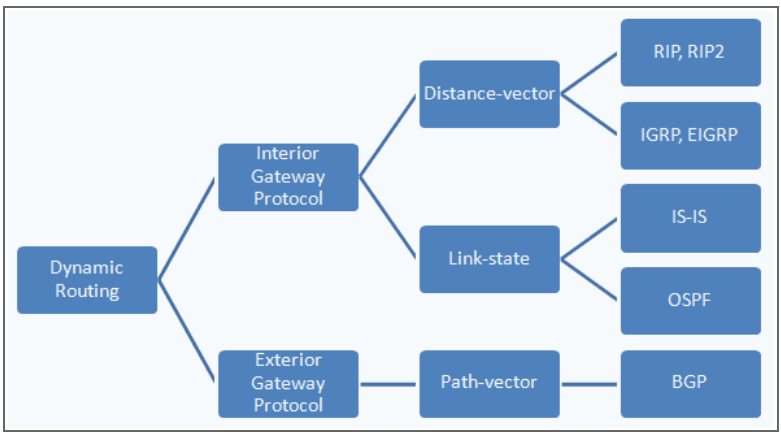
Dynamické routovací protokoly jsou dvou základních typů

- **distance-vector routing protocol** – routery udržují routovací tabulku s informací o (vektoru) vzdálenosti do dané sítě, periodicky ***routovací tabulku*** zasílají sousedům, ti si upraví svoji tabulku a tu opět odešlou dál, pro výpočet nejlepší cesty se používá jedna (počet hopů u RIP) nebo více metrik (propustnost linky a zpoždění u IGRP). Upraveným typem distance-vector protokolu je ***path-vector protocol***.
- **link-state routing protocol** – routery udržují komplexní databázi síťové topologie (vytvořenou pomocí LSA), vyměňují si ***link-state advertisements*** (LSA), LSA jsou vyvolány nějakou událostí v síti, do svého okolí také odesílá Hello pakety, kde zasílá informace o sobě, rychle reaguje na změny topologie, ale spotřebovává více pásma a zdrojů na routeru, metrika je komplexní, nejlepší cesta se počítá pomocí Dijkstrova algoritmu ***shortest path first*** (SPF)

> ***Pozn.:*** *Ještě je zde jeden speciální typ, který vychází z distance-vector protokolu a přidává některé vlastnosti link-state protokolu, označuje se jako* ***hybrid routing protokol*** *nebo* ***advanced distance-vektor protokol***. *Jeho jediným zástupcem je EIGRP.*

Dále dělíme dynamické protokoly podle toho, zda jsou určeny pro nasazení uvnitř lokální sítě (přesněji řečeno uvnitř autonomního systému (AS), který může obsahovat několik LAN) nebo fungují napříč sítěmi (spojují AS dohromady)

- **interior gateway protocol - IGP** – routuje uvnitř Autonomous System (AS)
- **exterior gateway protocol - EGP** – routuje mezi AS




## Obecné termíny

**Variable Length Subnet Masking (VLSM)**
používá se v Classless Inter-Domain Routing (CIDR). V tomto případě můžeme v subnetu použít různé velikosti masky. Můžeme například použít dohormady subnety 10.0.0.0/26 a 10.0.0.64/28.

**Autonomní systém - AS (Autonomous System)**
je skupina IP sítí a routrů, které jsou pod správou jedné (nebo více) jednotek.

**Administrativní vzdálenost - AD (Administrative Distance)**
je vlastnost používaná na routrech k určení nejlepší cesty mezi více routovacími protokoly. Definuje spolehlivost protokolu a prioritizuje lepší nižším číslem. Jinak řečeno na routeru může běžet více routovacích protokolů a podle AD se rozhoduje, který se použije. Na Cisco routrech můžeme měnit defaultní hodnoty.

| protokol | Administrativní vzdálenost |
|---|---|
| přímo připojený interface | 0 |
| statická routa | 1 |
| EIGRP | 90 |
| IGRP | 100 |
| OSPF | 110 |
| RIP | 120 |
| EGP | 140 |

**Split horizon**
metoda, která slouží k zamezení vzniku smyček v Distance Vector Routing protokolech. Používá se u RIP,

IGRP a EIGRP. Funguje tak, že zakazuje posílání routovací cesty zpět na rozhranní, z kterého se naučila.

**Hold-down timer**
je metoda, kterou používají routovací protokoly, aby zabránily zbytečnému nebo předčasnému rozesílání routovacích cest v nestabilním prostředí (ve chvíli, kdy dochází k časté změně stavu). Router čeká určitý čas než je síť stabilní.

## Jednotlivé routovací metody

### RIP - Routing Information Protocol

- jednoduchý pro konfiguraci a funguje všude
- pro malé a střední sítě
- RIP 1 nepodporuje VLSM
- plýtvá pásmem (velká režijní komunikace)
- pomalá konvergence (rozšiřování)
- hloupá metrika - počet hopů
- posílá celou routovací tabulku svým sousedům
- maximálně 15 hopů

Definují se pouze sítě, které jsou přímo na tomto routeru, ostatní se naučí pomocí updatů. Nastavení na Ciscu pomocí

```
SWITCH(config)#router rip
SWITCH(config-router)#network 132.43.54.0
SWITCH(config-router)#network 145.65.76.0
```

### IGRP - Interior Gateway Routing Protocol

- proprietární Cisco protokol
- nepodporuje VLSM
- jako metriku používá cenu, záleží na pásmu a zpoždění
- maximální počet hopů 255

Nastavení na Cisco routu (33 je číslo AS)

```
SWITCH(config)#router igrp 33
SWITCH(config-router)#network 134.43.54.0
SWITCH(config-router)#network 143.56.76.0
```

### EIGRP - Enhanced Interior Gateway Routing Protocol

- proprietární Cisco protokol
- rychlá konvergence
- redukuje spotřebu pásma pro routovací updaty
- podporuje různé protokoly (apple talk, IPX, IP) a VLSM
- metrika - pásmo, zpoždění (případně i zátěž, spolehlivost)
- routovací update se vyměňují pouze mezi routry ve stejném AS
- maximální počet hopů 255
- bez smyček

Používá *tabulku sousedů* (informace o přímých sousedech), *tabulku topologie* (všechny routovací záznamy, které se naučil) a *routovací tabulku* (nejlepší routy z tabulky topologie).

Nastavení na Cisco routeru (33 je číslo AS)

```
SWITCH(config)#router eigrp 33
SWITCH(config-router)#network 172.16.0.0
SWITCH(config-router)#network 10.0.0.0
```

### OSPF - Open Shortest Path First

- hierarchický systém - jedna nebo více oblastí je spojena k páteřní oblasti (oblast 0)
- routry posílají linkstate (pásmo a stav interfacu) informace všem sousedním routrům v oblasti
- routry vytvářejí databázi topologie, což je model celé oblasti
- z databáze se pomocí Dijkstry vypočítá nejkratší cesta a zapíše do routovací tabulky
- neomezený počet hopů
- určeno pro rozsáhlé sítě heterogenní sítě
- podporuje VLSM

Nastavení na Cisco routeru (1 je ID číslo procesu, pouze lokálně významné)

```
SWITCH(config)#router ospf 1
SWITCH(config-router)#network 132.43.56.0 0.0.0.255 area 0
SWITCH(config-router)#network 145.54.34.6 0.0.63.255 area 0
```

### Static routing

- používá se například mezi ISP a firmou, není zde třeba, aby běhal složitý routovací protokol
- pro každou síť je vložen záznam do routovací tabulky
- žádná zátěž
- pouze pro malé sítě

Nastavení na Cisco routru

```
ip route [destination_network] [mask] [next_hop or exit_interface] [administrative_distance] [permanent]
SWITCH(config)#ip route 192.168.50.0 255.255.255.0 192.168.1.1
```

***next_hop*** je IP adresa dalšího routeru v cestě, přesněji, je to adresa interfacu sousedního routeru, který sousedí s tímto routerem,
***exit_interface*** je jméno lokálního výstupního interfacu (třeba s0), přes který vede cesta k cílové síti.

### Default routing

- může se použít pouze na kraji sítě, kde je jeden port, který vede mimo síť
- pokud není definována jiná cesta k dosažení cíle, tak se použije defaultní routa, což je gateway

Nastavení na Cisco routru

```
ip route 0.0.0.0 0.0.0.0 [next_hop or exit_interface]
SWITCH(config)#ip route 0.0.0.0 0.0.0.0 62.102.58.12
```

## Routování a Windows

I pracovní stanice, kde je například OS Windows XP, musí provádět rozhodnutí o tom, kam směrovat síťová data. Vychází se z toho, jak jsou nastaveny síťové karty v počítači. Podle tohoto nastavení se dynamicky upravuje ***routovací tabulka***. Ale údaje v tabulce můžeme upravovat i ručně. Podle routovací tabulky se Windows rozhodují, přes kterou síťovou kartu zaslat určitá data.

***Základní routy*** jsou pro *gateway*, tedy to co nevím kam jinam poslat, označuje se jako ***default route***. Pak jsou routy pro *hosta* a *loopback* (127.0.0.0/8), které vedou na *MS TCP Loopback Interface*. Dále je routa pro *multicast* (224.0.0.0/4). V neposlední řadě je zde routa pro *lokální subnet*.

Jednotlivé routy mají ***metriku*** (1 až 9999), která určuje jejich prioritu. Čím nižší hodnota, tím větší priorita. Routa se vybírá nejprve tak, aby nejblíže odpovídala cílové adrese. Pokud je třeba, tak je druhým kritériem metrika.

Jak jsem psal, tak se routovací tabulky vytváří a upravuje dynamicky podle interfaců. Můžeme manuálně mazat, přidávat či upravovat záznamy, ale tyto změny se při restartu počítače ztratí. Pokud chceme vytvořit trvalou statickou routu, tak ji musíme vytvořit jako ***persistant***. Tyto routy jsou pak uloženy v registech `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\PersistentRoutes`.

Pro manipulaci s routovací tabulkou ve Windows XP (ale dalších MS OS) slouží příkaz `route`. Pár příkladů použití:

```
route [-f] [-p] [Command [Destination] [mask Netmask] [Gateway] [metric Metric]] [if Interface]]
route print                                          // vypíše routovací tabulku
route add 0.0.0.0 mask 0.0.0.0 192.168.12.1          // vytvoří dočasnou defaultní routu, tedy gateway
route -p add 10.41.0.0 mask 255.255.0.0 10.27.0.1 metric 7  // vytvoří trvalou (persistant) routu pro 10.41.0.0/16
route delete 10.41.0.0 mask 255.255.0.0              // smaže záznam pro 10.41.0.0/16
```

## Windows jako router

***Routování***, o kterém jsem tu psal, se týká pouze rozhodnutí, kam směrovat provoz ***generovaný na tomto počítači***. Pokud bychom chtěli použít počítač jako ***router***, tedy aby příchozí komunikaci posílal dál, tak musíme tuto funkci zapnout v registrech.

V registrech `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters` musíme změnit hodnotu položky `IPEnableRouter` na `1`.

> ***Pozn.:*** *Chtěl bych upozornit, že zapnutí této funkce je třeba dobře rozvážit, protože se může jednat o bezpečnostní riziko. Pomocí této funkce může dojít k propojení několika sítí (což asi v tomto případě chceme), takže ji nezapínat bezdůvodně.*

# Cisco IOS 18 - inter-VLAN routing a ACL - směrování mezi VLANy

Středa, 24.12.2008 11:50 | Samuraj - Petr Bouška | Cisco admin

Tento článek logicky navazuje na popis VLAN. Vezměme, že jsme si rozdělili naši LAN na VLANy (subnety), čímž jsme omezili broadcast domény, segmentovali síť a získali další výhody. Jenže teď bychom potřebovali, aby alespoň (nebo právě) některé VLANy mohli mezi sebou komunikovat. Potřebujeme tedy mezi jednotlivými VLANami routovat, tomu se říká inter-VLAN routing. Tento článek popisuje, jak takové routování provozovat a jak jej omezovat pomocí Access Control List (ACL).



V tomto článku budu vycházet z (myslím běžné) topologie, kdy máme switche ve dvou vrstvách (a ne doporučovaných třech). Je zde **jádro** (core), což je L3 switch (nebo dva kvůli redundanci), který provádí vlastní ***inter-VLAN routing***. Druhá vrstva je **přístupová** (access), do ní jsou zapojena všechna zařízení (stanice, v tomto případě i servery, tiskárny apod.). Všechny switche přístupové vrstvy jsou zapojeny do jádra, tyto spoje jsou konfigurovány jako **trunk** a používáme ***VLAN Trunking Protocol*** (VTP), takže stejné VLANy máme na všech switších.

> ***Pozn.:*** *Obdobně se dá použít i zapojení, kdy nevyužíváme L3 switch, ale router. Základní informace jsou v článku* ***VLAN - Virtual Local Area Network****.*

## Zapnutí routování mezi VLANami

Defaultně je switch v módu L2 switchování, pokud chceme použít L3 vlastnost IP routing, tak ji musíme zapnout.

```
SWITCH(config)#ip routing
```

Nepotřebujeme použít žádný ***routovací protokol***, protože veškeré routování se odehrává na jednom zařízení. Cisco IOS automaticky vkládá ***přímo připojené interfacy*** do routovací tabulky. Pokud je zapnuto routování, tak IOS routuje podle záznamů v routovací tabulce, což jsou statické routy a přímo připojené interfacy.

To je výhoda inter-VLAN routingu, že pro základní funkcionalitu není třeba téměř žádná konfigurace. VLAN interfacy jsou na core switchi ***přímo připojené interfacy***, mezi kterými se provádí routování automaticky. Musíme pouze pro **VLANy**, které chceme routovat, vytvořit ***VLAN interface*** a nastavit mu IP adresu. ***IP adresa*** je stejně nutná, protože se jedná o ***gateway*** (bránu) pro daný subnet.

Následující příklad vytváří VLAN interface pro VLAN 100, která má subnet 10.0.1.0/24 a chceme adresu gateway 10.0.1.1. VLAN 100 již máme vytvořenou.

```
SWITCH(config)#interface vlan 100
SWITCH(config-if)#ip address 10.0.1.1 255.255.255.0  // zadání IP adresy spolu s maskou, která určuje subnet
SWITCH(config-if)#no shutdown                         // výchozí stav interface je vypnutý
```

Ještě je vhodné nastavit ***default gateway***, pokud chceme komunikovat třeba do internetu, aby router věděl, kam poslat provoz, který nepatří do žádné z jeho VLAN. Možností nastavení je několik, ale první možnost je pouze pro úplnost, tu nemůžeme použít!

```
SWITCH(config)#ip default-gateway 10.0.1.250        // mohu použít, pouze pokud není zapnuto routování
SWITCH(config)#ip default-network 10.0.1.0          // nastaví defaultní síť, při používání routování
SWITCH(config)#ip route 0.0.0.0 0.0.0.0 10.0.1.250  // nejčastější metoda, vytvořím přímo záznam do routovací tabulky
```

Pouze jako poznámku zde zmíním užitečnou věc. Pokud jsme rozdělili původní jednu síť na VLANy a využívali jsme jeden DHCP server, tak můžeme stále pokračovat v používání jednoho serveru. Na serveru vytvoříme několik poolů (pro každý subnet jeden). A na switchi nakonfigurujeme pro dané VLANy ***DHCP Relay Agenta***, ten předává broadcast požadavky na přidělení IP adresy pomocí unicastu na zadaný server.

```
SWITCH(config-if)#ip helper-address 10.0.1.10
```

## Omezení routování a neroutované VLANy

Myslím, že mohou nastat dvě situace, pokud nechceme vše routovat, a to

- **neroutovaná VLAN** - chceme mít izolovanou VLAN, tady naprosto neroutovanou s ostatními VLANami, buď funguje jako uzavřená síť nebo ji dále propojujeme pomocí firewallu či jiné GW
- **omezeně routovaná VLAN** - chceme, aby jedna VLAN mohla komunikovat pouze s některými dalšími, případně aby byla povolena pouze určitá komunikace

### Neroutovaná VLAN

Z předchozí kapitoly je jasné, jak jednoduše vytvoříme izolovanou VLANu. Stačí, aby její ***VLAN interface neměl zadanou IP adresu*** (nemusí vůbec existovat či může být shutdown). Taková VLAN se neúčastní routovacího procesu. IP adresu potřebujeme mít na VLANě z několika důvodů, například když chceme v této VLANě přistupovat na switch (na jeho CLI nebo webové rozhranní) nebo když chceme využít přidělování IP adres z DHCP serveru na switchi (do této VLAN). Ale po promyšlení se asi vždy objedeme bez toho, aby VLAN, kterou nechceme routovat, měla adresu.

Druhá možnost je trošku složitější a znamená vytvoření ACL. O tom pojednává další kapitola.

### Omezeně routovaná VLAN pomocí ACL

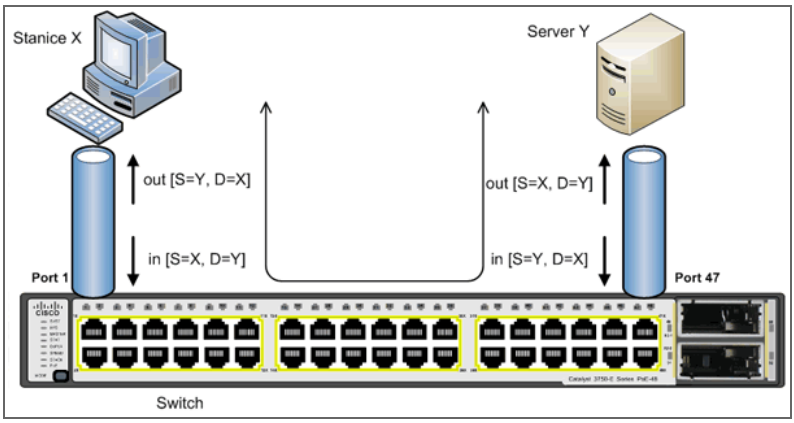
Pokud chceme, aby některá VLANa byla routována (mohla komunikovat) pouze s některými dalšími (a ne se všemi). Případně chceme ještě více specifikovat komunikaci mezi VLANami (nebo ji úplně zakázat). Tak k tomu můžeme použít **Access Control List (ACL)**. A tyto ACL aplikujeme na ***VLAN interface*** na routeru (v našem případě core switchi), těmto ACL se říká ***Router ACL***.

> ***Pozn.:*** *Další možností je použití* ***VLAN ACL*** *(VACL), jinak řečeno* ***VLAN map*** *(vlan access-map), ty se uplatňují nejen na routovaný provoz (jako Router ACL), ale i na bridgovaný (switchovaný) provoz uvnitř VLANy, tedy na všechny pakety. Mají určité výhody, ale také nevýhody. V tomto článku se jim nevěnuji.*

Podrobný (doufám) popis ACL je v článku **Cisco IOS 8 - Access Control List**, takže se zde nebudu věnovat vytváření ACL. Ale zkusím podrobně vysvětlit ***aplikaci ACL na interface***, protože jsem to delší dobu nechápal. Pokud uvažujeme ***port ACL***, které se aplikuje na nějaký port, tak je situace relativně jednoduchá. V článku o ACL je schematický obrázek, který snad jasně ukazuje, že se na komunikaci díváme z pohledu routeru, a pokud se chceme věnovat přicházející komunikaci do portu (do routeru), tak se aplikuje **in** a pokud odcházející, tak **out**.

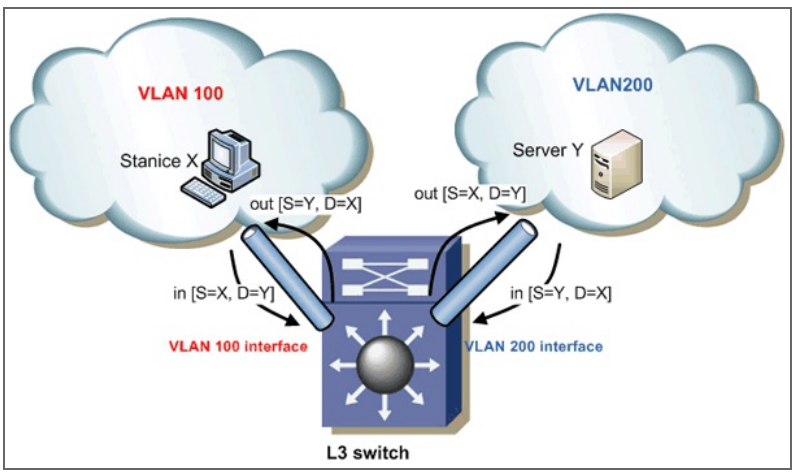
### Aplikace ACL na port

Následující obrázek obdobně ilustruje situaci. Počítač X je připojen do portu 1. Server Y je připojen do portu 47 switche. Pokud aplikujeme ACL na port 1 jako vstupní, tak se zabýváme komunikací, která odchází ze stanice X a směřuje (nemusí jenom na něj) na server Y. V tom případě zdrojové adresy jsou stanice X (S=X) a cílové adresy jsou serveru Y (D=Y). Další varianty jsou znázorněné v obrázku.




### Aplikace ACL na VLAN

Pokud situaci přeneseme na náš L3 core switch, tak je situace stejná, ale myslím, že vyžaduje vážnější zamyšlení. My sice ACL aplikujeme na VLAN interface, ale nedíváme se na směry z pohledu VLAN, ale z pohledu switche (nebo jeho routovacího procesu - VLAN interfacu). Jakoby VLANa byla mimo switch a vše z ní vstupovalo do routeru přes VLAN interface, na který aplikujeme ACL. Když aplikujeme ACL jako vstupní na VLANě, tak řeší komunikaci, která z této VLAN odchází a vstupuje do routovacího procesu (do interfacu). Situaci jsem se pokusil schematicky zobrazit na následujícím obrázku.




### Volba ACL a směru

Cisco udává jediné obecné doporučení:

- **standard ACL** aplikovat blízko ***cíle*** na jako ***out*** (pravidla obsahují pouze zdrojové adresy, takže omezujeme, co do cíle vstupuje)
- **extended ACL** aplikovat blízko ***zdroje*** jako ***in*** (pravidla obsahují zdrojové i cílové adresy, omezujeme provoz, který odchází ze zdroje, tedy ještě dříve než je zpracován routovacím procesem)

V praxi máme možnost ***aplikovat ACL*** u ***zdroje*** nebo u ***cíle*** a aplikovat buď směr ***in***, směr ***out*** nebo oba ***in*** i ***out***. V případě, že řídíme provoz na úrovni portů, tak si musíme také uvědomit, na kterou stranu komunikace (zdroj, cíl) nastavujeme port.

**U interVLAN routingu** nám někdy může stačit ***standard ACL***, ale asi častěji použijeme ***extended ACL***. Zda jej nastavíme ***in***, ***out*** nebo ***oboje*** dohromady je třeba důkladně zvážit. Valná část komunikace (protokol TCP) potřebuje navázat spojení, takže když omezíme pouze jeden směr, spojení se nenaváže, a komunikace neprobíhá. Ale určitá komunikace probíhá jednosměrně (klasicky UDP protokol).

Pokud aplikujeme ACL jako vstupní u zdrojové VLANy, tak jej stačí nastavit pouze na jednu VLANu (tu zdrojovou a omezujeme, s kým může komunikovat). Pokud bychom jej chtěli nastavit u cíle, tak jej musíme nastavit u všech cílů, do kterých nesmí zdrojová VLANa komunikovat.

Malé shrnutí, jak můžeme ACL aplikovat:

- na **in** u omezované VLANy - zdroj je omezovaná VLAN, cíl buď povolujeme, nebo zakazujeme, řídíme, co může z VLANy odcházet
- na **out** u omezované VLANy - cíl je omezovaná VLAN, řídíme, co do ní může vstoupit
- na **in** i **out** u omezované VLANy - cíl a zdroj v obou variantách, zamezíme vstupu i výstupu dat do dané VLANy

- na **out** u cílové VLANy - v některých případech můžeme potřebovat aplikovat ACL i u cílových VLAN, například chceme, aby všechny VLANy mohli komunikovat pouze s jednou (třeba se servery) a ne mezi sebou

## Příklad použití

Pokusím se o komplexnější příklad. Vezmeme situaci z předchozího obrázku, kde máme VLAN 100, VLAN 200 a přidáme ještě VLAN 300. Chceme, aby VLAN 100 komunikovala s VLAN 200 a VLAN 300, ale VLAN 300 komunikovala pouze s VLAN 100, stejně jako VLAN 200. Jinak řečeno plné routování, kde se VLAN 300 omezí pouze na komunikaci s VLAN 100.

```
SWITCH(config)#ip routing
SWITCH(config)#interface vlan 100
SWITCH(config-if)#ip address 10.0.1.1 255.255.255.0
SWITCH(config-if)#no shutdown
SWITCH(config)#interface vlan 200
SWITCH(config-if)#ip address 10.0.2.1 255.255.255.0
SWITCH(config-if)#no shutdown
SWITCH(config)#interface vlan 300
SWITCH(config-if)#ip address 10.0.3.1 255.255.255.0
SWITCH(config-if)#no shutdown
SWITCH(config)#ip access-list extended vlan300in
SWITCH(config-ext-nacl)#permit ip 10.0.3.0 0.0.0.255 10.0.1.0 0.0.0.255
SWITCH(config)#interface vlan 300
SWITCH(config-if)#ip access-group vlan300in in
```

Druhý příklad ukazuje možnost izolace jedné VLANy.

```
SWITCH(config)#ip access-list extended vlan300
SWITCH(config-ext-nacl)#deny ip any any
SWITCH(config)#interface vlan 300
SWITCH(config-if)#ip access-group vlan300 in
SWITCH(config-if)#ip access-group vlan300 out
```

# Cisco Routing 1 - obecné vlastnosti směrovacích protokolů

Pátek, 20.03.2009 14:43 | Samuraj - Petr Bouška | sítě

Tento seriál se věnuje routingu, tedy směrování v počítačových sítích. Jednotlivé popisy jsou obecně platné, ale zde jsou uváděny se zaměřením na produkty firmy Cisco a s popisem konfigurace v Cisco IOSu. Tento první díl probírá základní rozdělení routovacích protokolů a popis kategorií. Popisuje základní termíny a ukazuje obecné konfigurační příkazy IOSu. Dále se věnuje Policy Based Routingu, sumarizaci adres, filtrování a redistribuci cest.



*Celý tento seriál o routingu vznikl, když jsem se připravoval na Cisco test **642-901 BSCI**, jako moje poznámky. Následně jsem provedl pouze jednoduchou úpravu a text publikoval. Pokud myslíte, že něco podstatného chybí, něco není správně popsáno nebo není úplně jasné, tak uvítám vaše informace v komentářích.*

*Základy routování jsou popsány v článku **TCP/IP - Routing - směrování**.*

## Dělení routovacích protokolů

Do routovací tabulky se vytváří několik typů záznamů cest (routes), záleží na tom, jakým způsobem vznikly. Pakety jsou podle toho směrovány jedním ze základních způsobů routování:

- **statické routování** - ručně zadané cesty (záznamy v routovací tabulce), bezpečné a dobré, ale nereflektuje změny v topologii sítě
- **dynamické routování** - síť se automaticky přizpůsobuje změnám v topologii a dopravě, automaticky se vypočítávají cesty pomocí routovacího protokolu
- **defaultní routování** - pokud neexistuje jiná cesta, tak se použije defaultní

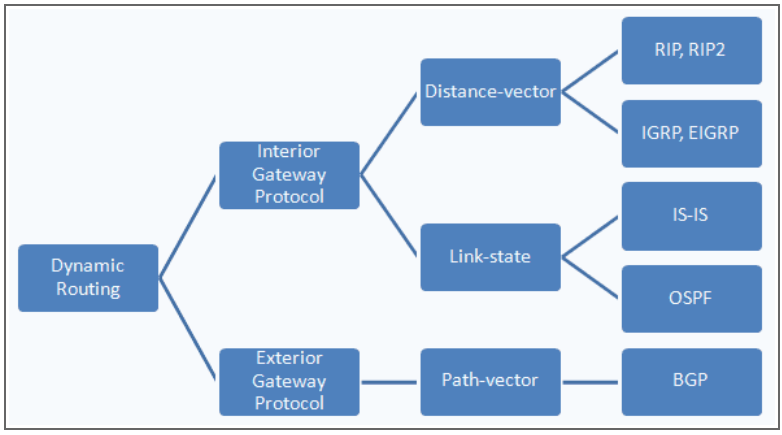
Dynamické routovací protokoly jsou dvou základních typů:

- **Distance-Vector Routing Protocol**
- **Link-State Routing Protocol**

*Pozn.: Také je zde **Path-Vector Routing Protocol**, který patří pod **Distance-Vector Routing Protocol**.*

Dále dělíme dynamické protokoly podle toho, zda jsou určeny pro nasazení uvnitř lokální sítě (přesněji řečeno uvnitř autonomního systému (AS), který může obsahovat několik LAN) nebo fungují napříč sítěmi (spojují AS dohromady).

- **Interior Gateway Protocol - IGP** - routuje uvnitř Autonomous System (AS)
- **Exterior Gateway Protocol - EGP** - routuje mezi AS




## Distance-Vector Routing Protocol

- RIP, RIP2, IGRP, EIGRP, BGP
- routery udržují **routovací tabulku** s informací o (vektoru) vzdálenosti do dané sítě
- periodicky **routovací tabulku** zasílají sousedům, ti si upraví svoji tabulku a tu opět odešlou dál
- pro výpočet nejlepší cesty se používá jedna (počet hopů u RIP) nebo více (propustnost linky a zpoždění u IGRP) metrik
- upraveným typem distance-vector protokolu je **path-vector protocol**.
- jednoduché DVRP (RIP, IGRP) nevytváří vztahy se sousedy
- problémem jsou **routovací smyčky** (routing loops) - řeší se pomocí definice maximální vzdálenosti (TTL), Split Horizon (neposílá routu na rozhranní přes které přišla), Route Poisoning, hold-down timer (čekací interval, než je síť stabilní, prodlužuje konvergenci)

## Link-State Routing Protocol

- OSPF, IS-IS
- routery udržují komplexní databázi **síťové topologie** (vytvořenou pomocí LSA)
- vyměňují si **Link State Advertisements - LSA**, LSA jsou vyvolány nějakou událostí v síti, také **Link State Packet - LSP** nebo Link State PDU

- do svého okolí také odesílá ***Hello pakety***, kde zasílá informace o sobě
- rychle reaguje na změny ***topologie***, ale spotřebovává více pásma (hlavně na počátku zasílá množství LSP) a zdrojů na routeru
- metrika je komplexní, nejlepší cesta se počítá pomocí Dijkstrova algoritmu ***shortest path first - SPF***
- pro zlepšení vlastností se rozděluje na menší oblasti, hraniční routery posílají sumární cesty, využívá multicast, číslování LSA

## Obecné termíny

### Autonomous System - AS

- je skupina IP sítí a routerů, které jsou pod stejnou technickou administrací
- čísla 1 až 65535
- privátní rozsah 65512 - 65535
- AS používá EGP pro komunikaci s jinou AS
- uvnitř AS se routy naučené z BGP mohou redistribuovat do IGP

### Administrative Distance - AD

- určuje důvěryhodnost protokolu - definuje spolehlivost protokolu a prioritizuje lepší nižším číslem
- je vlastnost používaná na routrech k určení nejlepší cesty mezi více routovacími protokoly
- Jinak řečeno na routeru může běžet více routovacích protokolů a podle AD se rozhoduje, který se použije.
- Na Cisco routrech můžeme měnit defaultní hodnoty.

**N**astavení ***administrativní vzdálenost*** AD (nižší lepší) provede následující příkaz. `Weight` je hodnota AD.

```
ROUTER(config-router)#distance weight [source-ip-address source-mask (access-list-number | name)]
```

| Route Source | Default Distance Values |
|---|---|
| Connected interface | 0 |
| Static route | 1 |
| EIGRP summary route | 5 |
| External BGP | 20 |
| EIGRP | 90 |
| IGRP | 100 |
| OSPF | 110 |
| IS-IS | 115 |
| RIP | 120 |
| EGP | 140 |
| On Demand Routing (ODR) | 160 |
| External EIGRP | 170 |
| Internal BGP | 200 |
| Unknown | 255 |

### Konvergence

- procesy a čas potřebný pro konverzi směrovacího protokolu
- konvergence je dosažena ve chvíli, kdy všechny routry mají kompletní aktuální informaci o topologii

### Síťové informace

- serial interface má default bandwidth rovnu `T1 = 1.544Mbps`
- point to multipoint Frame Relay má `default rychlost = propustnost_kanálu (T1) / počet_subinterfaců`
- ***Protocol Data Unit*** - **PDU** - záleží na protokolu / vrstvě OSI modelu, takže třeba L2 PDU je rámec, L3 PDU je paket

## Obecná konfigurace

### Statické routování

```
SWITCH(config)#ip route [destination_network] [mask] [next_hop or exit_interface] [administrative_distance] [permanent]
SWITCH(config)#ip route 192.168.50.0 255.255.255.0 192.168.1.1
```

### Defaultní routování - Default gateway

```
SWITCH(config)#ip default-gateway 10.0.1.250       // definice GW, mohu použít, pouze pokud není zapnuto routování
SWITCH(config)#ip default-network 10.0.1.0         // nastaví defaultní síť, propaguje ji pomocí routovacího protokolu
SWITCH(config)#ip route 0.0.0.0 0.0.0.0 10.0.1.250 // vytvořím záznam do routovací tabulky pro GW
```

### Dynamické routování

```
SWITCH(config)#ip routing                     // na L3 switchi zapne routing
SWITCH(config)#router [protokol] [keyword]    // obecně volba routovacího protokolu
SWITCH(config-router)#network [network-number] // definice sítě, která se propaguje, u classles se používá wildcard mask
```

### Nastavení loopback interface

**L**oopback se využívá pro řadu účelů, jeho hlavní výhoda je, že je stále ***up*** (nikdy nejde down). Takže tam, kde se například router ID počítá z IP adresy routeru se nastaví loopback (jinak, když by šel interface down, tak by se vše přepočítávalo).

```
SWITCH(config)#interface loopback 0
SWITCH(config-if)#ip address 215.10.7.1 255.255.255.255
```

### Informace - show příkazy

```
ROUTER#show ip protocols  // které sítě jsou routovány kterým protokolem a parametry (časovače, filtry, metrika, sítě.)
ROUTER#show ip route      // zobrazí routovací tabulku
ROUTER#show ip interface  // zobrazí informace o interfacech z pohledu IP
```

## Policy Based Routing - PBR - route-map

Obdoba **ACL** nebo lépe **class-map**, prochází se číslovaný seznam pravidel. Pravidla obsahují část dvě části:

- `match` - na co se uplatní, můžeme použít ***ip address*** podle ACL nebo prefix-list, ***length*** min, max délka paketu, ***tag***, ***route type***, pokud nezadáme část match, tak se aplikuje na všechny pakety
- `set` - co se provede, můžeme použít nastavení ***precedence***, ***tos***, ***ip next-hop*** - další router kam poslat, ***interface*** - přes který interface routovat, ***metric*** - změníme metriku

```
ROUTER(config)#route-map map-tag [permit | deny ] <Sequence Number>

ROUTER(config)#route-map ospf-to-eigrp deny 10
ROUTER(config-map-router)#match tag 6
ROUTER(config-map-router)#match route type external type-2  // musí platit toto, i předchozí
ROUTER(config)#route-map ospf-to-eigrp permit 20
ROUTER(config-map-router)#match ip address prefix-list pfx  // IP adresu nalezne pomocí prefix-list, jinak možno ACL
ROUTER(config-map-router)#set metric 40000 1000 255 1 1500
ROUTER(config-map-router)#set ip next-hop 142.5.9.1         // na který router se odešle
```

Následně aplikujeme `route-mapu` na interface. Můžeme aplikovat pouze jednu `route-map` per interface, ale `route-map` se může skládat z více číslovaných částí.

```
ROUTER(config-if)#ip policy route-map ospf-to-eigrp
```

## Agregace - sumarizace

Často nepotřebujeme znát jednotlivé routy (hlavně v případě propojení více sítí/protokolů), ale stačí znát jak se dostat do hlavní sítě. Takže můžeme skupiny podsítí dostupných přes jeden router nahradit jednou sumarizovanou sítí. Některé protokoly podporují automatickou sumarizaci přes classful hranice sítí (jako EIGRP).

Když se vytvoří sumární routa (v EIGRP nebo BGP), tak se vytvoří ***Null0 interface*** (nebo je třeba jej vytvořit), kam směruje tato sumarizovaná routa. Při vlastním routování se použije přesnější adresa (z které se prováděla sumarizace), takže musí existovat záznamy pro jednotlivé podsítě, jinak by data směrovala do Null0. V odesílaných routovacích updatech pak může figurovat pouze sumární routa.

Pokud se u EIGRP smaže poslední specifická routa, která byla použita uvnitř sumarizované, tak se smaže i summary route.

```
ROUTER(config)#ip route 198.10.0.0 255.255.0.0 null0        // pouze kvůli sumarizaci
ROUTER(config-router)#redistribute static                   // jenda možnost sumarizace, ručně zadáme routy, spolu s Null0
ROUTER(config-router)#network 198.10.0.0 mask 255.255.255.0 // další možnost agregace, také dohromady s Null0
ROUTER(config-router)#no auto-summary                       // další možnost by byla autosumarizace na classful (EIGRP), ale
ROUTER(config-if)#ip summary-address eigrp 100 192.1.0.0 255.255.0.0        // manuální definice sumarizace v EIGRP, 100 = č
ROUTER(config-router)#aggregate-address 200.52.0.0 255.255.0.0 summary-only  // v BGP, vloží do routovací tabulky sumární rou
```

## Route Filtering - filtrace cest v routovacích updatech

Často chceme řídit, jaké routy se mají nacházet v přijímaných (při učení od souseda se odfiltrují určité záznamy) a odesílaných (některé záznamy se ani neposílají sousedovi) ***updatech k sousedům***. Je to z důvodu základní bezpečnosti, skrytí určitých rout od zbytku organizace, kontroly nadbytečného zatížení provozu, zabránění routovacím smyčkám.  Pro řízení máme tři základní metody:

- **passive interface**
- **distribute list**
- **prefix list**

### Passive-interface

Nejjednodušší metoda filtrace, která ale nejde příliš do detailů. ***Pasivní interface*** neodesílá routovací updaty na daném interfacu, takže se jeho sousedé nedozví o routách. Pro ***RIP, RIP2*** a ***IGRP*** se neodesílají updaty sousedům na daném interfacu, ale stále se přijímají a používají updaty od sousedů. Pro ***EIGRP*** a ***OSPF*** se přestanou odesílat *Hello pakety*, takže se zruší sousedství, tím pádem nejen že neodesílá updaty, ale také je nepřijímá.

```
ROUTER(config-router)#passive-interface Serial0/0 // daný interface neposílá updaty
ROUTER(config-router)#passive-interface default   // každý interface, kde není určeno jinak (no passive-interface s0/0), je p
```

### Distribute-list

Využívá **standard IP ACL** (případně i extended), pomocí kterého definujeme, které routy nechceme posílat (`deny`) a které chceme (`permit`).  Například, když chceme z updatů vyřadit síť `192.168.1.0/24`, kterou máme v routovací tabulce, můžeme použít ACL.

```
ROUTER(config)#access-list 10 deny 192.168.1.0 0.0.0.255
ROUTER(config)#access-list 10 permit any
```

***Distribute list*** aplikujeme v rámci routovacího procesu a určujeme, zda se použije na odchozí (`out`) nebo příchozí (`in`) updaty.

```
ROUTER(config-router)#distribute-list 10 out   // filtrace odchozích updatů, 10 je číslo ACL
```

Na jaké updaty se bude omezení aplikovat, můžeme ještě více specifikovat. U příchozích i odchozích můžeme specifikovat ***interface***, kterého se aplikace týká. U odchozích updatů můžeme navíc zadat ***routing process*** nebo ***číslo AS***. To se týká redistribuce a pak se aplikuje pouze na updaty ze zadaného routovacího procesu nebo AS

```
ROUTER(config-router)#distribute-list 10 in ethernet 0 // updaty, které přichází přes interface e0
ROUTER(config-router)#distribute-list 10 out igrp 5    // odchozí updaty, které jsou redistribuovány z IGRP 5
ROUTER(config-router)#distribute-list 10 out 100       // odchozí updaty, které jsou redistribuovány z AS 100
```

U **BGP** můžeme použít distribute list na souseda.

```
ROUTER(config-router)#neighbor 10.0.10.1 distribute-list 10 out
```

> ***Pozn.:*** *Pro routovací proces můžeme specifikovat celou řadu distribute list, ale na jeden interface a směr může být pouze jeden.*

**Prefix-list**

***Prefix list*** sám o sobě není filtrační metodou. Jedná se pouze o prostředek pro filtraci. Vlastní prefix-list je něco jako ACL, ale pracuje trochu jiným způsobem. Slouží k blokování určitých sítí v updatech od/k sousedům. Je možno jej použít pro BGP s nastavením souseda nebo místo ACL v ***distribute listu***. Podobně jako ACL, je ***prefix list*** pojmenovaný číslovaný seznam pravidel, která jsou buď ***permit*** nebo ***deny***. Defaultně začíná číslem 5 a po 5 se zvyšuje. Oproti ACL filtruje pomocí prefixů, a to buď přesné délky prefixu nebo rozsahu. Do pravidel se zadává IP adresa a bitově maska (CIDR).

> ***Pozn.:*** *Prefix list můžeme využít nejen při filtraci, ale také pro* ***Policy Based Routing*** *na výběr adresy.*

```
ROUTER(config)#ip prefix-list test deny 0.0.0.0/0        // zakáže defaultní routu 0.0.0.0
ROUTER(config)#ip prefix-list abc permit 192.0.0.0/8 ge 8 le 24 // bere všechny prefixy >= 8 a <= 24
ROUTER(config-router)#neighbor 170.10.20.1 prefix-list test in  // aplikace filtrování na soused
ROUTER(config-router)#distribute-list prefix abc in             // jiná možnost aplikace je pomocí distribute listu
ROUTER#show ip prefix-list                                      // zobrazí prefix listy
```

## Propojení routovacích protokolů - Route Redistribution

Aby se dalo efektivně podporovat více routovacích protokolů v jedné internetwork, musí se sdílet informace mezi protokoly. Tomu se říká ***route redistribution***. Redistribuce může být ***jednosměrná*** nebo ***obousměrná***. Routery, které provádí redistribuci, se nazývají ***boundary*** (hraniční).

***Redistribuce*** je, když routovací protokol zveřejňuje routy, které se naučil jiným způsobem, to jsou ***statické routy***, ***přímo připojené interfacy*** nebo jiný ***routovací protokol***. Různé routovací protokoly mají různé charakteristiky (jako metriku, podporu classles).

Doporučené řešení pro redistribuci mezi dvěma protokoly, kde existuje více cest (kdy si navzájem vyměňují routy), je kvůli smyčkám použit:

- ***jednosměrnou redistribuci*** (na druhou stranu default route)
- ***route-mapy*** či ***distribution-list***, aby se zabránilo přenosu informací, které pochází z daného procesu zpět do něj
- ***statické routy***
- ručně nastavit ***Administrative Distance***

Když konfiguruji redistribuci, tak konfiguruji ten protokol, který chci, aby zveřejňoval routy z jiného zdroje. Používá se příkaz **redistribute**. Při redistribuci musíme nastavit metriku (nastavení záleží na protokolu) nebo se použije defaultní. ***OSPF*** má defaultní metriku pro ostatní protokoly **20** mimo ***BGP***, pro které je **1**. ***RIP***, ***IGRP***, ***EIGRP***, ***IS-IS*** mají defaultní metriku **0** (nekonečno, unreachable, not advertise).

Při redistribuci z ***classful protokolu*** (IGRP, RIP) ***do classless*** (OSPF) se nedistribují subneted routes. Při redistribuci do OSPF je potřeba použít klíčové slovo **subnets**, aby se přenesly subnetované routy. Defaultně se routy redistribují do OSPF jako ***External Type2*** (E2).

IGP protokoly (jako RIP, OSPF) redistribujeme do BGP, aby se dostaly do dalších EBGP. Ale musíme být opatrní, protože hrozí nebezpečí smyček, protože redistribuované routy nemusí pocházet z daného AS, ale mohli sem dorazit pomocí BGP. Pro redistribuci IGP route do BGP se doporučuje použít příkaz `aggregate-address`.

```
ROUTER(config)#router rip
ROUTER(config-router)#redistribute static metric 1  // redistribuce statických route
ROUTER(config-router)#redistribute ospf 1 metric 1  // redistribuce OSPF do RIP s nastavením metriky
ROUTER(config-router)#default-metric  1   // nastavení defaultní metriky, které se použije, tam kde ji nezadám
ROUTER(config)#router ospf 1
ROUTER(config-router)#redistribute static metric 200 subnets
ROUTER(config-router)#redistribute eigrp 1 metric 20 metric-type 1 subnet
ROUTER(config)#router eigrp 1
ROUTER(config-router)#redistribute rip metric 10000 100 255 1 1500
ROUTER(config-router)#redistribute ospf 1 match external 1 external 2  // redistribuujeme pouze typ E1 a E2
ROUTER(config-router)#redistribute isis level-1-2 metric 100 100 200 1 1500 // redistribuujeme ISIS L1 i L2
ROUTER(config-router)#redistribute ospf 1 route-map ospf-to-eigrp      // použití route-mapy pro filtrování, co se přenáší
```

Pro troubleshooting se mohou použít dva příkazy.

```
ROUTER#show ip route    // na routeru uvnitř i na ASBR, zda zde jsou redistribuované routy
ROUTER#show ip protocol // na ASBR, ověří konfiguraci redistribuovaných route, zobrazuje i filtry
```



Autor: Petr Bouška



Související články:

# Cisco Routing 6 - srovnání routovacích protokolů

Úterý, 28.04.2009 16:27 | Samuraj - Petr Bouška | sítě

V předchozích dílech seriálu o routování (hlavně z pohledu Cisco prvků) jsem stručně sepsal vlastnosti hlavních směrovacích protokolů. V tomto posledním díle se pokouším o stručné tabulkové porovnání hlavních vlastností. Není to určeno k rozhodování při výběru vhodného routovacího protokolu, ale spíše pro přehled potřebný na Cisco test.



*Srovnání není úplné a rád uvítám, když doplníte chybějící údaje (případně opravíte nepřesnosti).*

| | RIP \| RIP2 | EIGRP | OSPF | IS-IS | BGP |
|---|---|---|---|---|---|
| typ | distance vector | distance vector | link state | link state | path vector |
| class | classful \| classless (CIDR, VLSM) | classless (CIDR, VLSM) | classless (CIDR, VLSM) | classless (CIDR, VLSM) | classless (CIDR, VLSM) |
| metrika | počet hopů (max 15) | delay, bandwidth (reliability, load, MTU) | cost (100Mb/BW) | cost (default 10) (standardně také delay, expense, error, Cisco je nepodporuje) | atributy (weight, local_pref, originate, AS_path, MED .) |
| route algoritmus | Bellman-Ford | DUAL | SPF (Dijkstra) | SPF (Dijkstra), PRC (Partial Route Calculation) | Best Path |
| route update | 30s celá tabulka bez zpětných cest (split horizont) | multicast update (224.0.0.10) | 30m nebo při změně LSA (224.0.0.5, 6) | LSP, CSNP, PSNP | update |
| hello pakety | nemá | 5s/60s(pomalé)/multicast | 10s / multicast, unicast | 10s / multicast / 5x hello | 60s keepalive |
| komunikace | UDP 520 | IP 88 | IP 89 | IS-IS (L3) | TCP 179 |
| tabulky / způsob uložení směrovacích informací | routovací | routovací, topologie, sousedé | routovací, topologie, sousedé, link-state | LSA databáze | atributy, topologie |
| úrovně | - | AS | area, AS | area, domain | AS |
| agregace | RIP2 manual | auto, manual | manual (pouze na hranici oblasti) | manual (pouze na hranici oblasti) | auto, manual |
| IPv6 | RIPng | IPv6 EIGRP | OSPFv3 | ano | mBGP |
| sítě | | BMA, NBMA, Pt-Mpt | Pt-Pt, BC, NBMA, Pt-Mpt, virtual link | | |