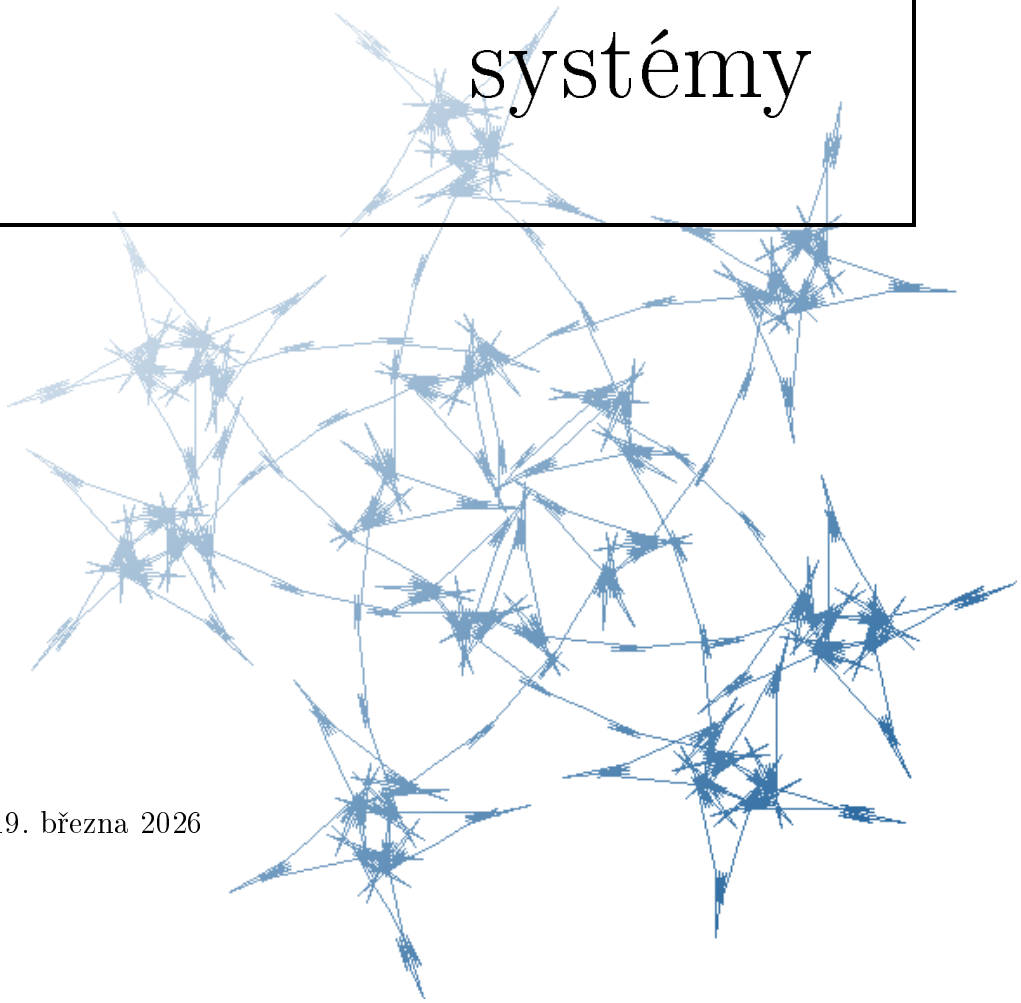
SLEZSKÁ
UNIVERZITA
FILOZOFICKO-
PŘÍRODOVĚDECKÁ
FAKULTA V OPAVĚ
ÚSTAV INFORMATIKY

Šárka Vavrečková

Cvičení do předmětu

Počítačové sítě a decentralizované systémy



Opava
Poslední aktualizace: 19. března 2026

Anotace: Tato skripta jsou určena pro studenty předmětu *Počítačové sítě a decentralizované systémy* pro navazující magisterské studium na Ústavu informatiky Slezské univerzity v Opavě, a to pro cvičení z tohoto předmětu.

V předmětu se zabýváme prostředky a postupy používanými v počítačových sítích, navazujeme na obdobný předmět z bakalářského stupně studia. Na cvičeních se budeme učit pokročilejší konfiguraci switche a routeru, VLAN, STP, CDP, ACL, DHCP, NAT, tunely, konfiguraci různých směrovacích protokolů (především OSPF a BGP).

Toto je novější verze skript, ale bohužel zatím není úplná. Do doby, než bude dokončen tento dokument, bude k dispozici i starší verze skript (se všemi kapitolami):

https://vavreckova.zam.slu.cz/obsahy/pocsit/skripta/sitemgr2025.pdf

Počítačové sítě a decentralizované systémy

Cvičení

RNDr. Šárka Vavrečková, Ph.D.

Dostupné na: http://vavreckova.zam.slu.cz/sitedec.html

Ústav informatiky
Filozoficko-přírodovědecká fakulta v Opavě
Slezská univerzita v Opavě
Bezručovo nám. 13, Opava

Sázeno v systému LaTeX

Předmluva

Co najdeme v těchto skriptech

Tato skripta jsou určena pro studenty Ústavu informatiky Slezské univerzity v Opavě. Obsahují látku vyučovanou na cvičeních předmětu *Počítačové sítě a decentralizované systémy*. Navazujeme na obdobný předmět v bakalářském stupni studia a předpokládá se, že čtenář už má určité znalosti z oblasti počítačových sítí.

Protože se jedná o cvičení, budeme se zabývat hlavně praktickými aspekty počítačových sítí: trochu hardwarem, ale hlavně tím, jak se co konfiguruje. Během semestru se postupně zaměříme na pokročilou konfiguraci switche, routeru, a pak úlohy související s bezpečností a správou síťových zařízení.

Některé oblasti jsou „navíc“ (jsou označeny ikonami fialové barvy), ty nejsou probírány a ani se neobjeví na zkoušce – jejich úkolem je motivovat k dalšímu samostatnému studiu či pokusům nebo pomáhat v budoucnu při získávání dalších informací. Pokud je fialová ikona před názvem kapitoly (sekce), platí pro vše, co se v dané kapitole či sekci nachází.

Značení

Ve skriptech se používají následující barevné ikony:

- *Rychlý náhled* (skript, kapitoly), ve kterém se dozvíme, o čem to bude.
- *Klíčová slova* kapitoly.
- *Cíle studia* pro kapitolu nám řeknou, co nového se v dané kapitole naučíme.
- Nové *pojmy*, značení apod. jsou značeny modrým symbolem, který vidíme zde vlevo.
- Konkrétní *postupy* a nástroje, způsoby řešení různých situací, do kterých se může správce počítačového vybavení dostat, atd. jsou značeny také modrou ikonou.
- Některé části textu jsou označeny fialovou ikonou, což znamená, že jde o *nepovinné úseky*, které nejsou probírány (většinou; studenti si je mohou podle zájmu vyžádat nebo sami prostudovat). Jejich účelem je dobrovolné rozšíření znalostí studentů o pokročilá témata, na která obvykle při výuce nezbývá moc času.

-  Žlutou ikonou jsou označeny odkazy, na kterých lze získat *další informace* o tématu. Nejčastěji u této ikony najdeme webové odkazy na stránky, kde se dané tématice jejich autoři věnují podrobněji.
-  Červená je ikona pro *upozornění* a poznámky.

Pokud je množství textu patřícího k určité ikoně větší, je celý blok ohraničen prostředím s ikonami na začátku i konci, například pro definování nového pojmu:

 **Definice**

V takovém prostředí definujeme pojem či vysvětlujeme sice relativně známý, ale komplexní pojem s více významy či vlastnostmi.

Podobně může vypadat prostředí pro delší postup nebo delší poznámku či více odkazů na další informace. Mohou být použita také jiná prostředí:

 **Příklad**

Takto vypadá prostředí s příkladem, obvykle nějakého postupu. Příklady jsou obvykle komentovány, aby byl jasný postup jejich řešení.

 **Úkol**

Otázky a úkoly, náměty na vyzkoušení, které se doporučuje při procvičování učiva provádět, jsou uzavřeny v tomto prostředí. Pokud je v prostředí více úkolů, jsou číslovány.

 **Poznámka**

Poznámky zabírající více řádků jsou uzavřeny v tomto prostředí. Zde upozorňujeme na různé zajímavosti nebo zdůrazňujeme důležité fakty.

 Na konci každé kapitoly najdete odstavec se souhrnem kapitoly.

Obsah

Kapitola 1

Opakování

Rychlý náhled: Na prvním cvičení si zopakujeme základní konfiguraci routeru a switche, a také si projdeme různé příkazy pro zjišťování informací o zařízeních. Podíváme se na možnosti rozšiřování funkcí zařízení pomocí různých typů zásuvných modulů. Prohlédneme si různá úložiště v síťových zařízeních, naučíme se zjišťovat fyzikální parametry zařízení, prohlédneme si ARP a MAC tabulku a projdeme si vlastnosti portů.

Klíčová slova: Router, switch, konzolový kabel, SSH, konzolový server, Packet Tracer, managed zařízení, SFP, SFP+, QSFP+, zásuvný modul, mód, ROM, flash PROM, NVRAM, filtrování výstupu, ARP tabulka, MAC tabulka, duplex, Auto-MDIX.

Cíle studia: Cílem této kapitoly je zopakovat si dosavadní znalosti a dovednosti z oblasti konfigurace počítačových sítí. Také se naučíte získávat různé typy údajů o síťových zařízeních a zorientujete se v různých typech modulů pro routery.

1.1 Aktivní síťové prvky

Následuje přehled běžných aktivních síťových zařízení. Ke každému je taktéž přidána informace o tom, na kterou vrstvu ISO/OSI patří.

Repeater (opakovač) je jednoduchý aktivní síťový prvek se dvěma porty. Příchozí signál zesílí (případně znovu vygeneruje) a pošle dál. Pracuje na vrstvě L1.

Dnes se s repeatery (příp. extendery) setkáváme spíše u bezdrátových lokálních sítí, ale také např. HDMI, kde slouží ke zvýšení dosahu signálu.

Hub (rozbočovač) je jednoduchý aktivní síťový prvek, který má typicky více než dva porty. Cokoliv přijde na některý port, jen přepošle na všechny ostatní porty (tento signál zesílí nebo znovu vygeneruje). Pracuje na vrstvě L1.

V běžných počítačových sítích se už dnes s huby moc nesetkáváme, používají se ale například v některých PAN sítích nebo u USB (ale s trochu jinými charakteristikami).

Switch (přepínač) je již inteligentnější aktivní síťový prvek, který si vede tabulku fyzických adres zařízení. Pokud v případě unicastového provozu najde v tabulce adresáta, přepne provoz na port, který má v tabulce u adresáta uvedený. Pokud adresáta nemá v tabulce nebo když jde o broadcast či multicast, provede flooding („záplavu“), tj. pošle provoz na všechny porty kromě toho, ze kterého původně přišel. Velká část funkčnosti je implementována hardwarově, což má pozitivní vliv na rychlost.

Pracuje na vrstvě L2, ale ve skutečnosti může obsahovat i funkcionalitu vyšších vrstev (multilayer switch).

Se switchi se běžně setkáváme v LAN, MAN i WAN sítích.

Bridge (most) je zařízení navzájem oddělující dva segmenty sítě, jako switch. Na rozdíl od switche mívá méně portů, jeho funkčnost je implementována softwarově a je standardizován (IEEE 802.1). V praxi se přímo s bridgi nesetkáváme. Most pracuje na vrstvě L2.

Router (směrovač) si vede směrovací tabulku, v ní však nemá adresy konkrétních uzlů, ale L3 adresy sítí a podsítí. Příchozí paket „pitvá“ o něco důkladněji než switch, přičemž podle cílové logické adresy (obvykle IP adresy) zjistí, do které sítě patří adresát, ve směrovací tabulce ověří, který port je cestou do této sítě, a na ten port paket odešle. Broadcasty zahazuje.

Routery pracují na vrstvě L3.

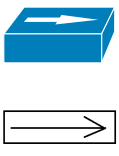
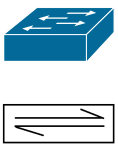
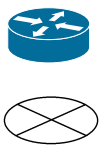
Výhodou routeru je možnost implementace pokročilých správních a bezpečnostních funkcí a schopnost oddělit komunikaci v různých sítích. Nevýhodou je výpočetně náročnější provoz, proto routery mívají méně portů než switche, výkonnější procesor a více paměti, aby vyšší výpočetní zátěž unesly.

Switch s přidanou funkcionalitou vrstvy L3 (L3 switch, multilayer switch) je takový přepínač, kterému, jak název napovídá, byla přidána funkcionalita vrstvy L3 (tedy síťové). Znamená to, že se sice pořád jedná o switch, ale má vestavěný směrovací modul (protože na vrstvě L3 se směruje mezi sítěmi) a některé porty tohoto zařízení se mohou chovat jako porty routeru. Rozlišujeme zde tedy L2 porty a L3 porty. S provozem je možné zacházet na vyšší úrovni (rozbalit PDU až na úroveň vrstvy L3, pracovat s IP adresami, směrovat apod.).

Brána (gateway) slouží k propojení dvou *různých typů* sítí, resp. slouží jako spojovací bod a zároveň „překladatel“. Například pokud máme v domácnosti VDSL router, pak toto zařízení má vestavěnou bránu pro komunikaci mezi naší lokální sítí a VDSL sítí poskytovatele internetu (v tomto případě VDSL modem).

Brána typicky pracuje na vyšší vrstvě než protokoly, které má propojit.

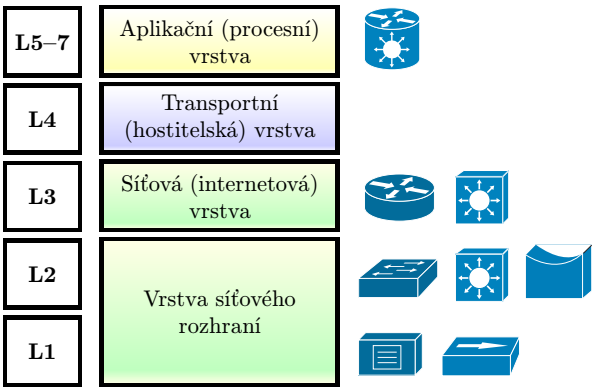
Tabulka 1.1: Zakreslení síťových zařízení rukou

Hub	Switch	Router	L3 Switch
			

Tabulka 1.1 ukazuje, jak jednoduše a rychle můžeme zakreslit tyto typy zařízení do schématu, pokud kreslíme rukou (na papír, na tabuli apod.).

Abychom si ujasnili vztah různých DCE k vrstvám ISO/OSI nebo lépe síťového modelu TCP/IP (DoD), podívejme se na obrázek vpravo. Na vrstvě L1 pracují repeatery a huby, na vrstvě L2 především switche (včetně těch s funkcionalitou vrstvy L3) a bridge, na vrstvě L3 routery a switche s funkcionalitou vrstvy L3, na aplikační vrstvě pak brány (gateway).

Routery sice mají část funkcionality vrstvy L2, ale jen takovou, kterou najdeme i u běžných koncových zařízení typu počítač, server nebo chytrý telefon. Z pohledu vrstvy L2 jde o zařízení „na konci sítě“, a vrstva L2 za tento konec nevidí. Takže router je pro vrstvu L2 také koncovým zařízením.



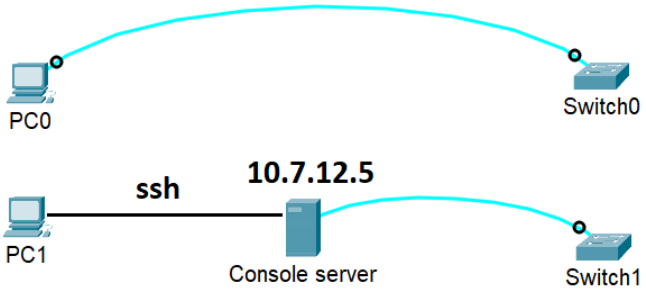
Obrázek 1.1: Umístění síťových zařízení na jednotlivé vrstvy síťového modelu

1.2 Jak se dostat ke konfiguraci zařízení

V tomto předmětu budeme používat tyto možnosti přístupu k síťovým zařízením:

- Připojení k zařízení pomocí konzolového kabelu: tuto možnost použijeme, pokud máme zařízení „v ruce“.
- Připojení k zařízení přes konzolový server: pro případ, že zařízení je umístěno v racku.
- Přístup k síťovému zařízení přes SSH.
- Packet Tracer: jde o program simulující síťová zařízení a jejich sítě.

Na obrázku 1.2 je naznačen rozdíl mezi prvními dvěma možnostmi.



Obrázek 1.2: Srovnání možností připojení k síťovému zařízení pro konfiguraci

1.2.1 Použití konzolového serveru

Předpokládejme, že máme správně nakonfigurovaný konzolový server a síťové zařízení (router, switch) je k němu připojeno konzolovým kabelem, jak je naznačeno na obrázku 1.2 dole. Pak zbývá jen navázat spojení s konzolovým serverem, což se obvykle děje přes SSH.

Potřebujeme znát IP adresu konzolového serveru, údaje pro přihlášení a jakýkoliv program, který umí navázat SSH spojení s touto IP adresou (SSH klienta). Náš konzolový server je dostupný pouze v síti přímo v učebně, přihlašovací údaje dostanete na místě.

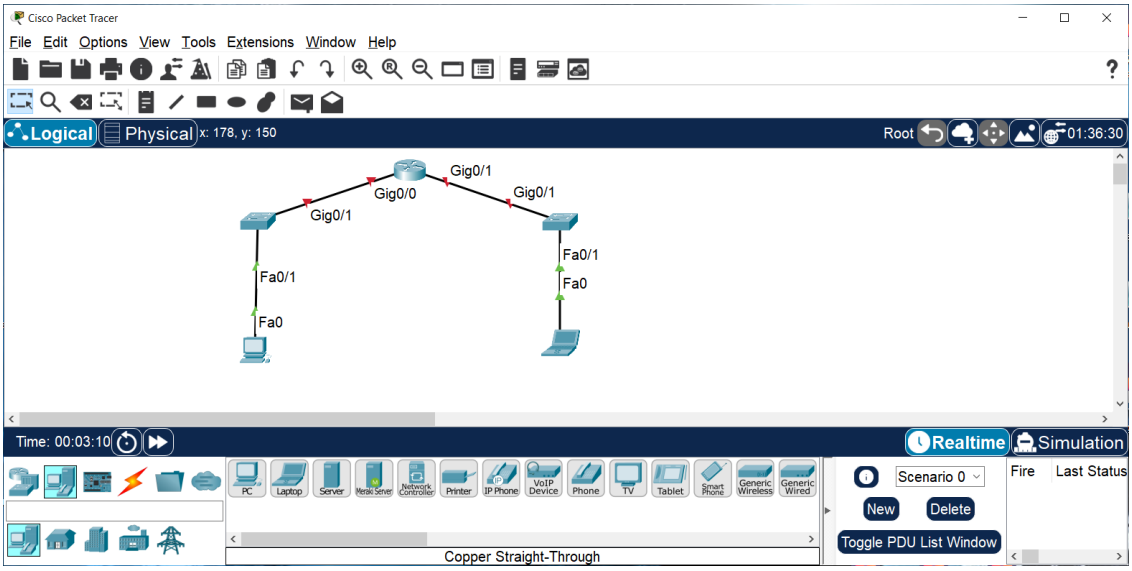
V učebně máme sice program Putty, který má vestavěného SSH klienta a tedy je poměrně univerzálním prostředníkem, ale jsou i další možnosti: například v každém operačním systému bývá dostupný příkaz `ssh`. Použijeme jednu z těchto možností:

```
ssh ip.adresa
ssh uzivatel@ip.adresa
```

V prvním případě jsme dotázáni na jméno a heslo, v druhém případě jen na heslo, protože přihlašovací jméno jsme zadali před adresou.

1.2.2 Program Packet Tracer

Program Cisco Packet Tracer je jednoduchá aplikace, ve které můžeme navrhovat a konfigurovat síť bez toho, abychom opravdu „fyzicky“ příslušná zařízení a kebely měli v ruce. Existuje pro různé platformy: Windows, Linux i MacOS.



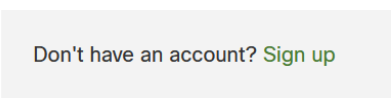
Obrázek 1.3: Prostředí aplikace Cisco Packet Tracer ve verzi 8.2.2

Novější verze jsou dostupné jen těm, kdo jsou registrovaní na webu Netacad.com (to je školicí server společnosti Cisco). Registrace je zdarma, jen bohužel zabere určitý čas a pak není úplně jednoduché najít na webu program Packet Tracer, ale při troše trpělivosti se to dá. Kdyby někdo nechtěl projít registrací, je možné stáhnout verzi 6.2, která ještě nevyžaduje přístupové údaje, ale bohužel už není aktuální se všemi riziky včetně bezpečnostních.

Pokud se rozhodnete potrpět registraci, můžete jí hned využít a poohlédnout se po různých kurzech, které jsou v systému dostupné. Mnohé z nich jsou zdarma (tzv. Self-paced), a to jak z oblasti počítačových sítí, tak i například programování.

Postup (Registrace na webu Netacad)

Příslušný web má adresu https://www.netacad.com/. Vpravo nahoře najděte tlačítko Login, pak hledejte odkaz pro vytvoření účtu:



Jako světa a kyberbezpečnosti znalí studenti určitě víte, že není dobrý nápad přihlašovat se přes účet Google či podobné…

Následuje vyplnění údajů (Cisco chce především vědět, ze které jste země, kdy jste se narodili a jakou máte e-mailovou adresu) a ověřování totožnosti (je třeba potvrdit přes e-mail, že opravdu existujete, tatáž adresa se pak používá i pro další komunikaci, tedy určitě nezadávejte falešnou adresu). Volíte si také heslo, to byste si určitě měli zapamatovat.

Až budete mít registraci za sebou, Packet Tracer si můžete stáhnout na
https://www.netacad.com/resources/lab/cisco-packet-tracer-resources

Po instalaci a spuštění se spustí průvodce, ve kterém zadáte své přístupové údaje, které jste si zvolili při registraci. Doporučuji zatrhnout, že se přihlášení má pamatovat pro příští 3 měsíce, ať tyto údaje nemusíte alespoň po tuto dobu zadávat znovu.

Postup (Starší verze Packet Traceru)

Pokud si chcete stáhnout Packet Tracer, ale nechcete se registrovat, starší verzi 6.2 bez nutnosti registrace najdete na https://www.filehorse.com/download-cisco-packet-tracer-32/27899/download/.

Jen upozorňuji, že opravdu není dobrý nápad používat starší verzi čehokoliv z bezpečnostních důvodů, a také některé funkce najdeme jen v novějších verzích, novější verze jsou také lépe vybaveny různými zařízeními (včetně Internetu věcí). Ovšem je třeba uznat, že grafické rozhraní starší verze má své kouzlo.

1.3 Síťová zařízení a jejich vlastnosti

V domácnostech a malých firmách se používají celkem jednoduchá a levná zařízení, která v sobě typicky kombinují více funkcionalit. Obvykle jde o router plus Wi-fi access point plus ethernetový switch plus VDSL modem plus ještě něco dalšího.

Ve firemní sféře se pak už setkáváme se specializovanými zařízeními, která mají mnohem rozsáhlejší možnosti konfigurace. Takové zařízení se označuje jako *managed* (česky poněkud zkomoleně „menežovatelný“), například managed switch. Konkrétní způsoby konfigurace záleží na výrobci. Často se konfiguruje v textovém rozhraní, ale může jít o „klikačku“ ve webovém rozhraní nebo speciální aplikaci.

Textové rozhraní se může zdát uživatelsky méně přívětivé, ale má výhodu lepší kontroly nad tím, co zrovna konfigurujeme. V grafickém rozhraní někdy bývá problém některé volby v složitých menu najít.

Některá síťová zařízení, zejména routery, mohou být modulární, tj. některé části můžeme podle potřeby dokupovat či vyměňovat.

U kabelů určených pro vyšší rychlosti se můžeme setkat s SFP moduly (Small Form-Factor Pluggable, novější varianty SFP+, QSFP+ a další). Tyto moduly jsou vlastně vyjímatelný transceiver (transmitter/receiver), což je část rozhraní, která je jinak uvnitř zařízení a zajišťuje funkcionalitu části vrstvy L1.

Na obrázku 1.4 vidíme různé druhy SFP modulů, konkrétně optický SFP modul (singlemode s plným duplexem, LC konektor) vnější a vnitřní strana, SFP modul pro metalický kabel RJ-45 na rychlost 1G/s (vnější strana a bok), a pak kabel s moduly kompatibilními s SFP/SFP+/SFP28.

Obrázek 1.4: Různé druhy SFP modulů. Zdroj: abctech.cz

Díky tomu, že hardwarově závislá část je „vytažená“ pryč ze zařízení, se do téže zdířky (s určitými omezeními) dá použít SFP modul pro metalický nebo optický kabel, případně i jiný. Modul může být buď přímo na kabelu, nebo může mít na vnější straně plug pro RJ-45 nebo některý optický konektor.

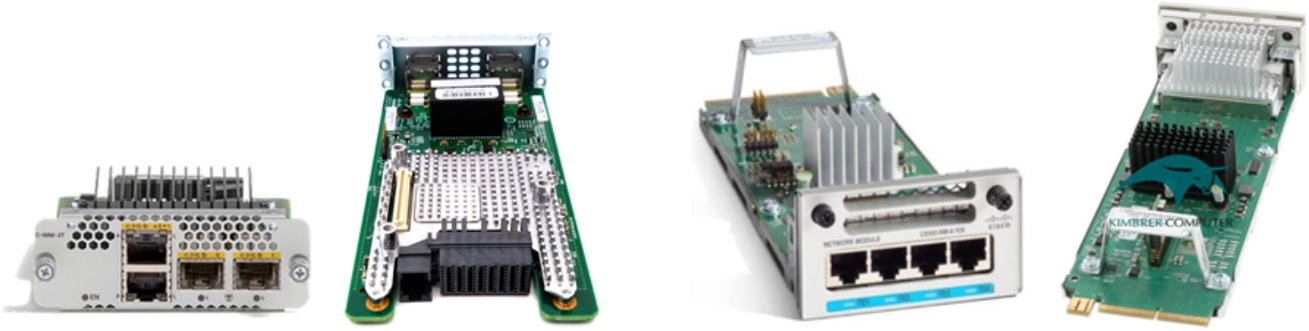
Poznámka

Různé typy modulů (podle různých standardů) se odlišují i rychlostmi, například SFP+ umožňuje komunikaci rychlostí až 10 Gb/s, SFP28 až 25 Gb/s, QSFP+ až 40 Gb/s.

Další typy modulů jsou větší a komplexnější. V případě Cisco zařízení se můžeme setkat s těmito:

- staré HWIC (High-speed WAN interface card) a eHWIC (enhanced HWIC), najdeme spíše na starších zařízeních,
- PIM (Pluggable Interface Modules),
- NIM (Network Interface Modules),
- SM (Service Modules),
- NM (Network Modules),
- napájecí moduly (přídavné či výměnné zdroje).

Kromě toho posledního tyto moduly slouží pro rozšíření funkcionality zařízení, například přidávají nové porty daného typu, mohou přidávat paměť (dokonce existují moduly s SSD úložišti). Pro konkrétní zařízení



Obrázek 1.5: NIM modul a NM modul

jsou použitelné pouze některé z těchto typů modulů, tedy si nejdřív zjistíme, jaký typ modulů naše zařízení podporuje, a pak teprve příslušný modul pořídíme.

Na obrázku 1.5 vlevo je NIM modul a vpravo NM modul, u každého pohled z venku a zevnitř. Oba jsou zásuvné, ale každý z nich se uvnitř zjevně napojuje do zcela jiné patice. V každém případě stačí zasunout a zacvaknout (žádné propojování kabelů uvnitř), oproti nejstarším typům modulů dokonce často ani nemusíme zařízení vypínat, jsou tedy typu hot-plug a dokonce plug-and-play (ale i to je dobré si zjistit předem).

Příklad

Podívejme se na jeden z Cisco routerů (Catalyst 8200 Series Edge) ze zadní strany:

U jiných modelů zařízení, popřípadě u jiných výrobců, to samozřejmě může být jiné, především v uspořádání a vybavení, ale základ bude podobný.

Předně zcela vlevo máme modře označený port RJ-45. Jde o konzolový port, který neslouží k běžné komunikaci, ale pro management. Na některých routerech můžeme vedle něj najít i konzolový port mini-B USB (zde není), pozor – neplést si s micro-B USB, nejsou kompatibilní. To, zda je port konzolový, neurčuje jeho tvar (konektor), ale to, jak se přes něj komunikuje (a u Cisca obvykle světle modrá barva).

O kousek dál je tmavě modrý USB typ A. Barva určuje, jakou rychlostí dokáže komunikovat (zde verze 3.0), a určení je taktéž podobné jako u počítače – používáme, pokud potřebujeme z/do routeru dostat nějaká data (soubor s obrazem IOS, nějaký konfigurační soubor apod.). Pozor, ne každé médium lze použít, některá nebudou rozpoznána. Buď dokumentace, nebo pokus-omyl.

Žlutě ohraničené porty jsou určeny pro běžnou komunikaci, tedy připojení do sítě. Zde máme dva porty RJ-45 pro kroucenou dvojlinku a dvě pozice na SFP moduly (příp. SFP+, QSFP apod. – pro různé rychlosti).

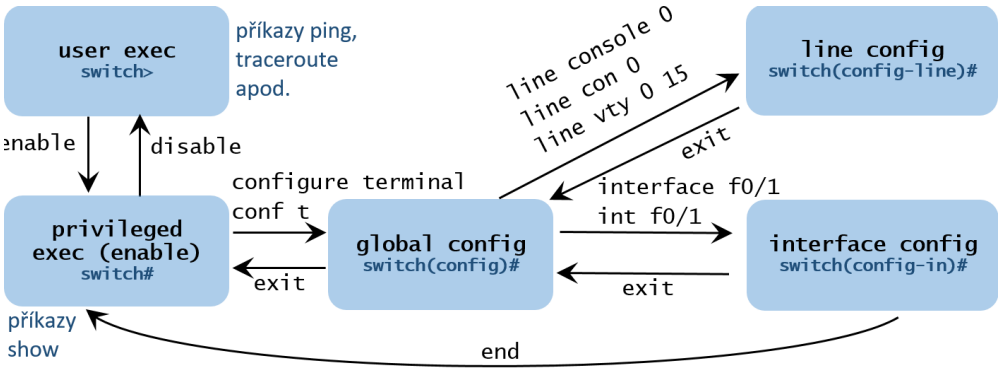
Na routeru také mohou být pozice pro výměnné moduly, zde konkrétně pozice na PIM a víc vpravo NIM modul (je tu jen záslepka, tu je třeba vyšroubovat a nahradit modulem). V dokumentaci bychom zjistili, jaký typ modulu lze použít.

Úkol

Jděte na web https://www.cisco.com/. V menu *Products* si najděte routery, řadu *Branch* a některý router si vyberte. Prohlédněte si specifikace. Najdete tam informace o tom, jaké typy modulů lze do zařízení dokoupit?

1.4 Konfigurace routeru a switche

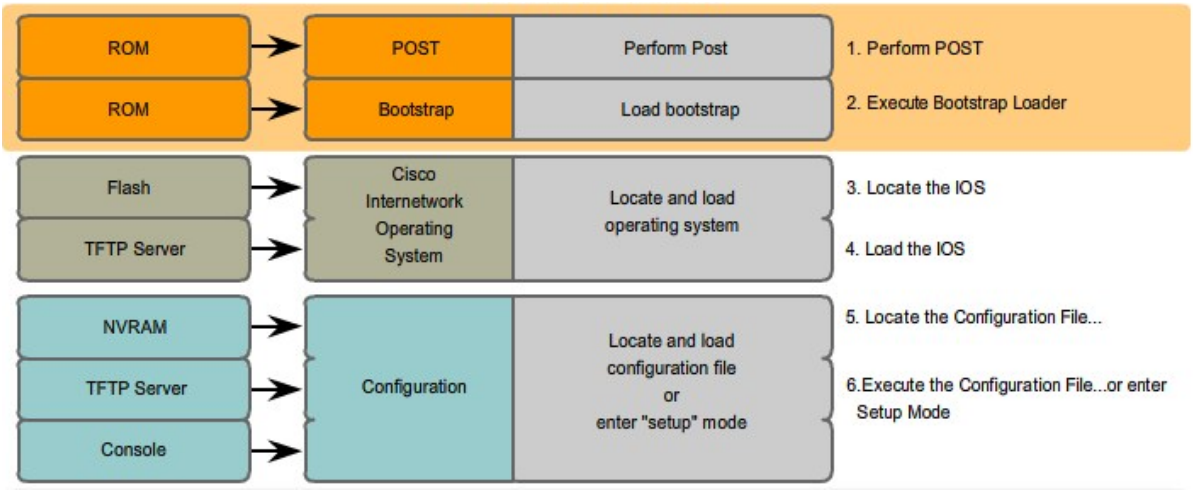
Jak Cisco, tak i další výrobci síťových zařízení, jdou u managed zařízení konfigurovaných v textovém režimu cestou různých módů pro příkazy. Každý mód je určený pro určitý typ příkazů. Pro jistotu si zopakujeme, jak je to s módy u zařízení Cisco.



Obrázek 1.6: Módy příkazů pro zařízení Cisco

Na obrázku 1.6 je přehled nejběžnějších módů s naznačením možností přecházení mezi nimi. Po připojení k zařízení se dostáváme do módu user exec, ve kterém se nelze dostat k interním informacím uloženým v zařízení ani ovlivňovat jeho funkce. V následujícím módu už můžeme používat `show` příkazy a tedy zjišťovat informace uložené v zařízení, ale ne ovlivňovat jeho funkce. Až konfigurační módy slouží ke konfiguraci zařízení.

Obrázek 1.7 shrnuje různé typy úložišť použitých v síťových zařízeních Cisco. Ve všech novějších zařízeních jsou ve skutečnosti úložiště ROM, Flash a NVRAM čipy typu flash PROM, tedy jejich obsah lze upravovat.



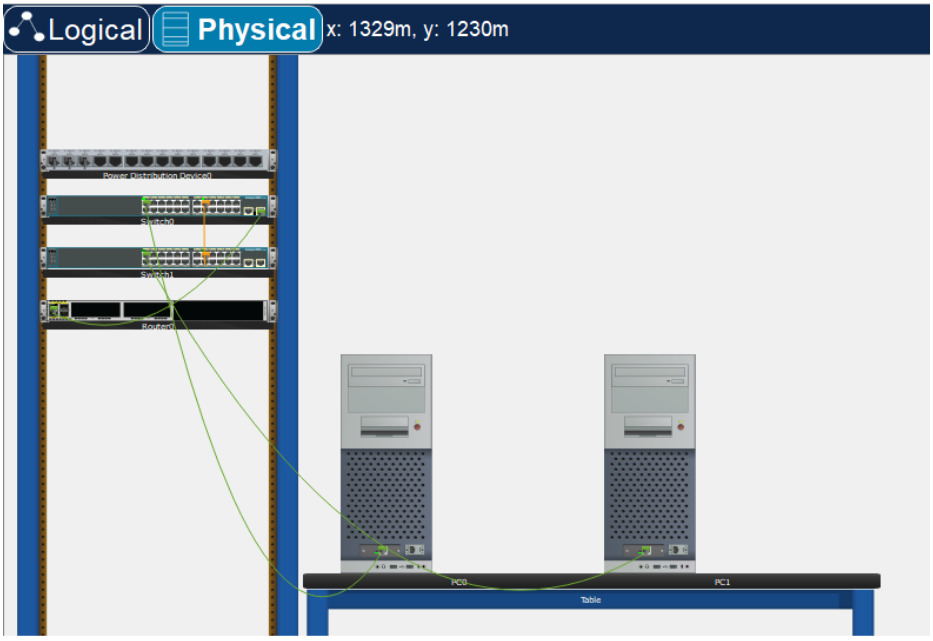
Obrázek 1.7: Úložiště v síťových zařízeních Cisco[1]

V souhrnném příkladu si zopakujeme některé postupy a ukážeme si různé možnosti získávání informací ze síťových zařízení.

[1] Zdroj: prezentační materiály Cisco

Úkol

Budete potřebovat dva switche, jeden router a dva počítače. Použijte buď fyzická zařízení, nebo když je nemáte k dispozici, Packet Tracer. V Packet Traceru využijte raději „fyzický pohled“ (přepínací volby vlevo nahoře), pak případně i logický:



Prověřte router. Je možné do něj dodat nějaké moduly?

Každý switch připojte k jinému rozhraní routeru, použijte správné kabely. Proveďte základní konfiguraci routeru a obou switchů (nebo alespoň jednoho z nich, jak budete stíhat). Nastavte názvy zařízení (R1, S1, S2).

Nastavte heslo do privileged exec módu (`ciscoen`), heslo na konzoli (`ciscocon`). Pro VTY linky vytvořte uživatele `newuser` s bezpečným heslem `secureUser`. Zajistěte, aby žádné z hesel nebylo v konfiguraci přímo čitelné.

Použijte sítě `10.10.0.0/16` a `10.20.0.0/16`, router bude mít první adresu z rozsahu, switche druhou adresu z rozsahu.

Zprovozněte SSH verze 2, vytvořte RSA klíč o délce 1024, na VTY linkách nastavte přihlašování přes lokální databázi (tj. přes vytvořeného uživatele), přístup pouze SSH. Vyzkoušejte, jestli správně funguje přístup přes SSH.

Uložte konfiguraci do `startup-config`.

Úkol

Pokračujeme v předchozím příkladu.

Následující `show` příkazy jsou absolutním základem:

```
show ip int br
show ipv6 int br
show run
```

První zobrazí tabulku rozhraní s informacemi o nastavení IPv4 adres a stavu portů. Druhý příkaz je variantou pro IPv6. Třetí příkaz zobrazí obsah souboru `running-config`, tedy běhovou konfiguraci zařízení.

Vyzkoušejte také následující příkazy (pokud to má smysl, tak na routeru i switchi). Nejdřív zjistíme, ze kterého obrazu zařízení nabootovalo, a pak podrobné informace o zařízení a verzi systému:

```
show boot
show version
```

Podíváme se na soubory, které se v zařízení nacházejí:

```
show flash
dir flash:
dir nvram:
dir usbflash0:
```

První příkaz zobrazí obsah flash PROM, ve které máme obraz systému a základní konfiguraci. Druhý příkaz provede totéž.

Třetí příkaz zobrazí obsah čipu NVRAM (ve skutečnosti je to taky flash PROM čip, jen se jinak jmenuje), měl by tam být soubor `startup-config`, pokud ovšem je vytvořen. Nic tam nebude, pokud jsme ještě ani jednou neukládali konfiguraci.

Poslední příkaz zobrazí obsah USB flash disku, pokud je nějaký připojen.

Jak víme, otazník nám pomáhá zjistit, jak může příkaz pokračovat, a to funguje i u příkazu `dir`. Klidně můžeme zadat toto:

```
dir ?
```

Pro práci se soubory můžeme používat i příkazy `delete`, `erase` a `copy`. Pro zobrazení obsahu souboru lze využívat i příkaz `more`, třeba takto:

```
more flash:config.text
```

Funguje to pro soubory ve flash pamětech včetně USB flash, ale pro soubory v operační paměti (především soubor `running-config`) používáme spíše příkaz `show`.

Máme k dispozici historii příkazů. K ní se dostáváme buď kurzorovými klávesami  a  , ale také si můžeme zobrazit nedávnou historii:

```
show history
```

Máme také příkazy vypisující údaje o uživatelích:

```
show users
show login
```

První z těchto příkazů má smysl v případě, že používáme lokální databázi uživatelských jmen a hesel, druhý nám pomůže v případě problémů s přihlášenými uživateli.

Podobný příkaz, ale s úplně jiným významem – informace o Syslogu, tedy o logování:

```
show logging
```

Další sada příkazů se vztahuje k fyzickým parametrům (teplotě, větrákům, napájení, atd.), nebude fungovat v Packet Traceru:

```
show env fan
show env power all
show env all
show env ?
```

První příkaz zobrazí informace o větrácích, druhý vše, co se týká napájení. Třetí příkaz má dlouhý výstup obsahující všechno, co může být monitorováno o jednotlivých komponentách. Poslední použijeme, když nevíme, na co se můžeme zeptat.

Další sada příkazů se také vztahuje k fyzickým parametrům, tentokrát jednotlivých síťových rozhraní (něco z toho pravděpodobně také nebude fungovat v Packet Traceru):

```
show int g0/1 status
show interfaces status
show int g0/1 capabilities
show interfaces accounting
```

Otazníkem zjistíme další možnosti a klíčová slova. U rozhraní nás může zajímat třeba `flowcontrol`. 

 Příkazy jsou kolikrát celkem dlouhé a jejich výstupy ještě mnohem delší, proto se hodí šetřit čas, klávesnici a ruce. Co se týče délky příkazů, už bychom mohli vědět, že můžeme zkracovat (samozřejmě jen do té míry, aby nebyla „kolize“ s jiným příkazem, který začíná stejně). Takže zbývají výstupy `show` příkazů.

Pro filtrování výstupů příkazů používáme symbol | (rouru) následovaný klíčovým slovem určujícím, co konkrétně chceme z výstupu vybrat. Pak dopíšeme řetězec pro daný filtr. Používají se obvykle tato kláčová slova:

- `begin` – ve výstupu bude nalezen první výskyt toho, co jsme dopsali za slovo `begin`, vypíše se vše od tohoto slova dále,
- `section` – zobrazí se pouze sekce se zadaným názvem,
- `include` – zobrazí se pouze řádky obsahující to, co zadáme za slovo `include`,
- `exclude` – zobrazí se pouze řádky neobsahující to, co dále zadáme.

Příklad

Z obsahu souboru `running-config` chceme zobrazit jen to, co následuje za řetězcem `line con 0` (až do konce výstupu), tedy nás zajímá nastavení linek:

```
show run | begin line con 0
```

Z téhož souboru nás zajímá pouze sekce pro VTY linky, protože například potřebujeme vědět, kolik těch linek je:

```
show run | section line vty
```

Potřebujeme vědět, jak je na switchi nastavený protokol STP (Spanning Tree):

```
show run | section spanning-tree
```

Zajímá nás, které porty jsou aktivní, popřípadě které vypnuté:

```
sh ip int br | include up
sh ip int br | exclude down
```

Chceme zjistit, které služby na zařízení běží a které ne, tedy zobrazit řádky obsahující slovo `service`:

```
show run | include service
```

Úkol

Nastavte na připojených počítačích některou IP adresu z daného rozsahu, brána bude IP adresa routeru pro danou síť. Otestujte konektivitu k routeru.

Nejdřív nás bude zajímat ARP tabulka. Její obsah zobrazíme takto (jedním z těchto příkazů):

```
show arp
show ip arp
```

Teď se zaměřte na switche (alespoň jeden z nich). Na switchi si vypište tabulku MAC adres, což je kterýkoliv z těchto příkazů:

```
show mac address-table
show mac-address-table
```

Jde o tabulku, podle které switch přepíná provoz. Někdy je třeba tuto tabulku vymazat, případně jen dynamicky získané záznamy (což bude pravděpodobně většina nebo dokonce všechny):

```
clear mac address-table
clear mac-address-table
clear mac-address-table dynamic
```

Zařízení si některé parametry dojednávají v rámci autonegociace – duplex, rychlost a také u switche to, zda se má automaticky rozpoznávat typ metalického kabelu (přímý nebo křížený), tedy vlastnost auto-MDIX.

Příklad

Dále vidíme výpis vlastností síťového rozhraní souvisejících s vrstvami L1 a L2 (to poznáme podle toho, že v příkazu není klíčové slovo `ip`).

```
Switch#sh int g0/1
GigabitEthernet0/1 is up, line protocol is up (connected)
  Hardware is Lance, address is 00d0.bc48.a019 (bia 00d0.bc48.a019)
 BW 1000000 Kbit, DLY 1000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Full-duplex, 1000Mb/s
  input flow-control is off, output flow-control is off
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:08, output 00:00:05, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
```

1 2 3

Kromě jiného ve výpisu vidíme, že linka je aktivní – samotné rozhraní i L2 protokol (označeno 1), dále MAC adresu portu (označeno 2), na dalším řádku šířku pásma (1 Gbit/s) a latenci (1000 μs), o něco níže plný duplex a rychlost 1 Gbit/s (označeno 3).

Formát adres je ARPA (tento formát používá například Ethernet a Wi-fi).

Kdyby nastavení duplexu a rychlosti nebylo správně, nastavíme je na rozhraní:

```
conf t
int g0/1
   duplex auto
   speed auto
```

```
conf t
int g0/1
   duplex full
   speed 1000
```

Vlevo nastavíme vše tak, aby fungovala autonegociace, vpravo ručně určíme hodnoty. První možnost je lepší, hlavně proto, aby druhá strana měla určitý manévrovací prostor v rámci autonegociace.

Teď k vlastnosti Auto-MDIX. To, zda správně funguje, zjistíme následovně (nelze použít v Packet Traceru):

```
show controllers ethernet g0/1 phy | include auto-mdix
```

Samozřejmě nemusíme použít filtrování, jen by výstup byl celkem dlouhý. Pokud výstup příkazu vypadá takto:

```
Auto-MDIX : On [AdminState=1 ...]
```

Hodnota na začátku hranaté závorky `AdminState=` určuje, zda je funkce autodetekce křížení zapnutá (`=1`) nebo vypnutá (`=0`). Hodnota `On` určuje, že port právě zajišťuje vnitřní křížení, hodnota `Off` by znamenala, že port na své straně vypnul křížení.

Pokud je Auto-MDIX vypnuté, zapneme ho takto:

```
...
int g0/1
   mdix auto
```

Úkol

Podle svých možností prozkoušejte příkazy z předchozího příkladu.

Pokračujte v souhrnném příkladu z předchozích stránek, přidejte IPv6 adresaci – použijte adresy sítí `2001:db8:a:10::/64` a `2001:db8:a:20::/64`. Adresy nastavte na routeru a na počítačích.

Na routeru nastavte banner s hláškou „Unauthorized entry prohibited“. Na všech zařízeních na konzoli nastavte synchronizaci logování a zakažte DNS lookup.

Souhrn kapitoly 1

Hlavním cílem této kapitoly bylo zopakovat si základní postupy při konfiguraci síťových zařízení, ale také jsme se zaměřili na různé typy modulů se síťovými rozhraními a zásuvných modulů a naučili jsme se zjišťovat různé parametry síťových zařízení.

Kapitola 2

Pokročilá konfigurace switche a funkcí na vrstvě L2

Rychlý náhled: V této kapitole se budeme zabývat různými nastaveními prováděnými zejména na switchi (třebaže místy zabloudíme i na router). Zaměříme se na konfiguraci virtuálních lokálních sítí (VLAN), dále se budeme zabývat protokolem STP a ukážeme si vytváření logických spojů EtherChannel. Následuje sekce o zabezpečení portů pomocí Port Security a sekce o zjišťování informací ze zařízení pomocí protokolů CDP a LLDP, a pak se naučíme blokovat neznámý unicastový a multicastový provoz.

Klíčová slova: VLAN, trunk, IEEE 802.1Q, VLAN rámec, router-on-stick, multilayer switch, STP (spanning-tree), konvergence, root, spanning-tree Portfast, spanning-tree BPDUGuard, EtherChannel, LACP, PaGP, Port Security, Sticky, CDP, LLDP.

Cíle studia: V této kapitole se naučíte konfigurovat VLAN sítě včetně zajištění komunikace mezi jednotlivými VLAN sítěmi, ovlivňovat průběh konvergence protokolu STP, naučíte se konfigurovat EtherChannel a zabezpečit porty switche pomocí Port Security, také využívat protokoly CDP a LLDP ke zjišťování informací o sousedních aktivních síťových zařízeních a blokovat neznámé unicasty a multicasty.

2.1 VLAN

Na přednáškách jsme si vysvětlili, proč používat VLAN, a taky základní princip: zařízení jsou rozdělená do skupin, přičemž fyzické umístění členů jednotlivých skupin může být v rámci celé sítě jakékoliv. Pro každou skupinu můžeme na vrstvě L3 definovat odlišná pravidla, zejména co se týče bezpečnosti a řízení přístupu. Konfiguraci VLAN provádíme především na switchi, komunikaci mezi různými VLAN provádíme na L3 zařízení, které bude tuto komunikaci zprostředkovávat.

2.1.1 Konfigurace na switchi

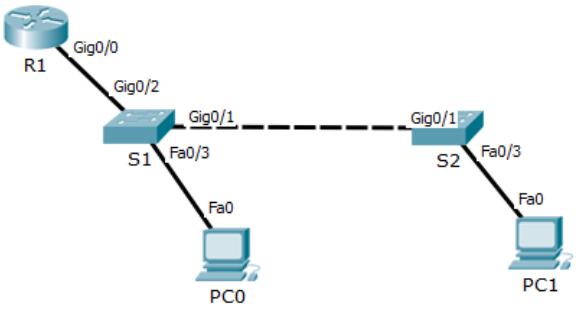
Na switchi provedeme následující:

- vytvoříme jednotlivé VLAN sítě a pojmenujeme je (čísla jsou praktická pro zařízení, pro lidi jsou přehlednější názvy),

- určíme, které porty budou přístupové (access), a určíme, který port bude patřit do které VLAN,
- nakonfigurujeme trunky mezi switchi.

Příklad

Na následujícím obrázku je topologie, podle které budeme konfigurovat. Nejdřív se zaměříme na switche. Propojíme je kříženým kabelem a vytvoříme VLANy.



Půjde celkem o 5 VLAN, každá bude mít svůj účel. Přesuneme se do globálního konfiguračního módu, v sub-konfiguračním módu pro konkrétní VLAN zadáme název.

```
enable
configure terminal
vlan 10
    name IT
vlan 20
    name IoT
vlan 30
    name others
vlan 40
    name guests
vlan 99
    name management
```

Pokud tutéž sekvenci příkazů máme provést na více zařízeních, je praktické si je napsat do textového editoru (bez odsazování a bez promptu, ale například vykřičníky nevadí), a pak je při konfigurování v konzoli jednoduše nakopírovat.

Port `g0/1` bude na obou switchích trunk, k portu `f0/3` připojte ke každému switchi počítač. Použijte IP adresy `10.10.0.3/16` a `10.10.0.4/16`. Adresa brány je `10.10.0.1`. Ověřte, zda jsou navzájem dostupné.

Dále na jednotlivých portech určíme, které budou patřit do konkrétní VLAN a které budou trunkové:

```
int g0/1
    switchport mode trunk
    switchport trunk native vlan 1
    switchport trunk allowed vlan 10,20,30,40,99 ! není nutné, pokud chceme všechny
int f0/3
    switchport mode access
    switchport access vlan 10
...

int vlan99   ; management vlan
    ip addr 10.99.0.101 255.255.0.0      ! to platí pro S1
    ip addr 10.99.0.102 255.255.0.0      ! to platí pro S2
    no shut
    exit
ip default-gateway 10.99.0.1     ! virtuální rozhraní switche patří do VLAN 99
```

Show příkazy pro VLAN:

```
sh int trunk
sh int status | include trunk
sh vlan brief
```

Případně další varianty. Ke každému switchi připojte jeden počítač (porty `f0/3`).

Ověřte, zda už mohou mezi sebou komunikovat switche. Proč ano/ne? Do které VLAN patří?

Otázky

K předchozímu příkladu: jak například mohou vypadat adresy pro připojené počítače? Mohou být taky ze sítě `10.99.0.0/16`?

Poznámka

Pokud se na portu nedaří nastavit trunk, bývá (například na multilayer switchi) důvodem to, že je zapouzdřování do VLAN rámce nastaveno na jiné než IEEE 802.1Q. Na portu pak postupujeme takto:

```
switchport trunk encapsulation dot1q
do sh int ... switchport    ! ověříme, jestli už se "zapouzdřuje" dot1q (tedy IEEE 802.1Q)
```

2.1.2 Směrování mezi VLAN pomocí Router-on-a-Stick

Účelem je umožnit směrování mezi L2_VLAN/L3_podsítěmi, potřebujeme L3 zařízení. Pokračujeme v předchozím příkladu.

Příklad

Na straně switche bude trunk s určitými povolenými VLAN, na straně routeru jedno rozhraní rozdělené na subrozhraní – pro každou VLAN jedno subrozhraní.

Ke switchi S1 zapojíme router (na switchi `g0/2`, na routeru `g0/0` či podobný). Ke switchi S2 připojíme ještě jeden počítač (port `f0/4`). Na počítači nastavíme IP adresu `10.20.0.4/16`, bránu `10.20.0.1`, na switchi S2 nastavíme u portu `f0/4` VLAN 20.

Ověříme vzájemnou dostupnost počítačů `10.10.0.3` a `10.20.0.4`.

Na switchi S2 nastavíme trunk směrem k routeru:

```
S2(config)# int g0/2
S2(config-if)# switchport mode trunk
S2(config-if)# switchport trunk native vlan 1
S2(config-if)# switchport trunk allowed vlan 10,20,99
```

Poslední dva příkazy nejsou nutné. VLAN 1 je jako nativní nastavena defaultně, a nastavování dostupných VLAN může být někdy kontraproduktivní (když vytvoříme další novou VLAN, musíme ji přidat do všech trunků). Pokud bychom ovšem chtěli nastavit jinou nativní VLAN než 1, pak má příslušný příkaz smysl, a také nastavení povolených VLAN může být užitečné. Například z bezpečnostních důvodů můžeme nastavit jako nativní takovou VLAN, kterou nebudeme pro nic jiného používat, a zároveň ji buď na trunku zakážeme, nebo vše, co se do této VLAN dostane, budeme přeposílat do karantény.

Na routeru:

```
R1(config)# int g0/0.10
R1(config-if)# encapsulation dot1q 10
R1(config-if)# ip addr 10.10.0.1 255.255.0.0
R1(config-if)# int g0/0.20
```

```
R1(config-if)# encapsulation dot1q 20
R1(config-if)# ip addr 10.20.0.1 255.255.0.0
R1(config-if)# int g0/0.99
R1(config-if)# encapsulation dot1q 99
R1(config-if)# ip addr 10.99.0.1 255.255.0.0
R1(config-if)# int g0/0
R1(config-if)# no shut
```

Poslední příkaz se opravdu má provádět na (celkovém) fyzickém rozhraní, nemá smysl zapínat jednotlivá subrozhraní.

Úkol

Ověřte vzájemnou dostupnost počítačů `10.10.0.3` a `10.20.0.4`. Taktéž si vypište seznam rozhraní s nastavením trunku a stav jednotlivých rozhraní.

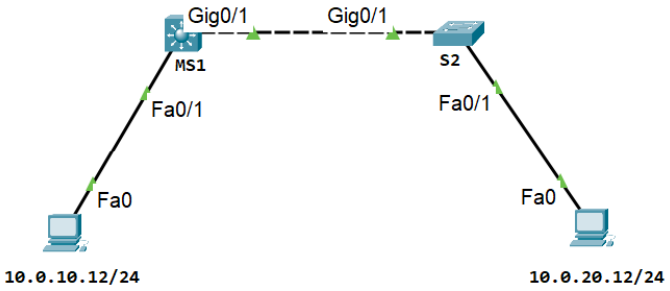
Co se stane, když odstraníte některou VLAN, například `no vlan 10`?

2.1.3 Směrování mezi VLAN pomocí multilayer switche

Pokud máme multilayer switch, můžeme sloučit definování VLAN a směrování mezi nimi do jediného zařízení. Pak uvnitř takového switche zprovozníme „vnořený“ směrovací modul a vytvoříme virtuální rozhraní pro jednotlivé VLANy.

Příklad

Vytvoříme si síť podle obrázku níže. Máme jeden multilayer switch MS1 a jeden „běžný“ L2 switch S2.



Na prvním počítači je IP adresa `10.0.10.12/24` a brána `10.0.10.1`. Na druhém počítači je IP adresa `10.0.20.12/24` a brána `10.0.20.1`.

Nejdřív si nakonfigurujeme switch S2. Vytvoříme VLANy a do VLAN 20 přiřadíme port `f0/1`:

```
S2(config)# vlan 10
S2(config-vlan)# name hlavni
S2(config-vlan)# vlan 20
S2(config-vlan)# name hoste
S2(config-vlan)# int f0/1
S2(config-if)# switchport mode access
S2(config-if)# switchport access vlan 20
```

Dále nakonfigurujeme trunk:

```
S2(config-if)# int g0/1
S2(config-if)# switchport mode trunk
```

Přesuneme se na multilayer switch. Vytvoříme VLANy a do první přiřadíme port `f0/1`:

```
MS1(config)# vlan 10
MS1(config-vlan)# name hlavni
MS1(config-vlan)# vlan 20
```

```
MS1(config-vlan)# name hoste
MS1(config-vlan)# int f0/1
MS1(config-if)# switchport mode access
MS1(config-if)# switchport access vlan 10
```

Nakonfigurujeme trunk. Protože jde o multilayer switch, musíme předem nastavit protokol IEEE 802.1Q:

```
MS1(config-if)# int g0/1
MS1(config-if)# switchport trunk encap dot1q
MS1(config-if)# switchport mode trunk
```

A teď přistupme ke zprovoznění L3 směrování v multilayer switchi. Pro každou VLAN vytvoříme virtuální L3 rozhraní a přiřadíme IP adresu, která zároveň bude sloužit jako brána pro podsíť příslušející k dané VLAN:

```
MS1(config-if)# int vlan10
MS1(config-if)# ip addr 10.0.10.1 255.255.255.0
MS1(config-if)# int vlan20
MS1(config-if)# ip addr 10.0.20.1 255.255.255.0
MS1(config-if)# exit
```

Vytvořená virtuální rozhraní by se měla automaticky zapnout. Kdyby ne, je třeba na nich použít příkaz `no shutdown`, nicméně kdyby to nešlo a rozhraní by zůstala vypnutá, musíme zkontrolovat, jestli není chyba jinde. Například – opravdu je příslušná VLAN vytvořená?

Zbývá zprovoznit IPv4 směrování (na routeru je defaultně spuštěno, ale na switchi ne), pak si necháme vypsat směrovací tabulku.

```
MS1(config)# ip routing

MS1(config)# do sh ip route | begin Gateway

Gateway of last resort is not set

10.0.0.0/24 is subnetted, 2 subnets
C 10.0.10.0 is directly connected, Vlan10
C 10.0.20.0 is directly connected, Vlan20
```

Jak vidíme, existují dvě (virtuálně přímo připojené) sítě a lze mezi nimi směrovat. Následující ping mezi koncovými zařízeními by měl fungovat (pokud ovšem máme na počítačích správně nastavené IP adresy a brány):

```
ping 10.0.20.12
```

Na multilayer switchi si můžeme vyzkoušet tyto show příkazy:

```
sh ip int br
sh ip route
```

První z těchto dvou příkazů by nám měl na obou virtuálních rozhraních ukázat jejich adresy, rozhraní by měla být ve stavu `up–up`. Druhý příkaz zobrazí směrovací tabulku, kde by měly být obě připojené virtuální sítě.

2.2 STP

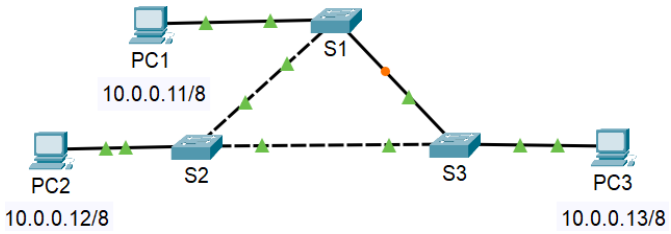
2.2.1 Konfigurace Spanning-tree

Mechanismus Spanning-tree funguje celkem samostatně, ale určitě se nám hodí `show` příkazy, a také je dobré někdy zasáhnout do výběru root switche.

Příklad

Vytvoříme síť switchů s počítači a nejdřív necháme STP, aby určil hierarchii switchů sám.

Předpokládejme, že tedy jako první byl „vytvořen“ switch S1, atd. Který switch je root switch, taky podle obrázku? Proč to není S1?



Vyzkoušejte tyto příkazy:

```
show spanning-tree
show spanning-tree detail
show spanning-tree summary
```

případně další, zjistěte, co je podporováno (otazník).

Dále budeme chtít ovlivnit výběr root switche, naším favoritem je právě S1. Nastavíme jeho prioritu na 20450:

```
spanning-tree vlan 1 priority 20450
```

Proč myslíte, že zrovna takovou? Jak to udělat jednodušeji – ať si nemusíme pamatovat vhodnou hodnotu priority pro root:

```
spanning-tree vlan 1 root primary
```

Ověřte, jak se změnily parametry.

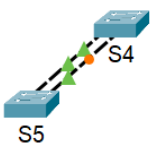
Pokud budeme chtít, aby záložním rootem byl switch S2, provedeme na něm tento příkaz:

```
spanning-tree vlan 1 root secondary
```

Kdyby primární root z nějakého důvodu přestal fungovat, novým primárním se stane ten, který jsme určili jako sekundární.

Úkoly

1. Když jste podle předchozího příkladu určili primární a sekundární root switch, ověřte si pomocí `show` příkazů, jaká je jejich skutečná priorita.
2. Pokuste se vyrobit „kravatu“, tj. dva switche propojené dvěma kabely (viz obrázek vlevo). Ověřte, jak se dohodly na nastavení spanning-tree.



2.2.2 Spanning-tree Portfast

Portfast konfigurujeme na portech, u kterých předpokládáme připojení pouze koncových zařízení nebo takových, která nejsou plně „důvěryhodná“, prostě na přístupových portech. Na takové porty nejsou posílány BPDU rámce. Důvodem je nejen bezpečnost, ale také urychlení konvergence sítě switchů.

Příklad

Předpokládejme, že porty `f0/5-20` nejsou připojeny k jinému switchi a předpokládá se u nich připojení koncových zařízení (třeba přes ethernetovou zásuvku), tedy právě zde budeme chtít zapnout funkci portfast:

```
interface range f0/5-20
    spanning-tree portfast
```

Nebo můžeme pro všechny přístupové porty nastavit tuto funkci jako defaultní (tento příkaz použijeme v globálním konfiguračním módu):

```
spanning-tree portfast default
```

a pak na těch portech, u kterých tuto funkci nechceme, ji vypnout, třeba `f0/1`:

```
interface f0/1
    spanning-tree portfast disable
```

Ověříme pro konkrétní port:

```
show spanning-tree interface f0/1 portfast
show run
```

V running-configu by toto vše mělo být zjistitelné. Protože jde o velký výpis, bylo by vhodné použít vhodný filtr s rourou.

2.2.3 Spanning-tree BPDUGuard a errorDisable Recovery

Funkce BPDUGuard zajiťuje, aby hacker (nebo osoba neznalá), který svůj switch připojí k přístupovému portu s nastaveným portfast a pokusí se stát root switchem, se dočkal patřičně rázné odpovědi. Pokud je tato funkce zapnutá, pak v takovém případě daný port přechází do stavu error-disabled a přestane fungovat.

Příklad

Nastavíme BPDUGuard pro množinu portů:

```
interface range f0/5-20
    spanning-tree bpduguard enable
```

Pro všechny porty můžeme BPDUGuard nastavit jako defaultní:

```
spanning-tree portfast bpduguard default
```

a potom zrušíme pro ty porty, za kterými je regulérní switch, třeba f0/1:

```
interface f0/1
 spanning-tree bpduguard disable
```

Pokud je port ve stavu error-disable, nebude fungovat, a taky není jednoduché ho aktivovat (administrátor by musel port příkazem `shutdown` vypnout a pak příkazem `no shutdown` zapnout).

Zda je některý port ve staru error-disable, zjistíme v následujícím výpisu:

```
show interface status
show interface f0/1 status
```

Je možné zajistit, aby se takto vypnutý port ve stavu error-disable po určité době automaticky zotavil, což je funkce „errordisable recovery“ (nefunguje v Packet Traceru).

Příklad

Nejdřív zjistíme, jestli je tato funkce zapnutá, a když ne, zapneme ji, nastavíme interval pro zotavení na 30 sekund:

```
show errdisable recovery
conf t
errordisable recovery cause bpduguard
errordisable recovery interval 30
```

Začátek výstupu bude vypadat nějak takto:

```
Switch(config)# do sh errdisable recov

ErrDisable Reason            Timer Status
-----------------            --------------
arp-inspection               Disabled
bpduguard                    Enabled
channel-misconfig (STP)      Disabled
dhcp-rate-limit              Disabled
dtp-flap                     Disabled
gbic-invalid                 Disabled
inline-power                 Disabled
link-flap                    Disabled
mac-limit                    Disabled
```

2.3 EtherChannel

EtherChannel (EC) je protokol používaný k vytvoření logického komunikačního kanálu sestávajícího z několika fyzických komunikačních linek, tj. slučujeme několik fyzických linek do jedné logické s vyšší propustností. Je to čistě point-to-point technologie, zabýváme se tedy právě dvojicí skupin portů na protilehlých zařízeních. Pokud z jednoho switche vede více EC logických spojů, jsou na sobě nezávislé (musejí mít jen jiné označení, aby vše bylo jednoznačné).

Dvojici switchů tedy propojíme několika kabely. Použité porty musejí být všechny stejného typu (rychlost, duplex, MDIX). Nemusejí mít nutně na obou stranách stejné názvy. Dále všechny zapojené porty na obou stranách musejí mít stejný typ EC.

EC buď nastavíme ručně (mód „on“ na obou stranách), nebo použijeme některý protokol pro vyjednání EC: na Cisco zařízeních máme na výběr mezi protokolem LACP (otevřený protokol, který podporují všichni výrobci, takže se dá použít i v případě propojení switchů různých výrobců) a protokolem Cisco PAg-P. Oba protokoly rozlišují aktivní a pasivní stranu, jen je trochu jinak nazývají.

Tabulka 2.1: Označení stavů EC portů při použití různých protokolů

Význam	Protokol PAg-P	Protokol LACP
ručně	on	on
pasivní strana	auto	passive
aktivní strana	desirable	active

V tabulce 2.1 jsou možná označení stavů portů při použití EC. Samozřejmě je nejze libovolně kombinovat, povolené kombinace najdeme v tabulce 2.2.

Tabulka 2.2: Módy pro vyjednání EtherChannelu

Porty na prvním zařízení	Porty na druhém zařízení	⇒ vybrali jsme protokol
`active`	`passive nebo active`	LACP
`desirable`	`auto nebo desirable`	PAgP
`on`	`on`	žádný

EtherChannel můžeme tedy nastavit ručně (to je volba `on`), ale ztratíme výhodu automatického znovuzprovoznění v případě problémů (krátkodobá ztráta konektivity apod.).

Příklad

Předpokládejme, že mezi dvěma switchi máme nataženy dva kabely, na obou stranách porty f0/1 a f0/2. Na portech nakonfigurujeme EtherChannel s navazováním kanálů pomocí LACP:

```
interface range f0/1-2
    shutdown
    channel-group 1 mode active
```

Tím jsme vytvořili nový logický port s názvem `port-channel 1`, resp. `po1`. Vše další budeme nastavovat právě na tomto novém portu, včetně trunku.

```
interface port-channel 1
    switchport mode trunk
    switchport trunk encap dot1q
    switchport trunk native vlan 1
    switchport trunk allowed vlan 10,20,99
```

Nebo pokud má jít o přístupový port (což není úplně běžné), zde půjde o accessový port pro VLAN 10:

```
interface port-channel 1
    switchport mode access
    switchport access vlan 10
```

V každém případě zbývá logický port aktivovat, což uděláme tak, že zapneme fyzické porty patřící do EC svazku:

```
interface range f0/1-2
    no shutdown
```

Pokud bychom chtěli použít protokol PAg-P, byl by na jedné straně mód desirable a na druhé auto. Pokud bychom chtěli vytvořit EtherChannel ručně, na obou stranách by bylo „on“.

Obvykle bývá EtherChannel vytvořen na L2 portech, ale dá se vytvořit i na L3 portech:

```
interface range f0/1-2
    shutdown
    channel-group 1 mode active
interface port-channel 1
    no switchport
    ip address 10.0.0.8 255.0.0.0
interface range f0/1-2
    no shutdown
```

Show příkazy pro Etherchannel – vyzkoušejte:

```
show interface f0/1 etherchannel
show interface po1
show etherchannel 1 detail
show etherchannel protocol
show etherchannel summary
show etherchannel ?
show lacp ?
show interfaces f0/1 capabilities
show run
show interfaces trunk
```

Poznámka

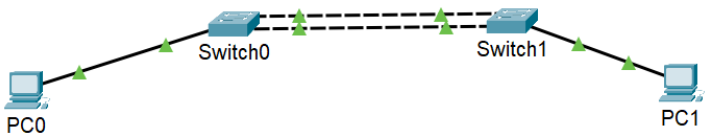
Nejdřív zařazujeme porty do EtherChannelu, a až potom na nich cokoliv nastavujeme. Pro jistotu je lepší nastavovat EtherChannel na vypnutých portech, a až po nastavení všechny fyzické porty (najednou v `range` rozsahu) zapnout.

Na fyzických portech zvlášť pak už nic neřešíme, kromě zapínání a vypínání. Kdybychom například dodatečně změnili nastavení trunku na jednom ze zapojených portů, celý logický kanál by přestal fungovat.

Úkol

Nakonfigurujte tuto jednoduchou síť pro Etherchannel, použijte LACP.

Počítače PC0 i PC1 budou ve VLAN 20, mezi switchi vytvořte trunk. Vyzkoušejte různé `show` příkazy, také ověřte, co hlásí CDP/LLDP, taky STP. Nejdřív řešte EtherChannel, až potom trunk.



2.4 Port Security

Účelem je zajistit, aby v případě, že se k portu připojí někdo „nepovolaný“ (podle MAC adresy), došlo k vhodné bezpečnostní reakci – podle toho, co nastavíme.

Port musí být v přístupovém módu a musí mít zapnutou funkci port security, pak můžeme tuto funkci nastavovat.

Příklad

Ukážeme si nastavení Port Security pro port `g0/1`. Nejdřív tuto funkci na portu zapneme:

```
int g0/1
    switchport mode access
    switchport port-security
```

Teď můžeme například určit maximum pro počet bezpečných MAC adres (pro 2 adresy):

```
switchport port-security maximum 2
```

Jsou tři možnosti určení bezpečných MAC adres:

1. *staticky*, tedy ručně určíme bezpečnou MAC adresu (uloží se do running-config, po restartu zmizí, ale u portu je uložena, tedy při vypnutí/zapnutí portu se neztratí),
2. *dynamicky*, první připojená MAC se zapamatuje (uloží se do RAM, ale ne do running-config, po restartu zmizí, při vypnutí/zapnutí portu taktéž a učí se znovu),
3. *sticky port-security*: učí se jako dynamicky (uloží se do running-config, po restartu zmizí, ale při vypnutí/zapnutí portu se neztratí) – něco mezi statickým a dynamickým určením MAC.

Pokud máme MAC adresu naučenou v running-configu (staticky nebo sticky), pak je vhodné uložit nastavení do startup-config.

Dynamické a statické určení MAC se dá kombinovat.

Příklad

Na portu `f0/1` nastavíme přístupový mód s VLAN 10 a pak port security, přičemž jednu MAC adresu určíme staticky. Povolíme maximálně 5 adres, tedy zbývající se budou učit dynamicky.

```
int f0/1
    switchport mode access
    switchport access vlan 10
    switchport port-security
    switchport port-security maximum 5
    switchport port-security mac-address 0001.42a1.5ebd
```

Pokud místo statického určení budeme chtít, aby všechny povolené adresy byly učeny stylem sticky:

```
int f0/1
    switchport mode access
    switchport access vlan 10
    switchport port-security
    switchport port-security maximum 2
    switchport port-security mac-address sticky
```

Reakce při připojení MAC adresy, která není bezpečná, mohou být následující:

- *protect*: provoz od „cizí“ MAC adresy je zahazován, jinak se nic neděje; pokud se později připojí „povolená“ adresa, bude vše fungovat běžným způsobem,
- *restrict*: „cizí“ provoz je zahazován, zvýší se counter sledující počet narušení bezpečnosti, je vyvoláno hlášení o narušení bezpečnosti (syslog); pokud se později připojí povolený stroj, opět vše bude fungovat běžným způsobem,
- *shutdown*: port je vypnut, dostane se do stavu error-disabled, musí se ručně vypnout+zapnout (tj. použít příkazy `shutdown` a `noshutdown`); zvýší se counter (defaultní stav).

Příklad

Ukážeme si nastavení Port Security na portu `f0/1` se změnou režimu reakce na nepovolenou MAC adresu. Pokud chceme nastavovat MAC adresu stylem sticky a chceme, aby port fungoval v režimu restrict:

```
int f0/1
    switchport mode access
    switchport access vlan 10
    switchport port-security
    switchport port-security maximum 2
    switchport port-security mac-address sticky
    switchport port-security violation restrict
```

Pro jiné módy bychom zadali klíčová slova `protect` nebo `shutdown` (ale `shutdown` je výchozí, to není nutno zadávat).

Ověřujeme těmito `show` příkazy:

```
show port-security int f0/1
show run | section interface
show port-security
show port-security address
show port-security int f0/1
```

Úkol

Vyberte dva až tři počítače a zjistěte si jejich MAC adresy.

Na switchi nastavte na třech různých portech Port Security:

- na jednom nechte dynamicky s maximem jedné adresy (z cvičných důvodů; obvyklejší je mít maximum alespoň 2),
- na druhém nastavte maximum na 2, ale staticky určete jednu MAC adresu z určených počítačů,
- na třetím nastavte sticky s maximem jedné adresy.

Uložte running-config do startup-configu (kvůli sticky učení adres). Na každém portu nastavte jinak reakci na připojení nepovolené adresy.

Zkoušejte, co se bude dít při připojení počítačů do limitu, vypisujte si naučené adresy. Také vyzkoušejte, co se bude dít po restartu zařízení, které adresy budou zapamatovány u kterého portu. Pak ověřte chování při připojení zařízení nad maximum.

2.5 Protokoly pro zjišťování parametrů aktivních síťových zařízení

2.5.1 CDP

Protokol CDP (Cisco Discovery Protocol) umožňuje zařízením od Cisca se vzájemně představit. Jeho otevřená obdoba je LLDP podporovaná na zařízeních různých výrobců.

Příklad

Vezměte dvě zařízení: jeden switch (nazvěte S1) a jeden router (nazvěte R1). Na routeru nastavte IP adresu na příslušném rozhraní, na switchi nastavte IP adresu na SVI, zfunkčněte. Ke switchi S1 připojte ještě jeden switch (klidně nějaký multilayer, jestli najdete). Na switchi S1 vyzkoušejte:

```
sh cdp neighbors
sh cdp entry R1
```

Místo R1 můžete dát hvězdičku = všichni sousedi.

Pokud jsme jako poslední změnu provedli nastavení názvu, může switch vidět nějakou dobu pod starším názvem, pak určitou dobu dokonce pod oběma názvy (dvě zařízení na stejném portu). Vyzkoušejte příkaz s klíčovým slovem `entry` i na druhém zařízení.

```
sh cdp interface f0/1
```

(dosaďte rozhraní na switchi připojené k sousedovi).

Také existuje varianta, kterou si v Packet Traceru nevyzkoušíme, funguje jen na skutečném zařízení:

```
sh cdp traffic
```

Zapnutí/vypnutí CDP se dá provést buď na jednotlivých rozhraních, nebo na celém zařízení. Na celém zařízení zadáme v globálním konfiguračním módu pro zapnutí, resp. vypnutí:

```
cdp run
no cdp run
```

Na jednotlivých portech se používá trochu jiný příkaz. Přejdeme do subkonfiguračního módu pro dané rozhraní a zadáme (pro zapnutí, resp. vypnutí):

```
cdp enable
no cdp enable
```

Příklad

Pokračujeme v předchozím příkladu. Vypněte CDP na switchi centrálně, zkontrolujte, co vypisuje příslušný show příkaz. Pak zapněte na portu f0/1 (resp. na tom, ke kterému je zapojený router) a opět zkontrolujte. Vezměte v úvahu, že bude určitý delay.

CDP Spoofing se provádí tak, že útočník posílá CDP rámce s cílovou skupinovou MAC adresou `01-00-0C-CC-CC-CC`, zdrojová adresa bývá obvykle podvržená. Pokud se zařízení nedokáže bránit samo, je jediná ochrana vypnout CDP na těch portech, kde nejsou „důvěryhodná“ zařízení typu switch, router apod., zejména na přístupových portech.

Co je to za adresu: Cisco používá `01-00-0C-CC-CC-CC` k více účelům (CDP, VTP, DTP,...), v LCP rámci použije jako OUI `00:00:0C` (= Cisco), místo EtherType identifikátor konkrétního protokolu, uvnitř CDP, VTP či jiný rámec.

2.5.2 LLDP

Protokol LLDP je otevřená varianta k CDP podporovaná různými výrobci, tento protokol používáme v heterogenních sítích se zařízeními od různých výrobců.

Příkazy pro LLDP jsou obdobné, princip používání je podobný jako u CDP. Vypínat a zapínat můžeme také buď centrálně nebo na jednotlivých rozhraních. Nejdřív `show` příkazy:

```
sh lldp
sh lldp neighbors
```

atd., použijte otazník.

Zapnutí a vypnutí LLDP na celém zařízení v globálním konfiguračním módu:

```
lldp run
no lldp run
```

Na rozhraní se na rozdíl od CDP zapíná zvlášť vstup a výstup, v subkonfiguračním módu rozhraní zadáme:

```
lldp receive
lldp transmit
```

Vypnutí by se provedlo tak, že před tento příkaz bychom dali „`no`“.

Příklad

Pokračujte v příkladu o CDP. Vypněte protokol CDP centrálně a zapněte LLDP na připojených rozhraních. Vyzkoušejte různé `show` příkazy.

2.6 Blokování neznámých unicast a multicast rámců

Následující bohužel nebude fungovat v Packet Traceru. Tyto příkazy jsou podporovány na některých switchích.

Vzpomeňme si, jak switch zachází se svou přepínací tabulkou: pokud cílovou adresu rámce najde v tabulce, pak rámec přepošle na port v tabulce uvedený. Jestliže ji v tabulce nenajde, provede flooding podobně jako u broadcastu. Jenže to za určitých okolností není žádoucí – z bezpečnostních důvodů.

Proto v některých firmách je běžné nastavení blokování neznámých unicastů na accessových portech:

```
interface range ...
    switchport block unicast
```

Otázky

Pokud na takový port přijde rámec s neznámou cílovou adresou, bude zahozen. Co myslíte, jaký to má důsledek na provoz sítě, hlavně při připojení nového zařízení do sítě? A jak to dopadá, když se někdo pokusí o útok na MAC tabulku? K čemu konkrétně může být toto nastavení dobré?

Switch nemá jen tabulku unicastových MAC adres, má také tabulku multicastových MAC adres, se kterou se zachází dost podobně, včetně floodingu neznámých MAC. Také zde můžeme zablokovat neznámý provoz:

```
interface range ...
    switchport block multicast
```

Na tomtéž portu mohou být provedena obě nastavení. Pro zrušení stačí před příkaz dát „`no`“.

Jak si ověřit nastavení:

```
sh int ... switchport
sh run | section interface ...
```

Nastavení blokování floodingu je nezávislé na VLANách.

Souhrn kapitoly 2

V této kapitole jsme se zabývali různými nastaveními specifickými hlavně pro switch, nicméně některá nastavení souvisela i s routerem. Nejdřív jsme se naučili konfigurovat virtuální lokální sítě (VLAN), následovala sekce o protokolu Spanning-tree (STP). Pak jsme se naučili nastavit EtherChannel, v další sekci jsme nastavovali Port Security, pak jsme si ukázali použití protokolů CDP a LLDP pro zjišťování parametrů zařízení a na konci kapitoly jsme si ukázali, jak na switchi zablokovat přeposílání neznámých unicast a multicast rámců.

Kapitola 3

Pokročilá konfigurace routeru

Rychlý náhled: V této kapitole se budeme zabývat tématy souvisejícími zejména s protokolem IP a jeho adresami, ať už unicastovými nebo multicastovými. Nejdřív se zaměříme na možnosti získání IPv4 a IPv6 adres a celkově komunikaci v rámci IP sítě. Dalším tématem bude statické směrování. Následující sekce je věnována filtrování provozu pomocí ACL seznamů, tedy seznamů řízení přístupu. V další sekci se budeme věnovat dynamickému směrování, kde se zaměříme na protokol OSPF pro IPv4 i IPv6, a v následující sekci půjde o směrování mezi autonomními systémy, tedy vnější směrovací protokol BGP (ve variantách eBGP a iBGP).

Klíčová slova: IPv4, IPv6, DHCPv4, DHCP Helper, SLAAC, stateless/stateful DHCPv6, M/O/A flag, DUID, zóna, Standard/Extended ACL, routing, OSPF, RIP, EIGRP, eBGP, iBGP.

Cíle studia: V této kapitole se naučíte nastavit přidělování IPv4 a IPv6 adres v síti, porozumíte problematice filtrování provozu pomocí ACL seznamů, naučíte se vytvářet a nasazovat standardní i extended ACL seznamy. Také se naučíte konfigurovat běžné směrovací protokoly, zejména protokol OSPF, a seznámíte se s protokolem BGP.

3.1 Konfigurace DHCP pro IPv4 a IPv6 adresy

3.1.1 Jak nastavit DHCPv4 na routeru

Je třeba stanovit tzv. *pool*, což je rozsah adres, které budou zájemcům přidělovány. Rozsah je určen adresou sítě s maskou, ale některé adresy z tohoto rozsahu budeme chtít vyjmout z přidělování (třeba proto, že patří samotnému routeru/bráně a případně některým serverům či různým síťovým zařízením). Například pokud chceme přidělovat rozsah `10.10.0.0/16`, ale z něj chceme vynechat prvních 8 adres a pak ještě jednu:

```
ip dhcp excluded-address 10.10.0.1 10.10.0.8
ip dhcp excluded-address 10.10.0.254
```

Teď vytvoříme DHCP pool pro určený rozsah adres, nazveme ho `LAN-1`. Náš název je „cvičený“, v praxi by měl hlavně sdělovat, o jakou síť (příp. VLAN) se jedná. Vytvoření DHCP poolu nás převede do

subkonfiguračního módu pro DHCP pool, ve kterém určíme parametry: adresu brány, přidělovaný rozsah adres, a pak další (už nepovinné) parametry jako adresa DNS serveru, název domény, *lease time* (doba přidělení adresy z poolu) atd.

```
ip dhcp pool LAN-1
    default-router 10.10.0.1
    network 10.10.0.0 255.255.0.0
    dns-server xxxxxxx
    domain-name xxxxxxx
    lease 2
```

První dva parametry (síť a adresa routeru) jsou povinné, zbývající nepovinné, záleží, co vše chceme klientům sdělovat zároveň s přidělením adresy. Parametr `dns-server` sdělí klientovi adresu DNS serveru v místní síti, dále můžeme informovat o názvu domény, a taky můžeme určit dobu trvání přiřazení adresy.

Poznámka

Jak je to s pořadím příkazů? Rozhodně nejdřív řešíme `excluded` adresy, a až potom vytváříme pool. Uvnitř poolu vřele doporučuji uvést příkaz `network` spíše až ke konci, protože se může stát, že okamžitě po jeho zadání přijde žádost o adresu od některého klienta, router odpoví rychleji než my stihneme dodat další informace, a klient tedy dostane neúplnou informaci. Především adresu brány (`default-router`) bychom rozhodně měli uvést ještě před příkazem `network`.

Samozřejmě existují `show` příkazy:

```
show run | section dhcp
show ip dhcp pool
show ip dhcp binding
show ip dhcp server statistics
show ip dhcp conflict
```

atd., použijte otazník. Různé systémy (verze) mohou mít trochu odlišnou podporu těchto příkazů, a hlavně v Packet Traceru určitě některá klíčová slova nelze použít. Varianta s klíčovým slovem `binding` nám zobrazí *stav přidělení adres* (proto jde o *stavový* DHCP – podle této tabulky).

Na jednom routeru může být definováno více DHCP poolů, protože každé rozhraní routeru „vidí“ do jiné sítě (nebo několika sítí, když používáme VLANy) a více takových sítí může používat soukromé adresy. Pro každou síť potřebujeme jiný rozsah.

Úkol

Tento úkol plňte v Packet Traceru, ať si výsledek můžete uložit a později se k němu vrátit, až se budeme učit konfigurovat DHCPv6.

1. Vytvořte síť s jedním routerem, dvěma switchi a několika počítači. Na switchích nakonfigurujte:
   - VLAN 10 s názvem Zamestnanci,
   - VLAN 20 s názvem Hoste,
   - VLAN 84 s názvem Management,
   - VLAN 90 s názvem Native,
   - VLAN 95 s názvem Parking,
   - mezi switchi bude trunk, nastavte native VLAN na 90, povolené VLANy budou 10, 20 a 84.

   Pro první tři VLANy budou existovat IPv4 sítě `10.xx.0.0/16`, kde místo `xx` dosaďte číslo VLAN.

2. Mezi jedním ze switchů a routerem zprovozněte Router-on-a-Stick, a to pouze pro VLAN 10, 20 a 84, ze strany switche bude nativní VLAN 90. Na subrozhraních dejte vždy první IP adresu z rozsahu sítě pro danou VLAN, to bude brána pro příslušnou síť. Nezapomeňte příslušné fyzické rozhraní zapnout. Ověřte si, zda mezi sítěmi správně funguje směrování:

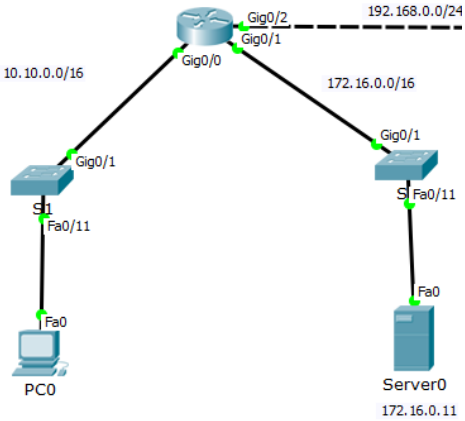
   ```
   show ip route | begin Gateway
   ```

3. Na routeru zprovozněte DHCPv4 pro sítě příslušející VLAN 10 a 20. Vždy prvních 5 adres z každého rozsahu vyjměte z přidělování. Kromě povinných parametrů uveďte alespoň název domény. V Packet Traceru bohužel nefunguje parametr `lease`.
4. Ke switchům připojte počítače, zvolené porty budou v přístupovém módu (access). Nasaďte na těchto portech čísla VLAN tak, aby do prvních tří VLAN patřil vždy alespoň jeden počítač. Na počítači spadajícím do VLAN 84 (Management) nasaďte staticky adresu `10.84.0.6`, maska bude `255.255.0.0`.
5. Na jednom ze switchů nastavte na rozhraní VLAN84 IP adresu `10.84.0.2`, masku `255.255.0.0`, vytvořte nového uživatele (zvolte vhodně přihlašovací jméno a heslo), nastavte vhodné heslo do enabled módu a na VTY linkách zprovozněte SSH, nastavte na nich přihlašování přes lokální databázi (tj. s využitím vytvořeného uživatele).
6. Alespoň na jednom switchi vypněte všechny nepoužité porty a nastavte je do accessového módu s uvedenou VLAN 95 (Parking). Použijte k tomu `interface range`.
7. Vyzkoušejte, zda funguje `ping` mezi jednotlivými počítači. Také vyzkoušejte, jestli z počítače patřícího do VLAN 84 funguje přístup na switch s nakonfigurovaným SSH.

Úkol

Pro tento úkol budete potřebovat předpřipravený soubor `DHCP_NAT.pkt`.

Na routeru `R1` nakonfigurujte pool pro rozsah adres `10.10.0.0/16` podle výše uvedeného návodu, z přidělování vyjměte prvních 10 adres a pak poslední adresu z rozsahu (stačí uvedený rozsah adres a rozsah vyjmutých adres).



Pak se podívejte, jestli už počítač PC0 už má adresu a bránu, a kterou adresu vlastně získal. Vyzkoušejte, jestli půjde z PC0 pingnout server, případně použijte webový prohlížeč na PC0.

Dále na routeru zkontrolujte stav přidělení adres (tj. dynamickou tabulku přidělených adres), případně vyzkoušejte další `show` příkazy.

Soubor uložte, využijete ho dál v úkolu v následující sekci.

Otázky

Proč myslíte, že není nutné definovaný pool nasazovat na rozhraní? Pokud by bylo definováno více poolů, podle čeho pozná router, ze kterého má žadateli adresu přidělit?

3.1.2 DHCP Helper

Pokud máme ve firmě více routerů s různými soukromými sítěmi a pouze jeden z nich bude plnit roli DHCP serveru, pak ostatní routery musejí plnit roli tzv. relay agenta. To je přeposílač, který přepošle broadcast daného typu do jiné sítě (to by jinak nešlo, broadcasty končí na hranici sítě). Pokud chceme, aby router plnil roli relay agenta, spustíme na něm tzv. helper. Helpery mohou být pro různé protokoly/služby, záleží, co vše na zařízení s určitou adresou běží.

Na budoucím relay agentovi, konkrétně rozhraní, za kterým se nacházejí žadatelé o adresu, nastavíme helper odkazující na adresu serveru poskytujícího danou službu (dáme tam tu adresu, pod kterou náš „agent“ vidí poskytovatele služby).

```
int g0/xxx
    ip helper-address 192.168.0.1
```

Nastavení helperu se dá ověřit třeba v running-configu nebo ve výpisu parametrů rozhraní:

```
sh ip int g0/...
```

Úkol

Pokračujte v příkladu z předchozího úkolu. Na routeru `R1` vytvořte druhý pool pro adresy `10.20.0.0/16`, opět prvních 8 adres vyjměte. Na routeru `R2` pak spusťte helper umožňující zařízením za rozhraním `g0/0` získat od routeru `R1` adresu. Vyzkoušejte, jestli jde z PC1 pingnout jak server, tak i PC0.

3.1.3 Možnosti získání IPv6 adresy

IPv6 adresu lze získat více různými způsoby: SLAAC, SLAAC s bezstavovým DHCPv6 nebo plně dynamicky přes stavový DHCPv6, případně staticky.

Nejjednodušší pro koncová zařízení je SLAAC (Stateless Autoconfiguration), která je na Cisco routerech defaultní, jen je třeba zprovoznit IPv6:

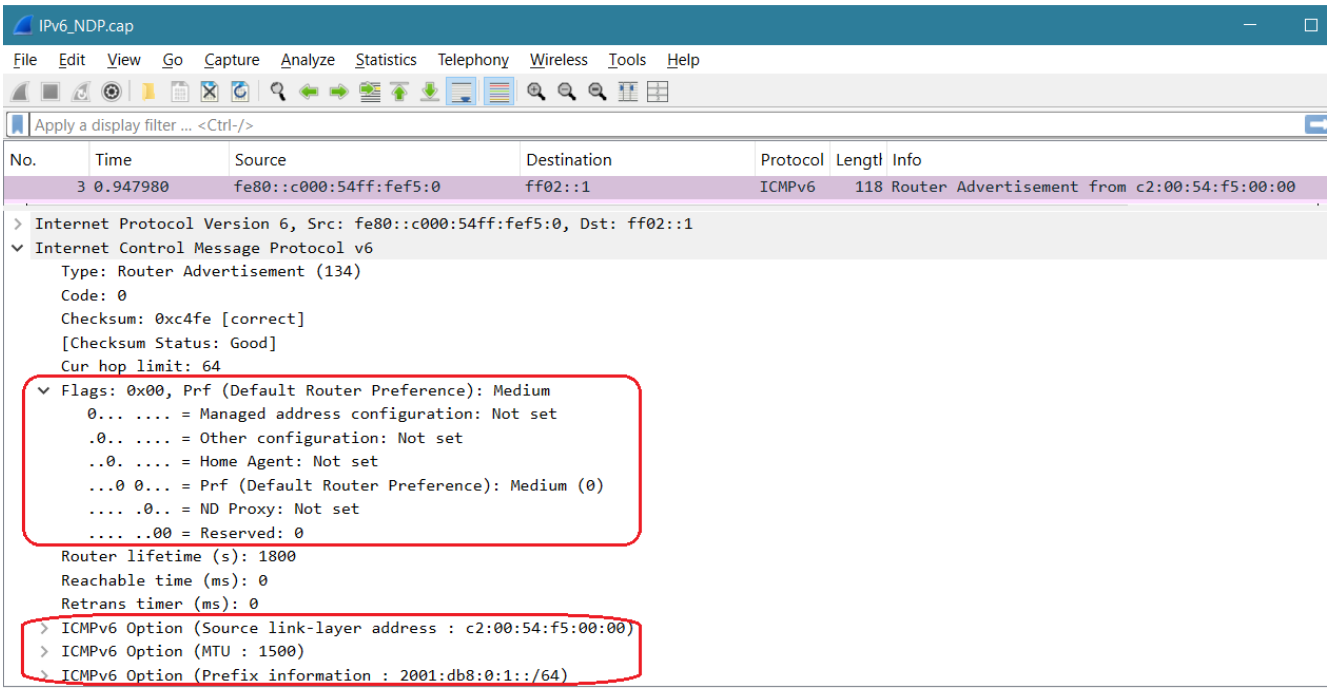
```
ipv6 unicast-routing
```

A samozřejmě je nutné přidělit IPv6 adresu na daném rozhraní routeru. Samotný router posílá RA zprávu (Router Advertisement), ze které se klienti dozví adresu sítě, adresu brány a případně další parametry. Hostitelskou část adresy si pak dopočtou sami.

Další možnosti jsou stavový a bezstavový DHCPv6. Stavový funguje stejně jako v IPv4 (přiděluje i adresy a vede si tabulku se stavem přidělení adres), ktežto bezstavový je „na půl cesty“: adresu (i bránu) si klient vyřídí pomocí routeru, ale ostatní informace dostane od DHCP serveru (například adresu DNS serveru, název domény apod.). Ten může běžet na tomtéž routeru.

Klient je o konkrétním nastavení sítě informován v RA zprávě od routeru, kde jsou dva důležité příznaky: M (managed flag) a O (other config flag).

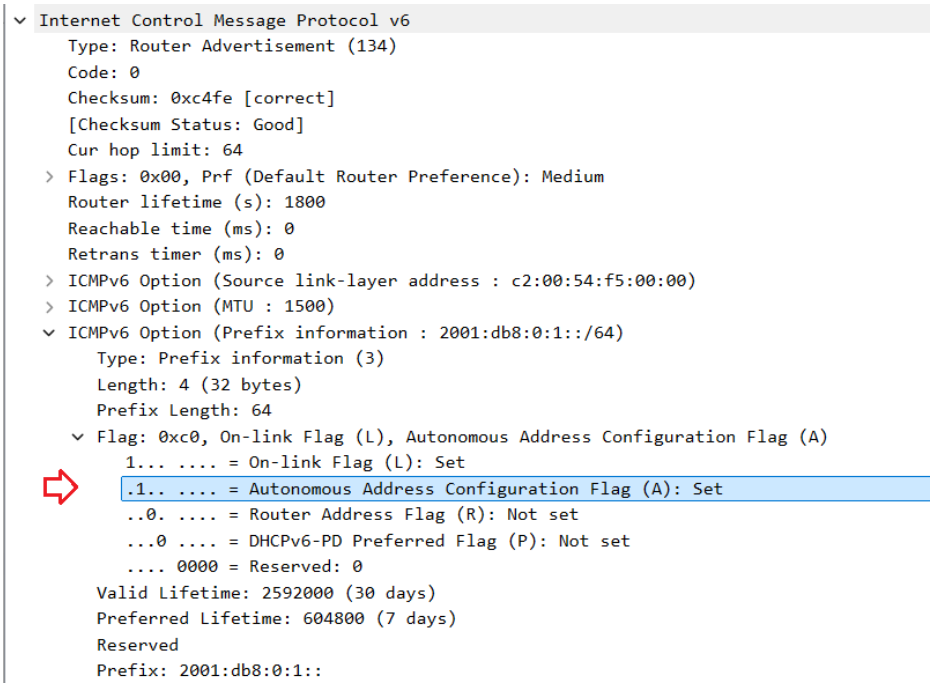
Na obrázku 3.1 je screenshot z Wiresharku, soubor `ipv6_NDP.pcap`, kde vidíme ICMPv6 RA zprávu oznamující všem zařízením v síti, jaký je prefix (adresa sítě) a další informace. Všimněte si údajů v poli Flags: není nastaven ani managed flag, ani other config flag. V následujících polích najdeme jak link-local



Obrázek 3.1: Umístění příznaků pro způsoby získání IPv6 adresy v RA zprávě

adresu routeru (která bude sloužit jako brána), tak i prefix sítě včetně jeho délky. Z toho vyplývá, že v síti se používá SLAAC a není tam žádný DHCPv6 server.

Pak je tu ještě jeden příznak: A (Autonomous Address Configuration Flag). V RA zprávě se hledá hůře, je zanořený ve volitelné části ICMPv6 Option (Prefix information...). Jeho umístění vidíme na obrázku 3.2, je označený šipkou. Zrovna na tom obrázku jsou příznaky M a O nastaveny na 0, takže by mělo jít o SLAAC.

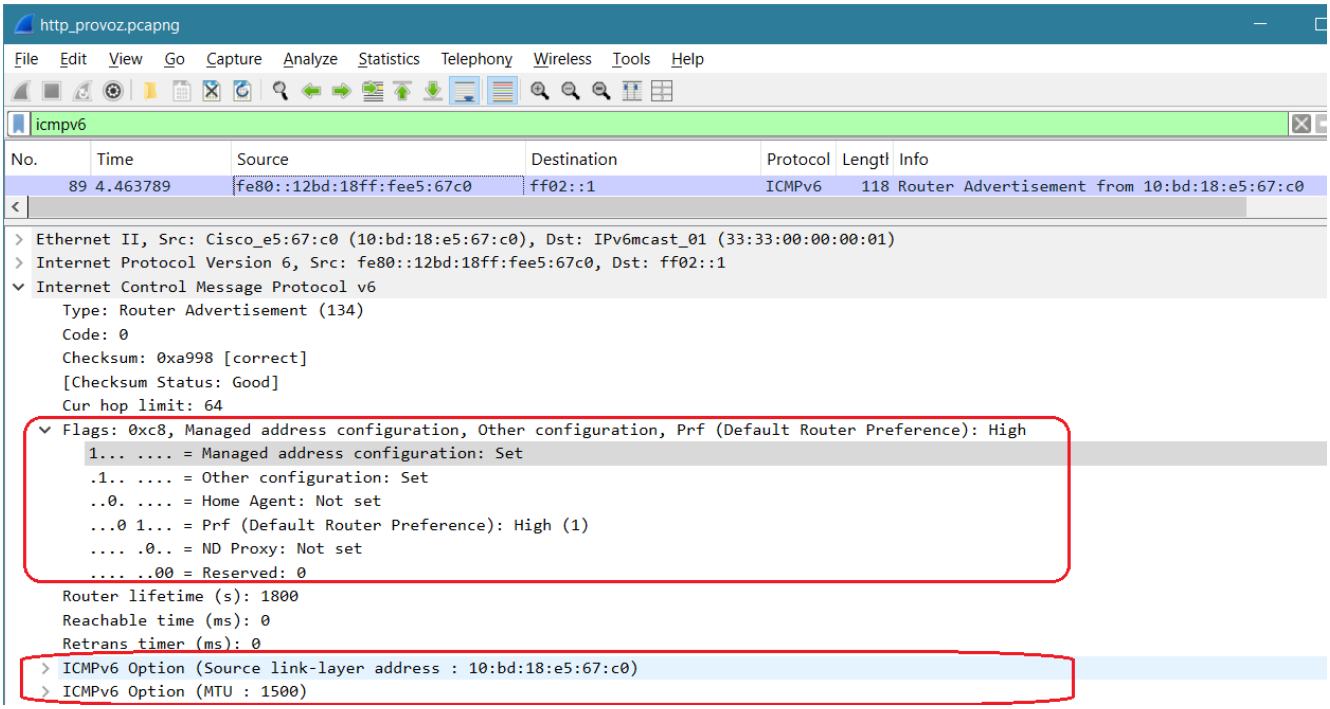


Obrázek 3.2: Příznak A (Autonomous Address Config) – zařízení si má samo dopočítat IPv6 adresu

Tabulka 3.1: Příznaky (flagy) určení způsobu získání IPv6 adresy pro klienty v síti

	SLAAC	Stateless DHCPv6	Stateful DHCPv6
Nastavení flagů:	M = 0, O = 0, A = 1	M = 0, O = 1, A = 1	M = 1

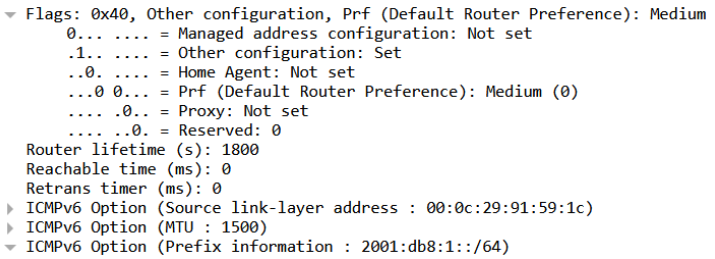
Obrázek 3.3 ukazuje ICMPv6 RA zprávu v síti se stavovým DHCP. Managed flag je nastaven na 1 a router ohlašuje pouze svou link-local adresu (a MTU).



Obrázek 3.3: Příznaky pro stateful DHCPv6 v RA zprávě

Na obrázku 3.4 je ukázka nastavení příznaků Managed a OtherConfig v RA zprávě pro bezstavový DHCPv6. Kromě link-local adresy brány zde najdeme také údaj o prefixu sítě, ostatně stejně jako na obrázku 3.2 pro bezstavový DHCPv6.

Příklad stateless DHCPv6 najdeme například na https://www.cloudshark.org/captures/2831e78f8b44.



Obrázek 3.4: Příznaky pro stateless DHCPv6

Úkol

Pokračujte v úkolu ze strany 3.1.2 o DHCP Helperu. Zprovozněte IPv6 na routeru `R1`, nezapomeňte zapnout IPv6 směrování, přidělte na rozhraní `g0/0` tyto adresy:

- `2001:db8:acad:10::1/64` jako globální unicast
- `fe80::1` jako link-local adresu

Na rozhraní `g0/1`:

- `2001:db8:acad:172::1/64` jako globální unicast
- `fe80::1` jako link-local adresu

Ověřte, jakou adresu si PC0 dopočítalo. Vyzkoušejte příkaz `show ipv6 int g0/0` a projděte si výstup až do konce.

3.1.4 Stateless DHCPv6

Na routeru je třeba kromě toho, co bylo uvedeno u SLAAC, provést dvě další nastavení:

- vytvořit pool, ve kterém však nebude přidělitelný rozsah adres, ale jen ty další parametry, o které budou klienti žádat (DNS server apod.),
- na daném rozhraní nasadit pool a nastavit other config flag.

Příklad

V globálním konfiguračním módu routeru vytvoříme DHCPv6 pool, ale do něj zadáme jen to, co bude klientovi sdělováni „navíc“ oproti RA zprávě:

```
ipv6 dhcp pool BEZSTAVOVY1
    dns-server xxxx
    domain-name xxxx
    exit
```

Pak půjdeme na rozhraní, za kterým je daná síť (případně subrozhraní) a kromě nastavení IPv6 adresy (a dalšího, např. link-local adresa) určíme, že na tomto rozhraní bude odpovídat DHCP server, a nastavíme „O“ flag:

```
int g0/1
    ipv6 addr xxxx yyyy
    ....
    ipv6 dhcp server BEZSTAVOVY1
    ipv6 nd other-config-flag
```

Úkol

Pokračujte v některém z předchozích příkladů, kde jste nastavovali DHCPv4 server. Zprovozněte IPv6 adresaci a bezstavový DHCPv6.

3.1.5 Stateful DHCPv6

Stavový DHCPv6 přiděluje klientům přímo adresy (zároveň s informováním o dalších parametrech) a vede si je v tabulce přidělených adres.

Oproti stavovému DHCPv4 je zde několik rozdílů, z nichž jeden je trochu problematický: pro IPv6 neexistuje obdoba příkazu `ip dhcp excluded addr`. Tedy musíme jiným způsobem dát na vědomí, které

adresy nesmí být takto přiděleny klientům. Obvykle se vhodně upraví adresní rozsah, aby do poolu nespadaly staticky přidělované adresy.

Příklad

Celá podsíť má adresu `2001:db8:abab:12cd::/64`, ale určitý počet adres ze začátku rozsahu nechceme dynamicky přidělovat. Takže adresu do poolu trochu upravíme, třeba na `2001:db8:abab:12cd:1000::/68`, takže budou přidělovány adresy až od této dále.

Postupujeme podobně jako v předchozím případě, jen při vytváření poolu definujeme také rozsah IP adres a na rozhraní nastavíme managed config flag:

```
ipv6 dhcp pool STAVOVYDHCP
    address prefix 2001:db8:abab:12cd:1000::/68 lifetime infinite
    dns-server 2001:db8:abab:12cd::4
    domain-name firma.cz
    exit
int g0/1
    ipv6 addr 2001:db8:abab:12cd::1/64
    ipv6 dhcp server STAVOVYDHCP
    ipv6 nd managed-config-flag
```

Všimněte si, jakým způsobem se stanovuje rozsah přidělovaných adres – trochu jiné je klíčové slovo, a také doba pronájmu adresy je součástí určení rozsahu.

Při použití stavového DHCPv6 jsou na DHCP serveru jednotlivá zařízení evidovaná ve stavové tabulce tak jako u DHCPv4. Protože jedno zařízení může mít přiděleno více IPv6 adres (na různých rozhraních, ale také na jednom rozhraní), je zařízení v tabulce jednoznačně identifikováno číslem *DUID* (DHCP Unique ID).

DUID můžeme vidět ve Windows ve výstupu `ipconfig`, v Linuxu a jiných UNIXech je v konfiguračních souborech. Informace najdeme v RFC 8415.

3.1.6 Zóny a LLA adresy

Firemní síť mohu rozdělit do několika zón: příslušný standard je RFC 4007 (IPv6 Scoped Address Architecture). Lokálně platné adresy pak mohou mít omezenou platnost jen na danou zónu, zóny stejné úrovně se nesmí překrývat. Každé rozhraní vede do nejvýše jedné zóny.

Existuje celá řada úrovní (scope), což zjistíme na přednášce o multicastových adresách, jsou v hierarchii: podřízená zóna je vždy plně vnořena do nadřízené, nemůže se rozprostírat do více nadřízených zón (v standardu se hovoří o konvexitě).

Příklad

Na zařízeních se příslušnost do zóny projeví symbolem `%` a identifikátorem zóny za LLA adresou, například ve Windows (část výpisu příkazu `ipconfig`):

```
Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . : fpf.slu.cz
   IPv6 Address. . . . . . . . . . . : 2001:718:2601:265:5a1c:6b7:a869:e116
   IPv6 Address. . . . . . . . . . . : 2001:718:2601:265:ffff:4c33:327c:6f4a
```

```
   Temporary IPv6 Address. . . . . : 2001:718:2601:265:3de6:6c19:2c27:c16a
   Link-local IPv6 Address . . . . . : fe80::ad34:57c1:fb03:bcb7%5
   IPv4 Address. . . . . . . . . . . : 10.6.13.204
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : fe80::208:e3ff:feff:fc08%5
                                       10.6.13.1
....
Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . : wifi.slu.cz
   Link-local IPv6 Address . . . . . : fe80::d349:5fe5:dce3:93d2%5
....
```

V našem případě zjevně rozhraní Ethernet i Wi-fi míří do téže zóny (což taky není problém a je to celkem běžné).

Identifikátor zóny (číslo nebo ID rozhraní) má lokální platnost, na každém zařízení zóny můžeme mít jiný. Forma identifikátoru také záleží na operačním systému. V příkladu jde o číslo, ale může jít o řetězec.

Jako parametr příkazu (třeba ping) můžeme použít jak samotnou LLA adresu, tak i „obohacenou“ o identifikátor zóny:

```
ping xxxxxx%zzz
```

3.2 Statické směrování a směrovací tabulka

Při použití statického směrování vždy na konkrétním routeru ručně nastavíme cestu k cílové síti (v globálním konfiguračním módu). Směr se určuje buď zadáním síťového rozhraní, za kterým leží cíl, nebo uvedením IP adresy souseda, přes kterého je cíl dosažitelný.

 Pro IPv4 je forma následující:

```
ip route cíl maska směr
```

Například pokud je cílem síť `192.168.0.0/16` a dá se do ní dostat přes rozhraní `g0/2` a souseda `10.142.0.1`, můžeme použít jeden z těchto příkazů:

```
ip route 192.168.0.0 255.255.0.0 g0/2
ip route 192.168.0.0 255.255.0.0 10.142.0.1
```

Pokud existuje třeba jen malá pravděpodobnost, že cílová síť se může někdy ocitnout za jiným rozhraním než `g0/2` (což se může stát, pokud v síti existuje více cest k cíli a některá z nich může selhat nebo například být zahlcená), pak je lepší druhá možnost.

 Pro IPv6 je to podobné, jen se určitě více používá určení cesty přes souseda. Na některých zařízeních dokonce ani nemusí být dostupná možnost určení cesty přes rozhraní.

```
ipv6 route 2001:db8:acad:a::/64 2001:aaaa:7007:ba25::4
```

 Staticky se často konfiguruje *default route*, tedy cesta z našich sítí do sítě poskytovatele internetu, zde určíme odchozí rozhraní:

```
ip route 0.0.0.0 0.0.0.0 g0/2
```

Proč jsou zde samé nuly (IP adresa i maska)? Maska v procesu směrování určuje, které bity cílové adresy ze směrovaného paketu musí být ve shodě s cílovou adresou uvedenou na příslušném řádku tabulky. Pokud je celá maska nulová, pak to znamená, že se nekontroluje shoda vlastně vůbec, nulová IP adresa je v takovém případě k masce jen „do počtu“ (když se nebude porovnávat, tak proč se s ní mořit...). To je přesně to, co chceme od brány – pokud cílová adresa v paketu nesouhlasí s žádnou „konkrétní“ adresou v tabulce, měla by spadnout právě do tohoto záznamu.

Poznámka

Při práci se směrovací tabulkou existuje jedno základní pravidlo: čím delší prefix, tím lepší. Pokud se najde shoda cílové adresy s více řádky tabulky, zvolí se ten, kde je u sítě uveden delší prefix.

Příklad

Router přijal paket s cílovou adresu `10.84.132.25`. Ve směrovací tabulce našel shodu na těchto řádcích:

```
10.84.128.0/17   ...
10.84.0.0/16     ...
0.0.0.0/0        ...
```

Chybějící sloupce by obsahovaly adresu souseda, přes kterého vede cesta k cílové síti (nebo označení rozhraní), metriku a další informace.

Všechny tři řádky odpovídají, bity spadající do uvedené délky prefixu souhlasí. Poslední uvedený řádek je zjevně default route. Ale protože jsou k dispozici odpovídající řádky s delším prefixem, nezvolí se. Rozhodně bude zvolen první řádek, protože se shoduje v nejvíce bitech (17) s cílovou adresou.

Default route pro IPv6 je taky plně nulová, a protože zkracujeme, bude příkaz vypadat takto:

```
ipv6 route ::/0 adresa_souseda
```

Co když máme dva spoje vedoucí ven (ať už k témuž ISP nebo ke dvěma různým ISP), přičemž jeden spoj chceme využívat defaultně a druhý je záložní? Tomu se říká *Floating static route*. Pro oba spoje vytvoříme statickou cestu, ale tomu záložnímu nastavíme vyšší číslo Administrative distance (AD, vyšší než 1, pokud jsou oba staticky vytvořené). Takže pokud by záložní cesta na internet vedla přes rozhraní g0/3, bude to:

```
ip route 0.0.0.0 0.0.0.0 g0/3 10
```

Vytvořili jsme statickou cestu s AD=10 (statické cesty mívají defaultní AD=1). Ve směrovací tabulce bude pouze ta první cesta s výchozí hodnotou AD (záložní najdeme jen v running-configu), ale pokud přestane fungovat, automaticky se nahraní záložní cestou. Zvolené AD by mělo být nižší než jaká je administrativní vzdálenost pro cesty určené směrovacími protokoly.

Floating static route pro IPv6 se určuje podobně.

3.3 ACL – seznamy řízení přístupu

ACL (Access Control List, seznam řízení přístupu) je mechanismus filtrování na vrstvě L3, částečně L4. Standardní ACL filtruje podle IP adres, extended ACL umí i podle portů a protokolů.

3.3.1 Jak se ACL používají

Každý ACL je identifikován svým číslem, může (nemusí) mít také jméno (to je pak *pojmenovaný ACL*). Takže můžu vytvořit standardní ACL určený číslem nebo jménem, a podobně rozšířený.

Tabulka 3.2: Vyhrazené rozsahy čísel ACL

Čísla	Typ
1–99 nebo 1300–1399	standardní ACL
100–199 nebo 2000–2699	extended ACL (rozšířené)
zbytek	AppleTalk, IPX, extended IPX,... (nepoužíváme)

 ACL nejdřív vytvoříme, a pak nasadíme na rozhraní a směr (tj. filtrujeme na konkrétním rozhraní, a to v příchozím nebo odchozím směru). Speciálním typem ACL je taky ACL na ochranu VTY linek.

ACL je posloupnost pravidel (taktéž očíslovaných), a paket na daném rozhraní a daném směru musí touto posloupností projít. Mohou nastat tyto situace:

- podmínka v pravidle neodpovídá paketu, tedy se jde na následující pravidlo v ACL,
- v pravidle je určeno, že pokud paket splňuje určitou vlastnost (například IP adresa patří do určité sítě), tak může pokračovat v cestě (`permit`),
- v pravidle je určeno, že pokud paket splňuje určitou vlastnost, tak má být zahozen (`deny`),
- paket postupně projde všemi pravidly v ACL a u žádného nedojde ke shodě s podmínkou (tedy vždy nastává první odrážka tohoto seznamu), pak záleží, jaké pravidlo je na konci:
  - `deny any` ⇒ bude zahozen,
  - `permit any` ⇒ projde, může pokračovat.

 ACL pravidla obvykle obsahují IP adresy, k nim potřebujeme dopsat masku. Jenže pozor, Cisco u ACL používá *wildcard masky*, tj. inverzní. Wildcard maska vznikne tak, že v běžné (binární) masce převrátím 0 a 1. Wildcard maska nemusí mít nutně na začátku samé 0 a na konci samé 1, prostě určujeme, že v IP adrese na dané binární pozici buď musí být číslice předepsaná uvedenou adresou, nebo naopak je jedno, jestli v paketu v dané IP adrese je na tom místě 0 nebo 1.

 **Příklad**

Pokud chci do ACL pravidla uvést konkrétní unicast IPv4 adresu (jednu, žádnou síť), běžná maska by byla `255.255.255.255`, ale wildcard maska bude `0.0.0.0`.

```
172.16.0.12 0.0.0.0
```

Pokud chci určit tuto podsíť: `172.16.0.0 255.255.0.0`, bude to:

```
172.16.0.0 0.0.255.255
```

Jestliže v síti `10.0.0.0/8` chci v ACL zachytit právě všechny pakety s IP adresou, kde v druhém oktetu je pevně dané číslo:

```
10.96.0.0 0.0.255.255
```

Případně jsou napevno jen některé bity z toho oktetu:

```
10.96.0.0 0.15.255.255
```

Číslo 15 je binárně `0000 1111`, po invertování dostaneme `1111 0000`, tedy horní polovina daného oktetu musí být podle předpisu. Číslo 96 je binárně `0110 0000`, což znamená, že horní polovina druhého oktetu musí být `0110`, druhá polovina může být jakákoliv.

Otázky

Vezměme adresu s wildcard maskou `10.96.0.0 0.15.255.255` z předchozího příkladu. Pro které adresy z následujících bude takové pravidlo v ACL platit?

```
10.96.52.4
10.105.0.4
10.128.22.1
10.112.150.32
```

A co když to pomícháme? V pravidle bude uvedeno například toto:

```
10.96.32.0 0.0.15.223
```

Co to bude znamenat? Bude tomuto předpisu odpovídat adresa `10.105.32.8`?

3.3.2 Standardní ACL

Standardní ACL určený číslem vytvoříme takto:

```
access-list 10 permit 10.96.0.0 0.0.255.255   ! zdrojová IP
access-list 10 deny host 10.96.100.84         ! místo wildcard masky 0.0.0.0 uvedeme slovo host
access-list 10 deny any                       ! defaultní zakončení, ale hodí se to uvádět
access-list 10 permit any                     ! pokud chceme, aby prošlo vše, co došlo sem
```

Existují tyto možné akce:

- *permit* – paket je povolen,
- *deny* – paket bude zahozen,
- *remark* – chci do ACL přidat poznámku, komentář.

Seznam ACL obsahuje jednu nebo více položek řízení přístupu ACE, každá má své vlastní číslo. Položky ACE číslovaného ACL zadáváme jako samostatné příkazy v globálním configu, a to, že patří k sobě, je určeno číslem ACL za příkazem `access-list`.

Vytvoření pojmenovaného ACL se dvěma položkami ACE (tradičně se název píše velkými písmeny):

```
ip access-list standard NO_ACCESS
    deny host 10.96.100.84
    permit 10.96.0.0 0.0.255.255
    exit
```

Princip je podobný, jen neurčujeme číslo (bude přiděleno automaticky), ale zato určujeme jméno, což je přehlednější. Oproti číslovaným ACL přecházíme do subkonfiguračního módu, ve kterém pracujeme s jednotlivými položkami ACE.

Nasazení na rozhraní:

```
interface f0/0
    ip access-group 10 out   ! nasazeno na odchozí provoz
    ip access-group 20 in    ! nasazeno na příchozí provoz
```

Verifikace:

```
sh access-lists
sh access-list 10
sh ip int f0/0
sh run
```

Poznámka

Vzhledem k tomu, že v standardním ACL je vždy IP adresa zdroje a nic jiného, takové ACL nasazujeme vždy *co nejblíž chráněnému cíli*. Účelem je předejít vedlejším efektům, aby například nedošlo k odmítnutí přístupu do „nadřízené“ sítě.

Úkol

Na webu je předpřipravený soubor `ACL.pkt`. Stáhněte si ho a otevřete. Projděte si všechna rozhraní (včetně loopbacků) a sítě. Ověřte, jestli na routerech opravdu nejsou žádné vytvořené seznamy ACL. Ověřte, jestli funguje směrování (který směrovací protokol je použit?). Ověřte, jestli jsou počítače navzájem dostupné.

Vytvořte standardní číslovaný ACL (č. 1), který povolí provoz ze sítí `192.168.30.0/24` a `192.168.20.0/24` do sítě `192.168.30.0/24`, vše ostatní vedoucí do této sítě zakažte. Nasaďte ACL na vhodném rozhraní. Nezapomeňte, že standardní ACL nasazujeme vždy co nejblíž cíli.

Pomocí příslušných příkazů show zobrazte vytvořený ACL a také jeho nasazení na rozhraní.

Ověřte, jestli ACL opravdu funguje – pro povolené i zakázané sítě. Můžete použít také extended ping.

Dále vytvořte pojmenovaný standardní ACL `ACL-CONFIG`, který povolí provoz ze sítě `192.168.40.0/24` do sítě `192.168.10.0/24`, a dále povolí přístup PC-C (`192.168.30.3`) do této sítě. Nasaďte na vhodném zařízení a rozhraní.

Opět si prohlédněte vytvořený ACL a také místo, kde je vytvořen.

Ověřte funkčnost ACL na spojeních jak povolených, tak i zakázaných. Prohlédněte si také statistiku přes `show access-lists`.

3.3.3 Modifikace existujícího standardního ACL

Jsou dvě možnosti, jak pozměnit existující ACL:

- bokem v textovém editoru si vytvořím celý ACL (zkopíruji z konzole, pozměním), původní odstraním (`no access list ...`, `no ip access-list...`) a během znovuvytváření vkopíruji),
- použiju příkazy pro přidání nových ACE (pravidel).

Pro pojmenovaný ACL použijeme tento postup:

1. Nejdřív ověříme, jak jsou čísla ACE přiřazená a která jsou volná:

```
sh access-lists
```

   Na pozici 30 se rozhodneme přidat nové pravidlo, a pak na pozici 40 budeme chtít ručně přidat zakončení typu `deny any`:

```
ip access-list standard název
    30 permit adresa inv-maska
    40 deny any
    end
```

2. Můžeme taky odstranit položku, a to tím, že ji opíšeme a před ni dáme „`no`“.

ACL zůstal nasazen na rozhraní i po případném smazání či modifikování ACL, takže po modifikaci nemusíme znovu umísťovat na rozhraní.

Pokud se jedná o číslovaný ACL, je to trochu složitější. Sice v příkazu show access-lists taky máme čísla pravidel, ale nelze takto snadno přidávat nová pravidla mezi stávající. Můžeme odstranit pravidlo ze seznamu pomocí „`no`“ notace, nové se však přidá na konec pravidla.

Poznámka

Položka `deny any` je sice defaultně platná, ale pokud ji do seznamu přímo umístíme, můžeme si zobrazit související statistiku.

Úkol

V příkladu proveďte tuto modifikaci: zařízení ze sítě `209.165.200.224/27` mají mít přístup do `192.168.10.0/24`, a dále má být na konec přidána položka deny any. Otestujte, jestli přidané položky fungují jak mají.

3.3.4 ACL pro ochranu VTY linek

Seznam ACL lze nasadit i na VTY linky. Tím určujeme, odkud bude fungovat přístup do administrace zařízení.

ACL se vytváří úplně stejně jak je výše uvedeno, jen nasazení na linky je trochu jinak:

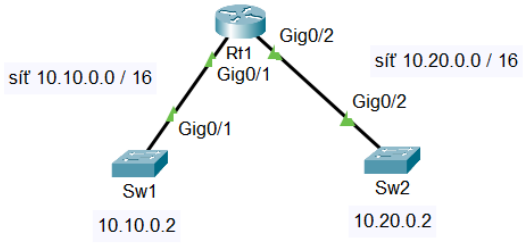
```
line vty 0 4              ! nebo line vty 0 15, podle skutečného počtu linek
 access-class jméno-nebo-číslo-ACL in
```

Příklad

Vytvoříme síť podle obrázku vpravo. Nastavíme IP adresy u switchů na rozhraní vlan1. Na routeru nastavíme na obou rozhraních vždy první IP adresu z rozsahu pro danou síť.

Na všech třech zařízeních vytvoříme uživatele admin, zvolíme vhodné heslo. Heslo tak, aby bylo na zařízení šifrované. Na VTY linkách nastavíme autentizaci podle lokální databáze (pozor na počet linek na jednotlivých zařízeních), povolíme přístup jen přes SSH (proto musíme také nastavit doménu, verzi SSHv2 a vygenerovat klíč).

Ukázka pro switch Sw1 (začínáme v globálním konfigu):




```
hostname Sw1
username admin secret silneHeslo
enable secret dalsiSilneHeslo
ip domain-name slu.cz
ip ssh version 2
crypto key mypubkey rsa
                          ! pak zadáme délku klíče 1024
line vty 0 15
    login local
    transport input ssh
int vlan1
    ip addr 10.10.0.2 255.255.0.0
    no shut
    exit
ip default-gateway 10.10.0.1
```

Na switchi Sw2 to bude podobné, jen s adresami to bude trochu jinak. Na routeru samozřejmě nasadíme adresu na příslušném fyzickém síťovém rozhraní, nikoliv VLAN1, a tam taky nebude brána.

Na Sw2 vytvoříme standardní pojmenovaný ACL `VTYLINES`, ve kterém povolíme síť `10.20.0.0/16`, dále povolíme přístup z jedné adresy `10.10.0.2` (jedna adresa = „host“), a vše ostatní zakážeme.

Tento ACL nasadíme na VTY linky. Jak vidíte, ACL nemusí být jen na zařízeních vrstvy L3, ostatně i zde máme IP adresu, třebaže na virtuálním rozhraní.

```
ip access-list standard VTYLINES
    permit 10.20.0.0 0.0.255.255
    permit host 10.10.0.2
    deny any
    exit
line vty 0 15
    access-class VTYLINES in
```

3.3.5 Extended ACL

Rozšířené seznamy řízení přístupu dokážou filtrovat podle více kritérií než jen zdrojové IP adresy: zdrojová a cílová IP, protokol, port. Jinými slovy: filtruje se podle různých informací v L3 a L4 záhlaví.

Na rozdíl od standardních ACL je umísťujeme *co nejblíž zdroji*, aby zakazovaný provoz co nejméně zatěžoval síť.

 Vytvoření ACE číslovaného rozšířeného ACL:

```
access-list 105 permit protokol zdroj-ip              cil-ip eq cil-port
access-list 105 permit protokol zdroj-ip eq zdroj-port cil-ip eq cil-port
access-list 105 permit protokol zdroj-ip              cil-ip established
```

Měla by tedy být uvedena zdrojová a cílová IP adresa, u každé může být za klíčovým slovem `eq` přidaný port.

Místo adresy můžeme uvést `any`, pokud tento parametr nemá být rozlišován. Možnosti pro protokol zjistíme otazníkem (tcp, udp, icmp, smtp, ftp, ftp-data,...). Porty mohou být zadány číslem nebo jménem (např. http je číslo 80, https 443 – možnosti opět zjistíme otazníkem).

Na konci se předpokládá `deny ip any any`, nicméně může být praktické uvést explicitně.

Vytvoření pojmenovaného rozšířeného ACL:

```
ip access-list extended jmeno
 permit tcp...
 exit
```

Pokud na konec ACL připíšeme klíčové slovo `established`, vyhodnotí se shoda jen tehdy, pokud má příslušný TCP segment nastavený bit ACK nebo RST, které nebývají u prvního TCP segmentu v handshaku (u protokolu UDP nemá smysl, tam se handshake nekoná). Takže pokud chceme ztížit hackerům pokusy o navázání spojení do naší části sítě, ve které nemáme žádné servery, můžeme nastavit:

```
permit tcp any adresa.chráněné.sítě eq 80 established
```

a zbytek provozu zakážeme, tento ACL nastavíme na vstupním rozhraní z „nebezpečné“ sítě nebo výstupním do „chráněné“ sítě. Důsledkem je, že zde konkrétně lze navázat spojení na webové servery z chráněné sítě ven, ale nikoliv dovnitř.

Jako první ACE do sítě bez serverů bývá velmi často tato položka:

```
permit tcp any any established
```

Což znamená, že provoz, který byl navázán z vnitřní sítě ven, projde rychle a bez problémů, a až další bude podrobněji filtrováno. Pozor na UDP provoz, pokud bychom ho nepovolili některým dalším pravidlem, nefungoval by mnohý multimediální provoz. Defaultní „zákaz ostatního“ může být explicitně uveden takto:

```
deny ip any any
```

Znamená to, že veškerý IP provoz (tedy to, v čem jsou zapouzdřeny všechny TCP a UDP segmenty) bude zakázán.

Otázky

Zamyslete se: pokud zadáme `deny ip any any`, co projde?

Úkol

Na webu si najděte předpřipravený příklad `extendedACL.pkt`. Jsou tam tři routery, dva switche, jeden server a dva počítače. Projděte si jejich nastavení, zjistěte, kde je zprovozněn přístup přes SSH.

Nastavte v síti ACL takto:

- Půjde o pojmenovaný extended ACL s názvem `WEB_ACCESS`.
- Počítač s adresou `10.10.0.11` může přes SSH přistupovat k routeru R2.
- Webový provoz ze sítě `10.10.0.0/16` může kamkoliv.

Vyzkoušejte na počítači `10.10.0.11` ping na druhý počítač (`10.20.0.11`). Bude to fungovat? Dále ve webovém prohlížeči na tomtéž počítači zobrazte web `192.168.0.11`. Funguje? Vyzkoušejte na tomtéž počítači přístup přes SSH k routerům R1 a R2. Který z nich funguje? Jsou to tyto příkazy (pokud jste v Packet Traceru):

```
ssh -l admin 10.40.0.2
ssh -l admin 10.10.0.1
```

Na PC s Windows, Linuxem nebo MacOS je syntaxe trochu jiná:

```
ssh admin@10.40.0.2
ssh admin@10.10.0.1
```

Na routeru ISP vytvořte loopback lo0 s IP adresou `209.165.200.225/27` (tj. maska `255.255.255.224`). To bude simulace našeho poskytovatele internetu. Zakažte veškerý telnet provoz ve směru z internetu

3.3.6 Filtrace ICMP provozu pomocí Extended ACL

Pro protokol ICMP existuje několik „portů“ (zde samozřejmě o porty nejde, ale spíše o typy ICMP):

- unreachable
- time-exceeded
- echo
- echo-reply
- parameter-problem
- source-quench

Poslední dva uvedené typy jsou důležité například při zjišťování MTU na cestě.

Pokud tedy chceme povolit všechny kromě `echo`, můžeme je buď vyjmenovat v jednotlivých ACE, nebo to může být třeba takto:

```
deny icmp any any echo
permit icmp any any
```

a další, co chceme povolit či zakázat. Mezi zařízeními mohou být v tomto seznamu rozdíly, také Packet Tracer má méně možností.

Pokud máme obavy z DoS útoku zneužitím ICMP echo, můžeme místo zákazu nastavit limit (nejde v Packet Traceru). Nastavuje se vždy na rozhraní (na každém rozhraní to můžeme mít jinak, jak co se týče ACL, tak i tohoto limitu). Je to příkaz `rate-limit`, takže:

```
interface g0/1
    rate-limit input access-group ozn_ACL bps bpsnormal bpsmax conform-action transmit exceed-action drop
```

To znamená, že ACL nasadím na rozhraní s tím, že pokud je provoz uvedený v ACL do limitu, tak projde, ale pokud už je nad limit, bude zahozen. Za označením ACL jsou tři čísla (mohou být stejná) – je to provoz v bitech za sekundu. Podrobnosti najdete na následujícím odkazu.

Další informace

- https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/qos_classn/configuration/15-mt/qos-classn-15-mt-book/qos-classn-car.html
- https://www.networkcomputing.com/network-security/protecting-cisco-router-ping-flooding

Příklad

Pokračujte v zadání z předchozího úkolu.

Na routeru R2 vytvořte ACL (číslovaný s číslem 102), který bude filtrovat provoz přicházející od ISP, a to takto:

- povolte veškerý TCP provoz, který byl navázán z vnitřních sítí,

- povolte ICMP echo reply a další ICMP provoz zjišťující chyby (vyjmenujte v jednotlivých ACE),
- zakažte veškerý ostatní IP provoz.

Otestujte ping vedoucí z a do chráněné sítě, pak si vypište statistiky.

3.3.7 Pár doporučení k ACL

Nejdřív obecně k ACL: různá zařízení obvykle podporují pojmenované i číslované seznamy, ale nemusí to být pravidlem. Výjimečně můžeme narazit na zařízení (ani ne tak router, jako spíše hardwarový firewall), kde něco z toho „nejde“.

 V příkladech (nejen zde) vidíme obvykle IP adresy, ale ve skutečnosti mohou být použity i jmenné adresy. Například:

```
access-list 104 permit tcp any host web.domena.cz eq www
```

Zařízení, na kterém něco takového použijeme, však musí být schopno toto jméno přeložit.

 V ACL existují určitá doporučení ohledně řazení položek ACE:

- bereme v úvahu, že se ACE v seznamu zpracovávají shora dolů, tedy nejdřív mají být specifičtější položky a až potom obecnější,
- obvykle bývají nejdřív položky typu `permit`, ale záleží na podmnožinách/nadmnožinách sítí: můžeme třeba chtít, aby pro některá zařízení dané sítě platilo určité nastavení, ale pro zbytek sítě jiné nastavení, což se v pořadí položek musí odrazit (bez ohledu na to, co je `permit` a co `deny`),
- pokud se to „nebije“ s předchozím bodem, umísťujeme dřív to, co bude častěji používáno.

Co s druhým bodem, jak zjistíme typickou frekvenci používání konkrétních ACE? Jednoduše tak, že položky ACE nejdřív nasadíme podle odhadu, ale po určité době používání si zobrazíme statistiku pomocí show access-list. U každé ACE bude uveden počet „zásahů“, podle toho přeuspořádáme.

Proč je vlastně druhý bod důležitý? Protože tak zlepšujeme průchodnost (propustnost) filtrování, a taky šetříme výpočetní zdroje daného zařízení.

Poznámka

Je jedno úplně nejdůležitější pravidlo: přehlednost. Pokud aplikace výše uvedených pravidel snižuje přehlednost, u seznamu s několika položkami to nevadí, ale u seznamu s desítkami či stovkami položek to už může vadit velmi podstatně.

 Některá zařízení umožňují vytvářet *skupiny* (sítí, rozsahů portů, protokolů), obvykle jen u extended ACLs a jen pro IPv4 adresy. Pokud zjistíme, že na daném zařízení to lze (tedy je podporován příkaz `object-group` v globálním konfiguračním módu), může nám to hodně pomoci při zvýšení přehlednosti ACE v seznamu. Má to své výhody (zpřehlednění ACL), ale i nevýhody: navyšuje se výpočetní náročnost práce se seznamem a zabírá to více paměti v NVRAM.

Další informace

Práce se skupinami v ACL: https://www.cisco.com/en/US/docs/ios-xml/ios/sec_data_acl/configuration/15-2mt/sec-object-group-acl.html

Dalším „vylepšením“ jsou *Turbo ACL* (čte se údajně jako turbo-ackle). Jsou to kompilované ACL, tedy přeložené do strojového kódu (běžné ACL jsou vlastně obdobou interpretovaných programů v textové podobě), čímž se urychluje práce s nimi. Zapnutí Turbo ACL znamená, že všechny definované ACL budou zkompilovány, přičemž se dá i ovlivnit množství paměti pro ně vyhrazené.

```
access-list compiled
show access-lists compiled
```

3.4 Vnitřní dynamické směrování

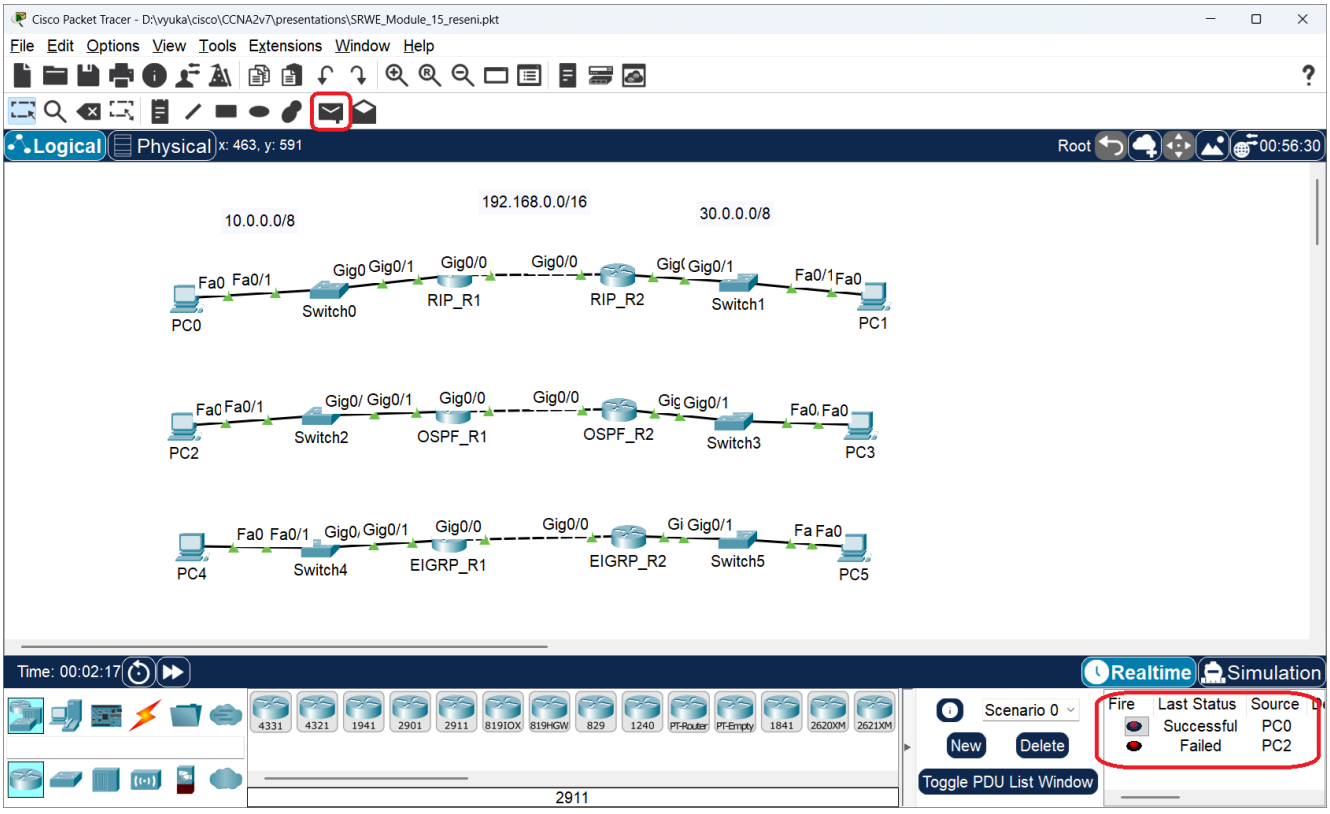
Protokoly určené pro vnitřní směrování (interior routing) jsou určeny pro směrování mezi sítěmi patřícími téže organizaci (nikoliv na páteřní sítě).

3.4.1 Úvod do dynamického směrování

Při dynamickém směrování necháváme určení cesty a její zhodnocení do podoby metriky na zvoleném směrovacím protokolu. Takových protokolů je víc, nicméně většinou se setkáváme s protokolem OSPF.

Úkol

Otevřete si předpřipravený soubor `dynamicke-smerovani.pkt`. Najdete v něm tři řady předkonfigurovaných zařízení, v každé řadě je pro směrování použit jiný protokol.



Obrázek 3.5: Ikona pro poslání jednoduchého pingu v Packet Traceru

Ověřte si, zda opravdu směrování funguje jak má: nahoře na panelu nástrojů klepněte na „zavřený dopis“ (simple PDU), pak klepněte na počítač vlevo v dané řadě, a pak na počítač vpravo v téže řadě. Vpravo dole se objeví výsledek. Pokud tam najdete „Failed“, je drobný problém v čase (zařízení se ještě nestihla domluvit na tom, kde kdo je). Pak stačí poklepat na červenou ikonu před slovem Failed a paket bude poslán znovu, pravděpodobně už úspěšně (Successful).

Poklepejte postupně na některý router v každé řadě a v enable módu si vypište směrovací tabulku, například pro první router v první řadě by měla vypadat takto:

```
R1# sh ip route | begin Gateway
Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C       10.0.0.0/8 is directly connected, GigabitEthernet0/1
L       10.0.0.1/32 is directly connected, GigabitEthernet0/1
R    30.0.0.0/8 [120/1] via 192.168.0.2, 00:00:25, GigabitEthernet0/0
C    192.168.0.0/16 is directly connected, GigabitEthernet0/0
     192.168.0.0/32 is subnetted, 1 subnets
L       192.168.0.1/32 is directly connected, GigabitEthernet0/0
```

Řádky začínající písmenem `C` představují connected (připojené) sítě, řádky začínající `L` jsou k nim příslušející lokální rozhraní (adresy nasazené na rozhraních mířících do sítí `C`). Nás zajímá to zbývající. Zde konkrétně je použit protokol RIP, tedy řádek začíná písmenem `R`. V dalších dvou řadách to bude řádek začínající `O` (protokol OSPF) nebo `D` (protokol EIGRP).

V hranatých závorkách za adresou máme nejdřív číslo AD pro směrovací protokol (zde pro RIP je 120), za lomítkem je metrika pro danou cestu. Pro číslo AD platí, že čím lepší protokol, tím menší číslo. Jak si stojí tyto tři protokoly?

Srovnejte metriku těchto tří protokolů, všimněte si rozdílu především u EIGRP. Číslo metriky určené různými protokoly nelze ve skutečnosti porovnávat, protože každý protokol má vlastní algoritmus pro výpočet metriky a každý do výpočtu zahrnuje jiné údaje. Porovnávat lze metriky pouze u cest zjištěných tímtéž protokolem.

K jednotlivým protokolům lze zjistit podrobnější informace pomocí příkazů

```
sh ip rip ....
sh ip ospf ....
sh ip eigrp ...
```

Místo teček se doplní klíčové slovo určující, co konkrétně chceme zjistit. Použijte místo teček otazník a podívejte se, jaká klíčová slova který protokol podporuje. Všechny by měly podporovat alespoň klíčové slovo `database`, vyzkoušejte ho a případně další.

3.4.2 Single-area OSPF pro IPv4

Pokud má router směrovat a využívat k tomu informace protokolu OSPF, musíme na něm nejdřív vytvořit směrovací proces protokolu OSPF a určit mu jeho parametry:

- RouterID (tak se bude představovat ostatním routerům v OSPF doméně)
- které vlastní připojené sítě má router ohlašovat sousedům
- případně další, například seznam pasivních rozhraní (takových, za kterými není router), určení referenčního čísla pro výpočet ceny jednotlivých spojů, atd.

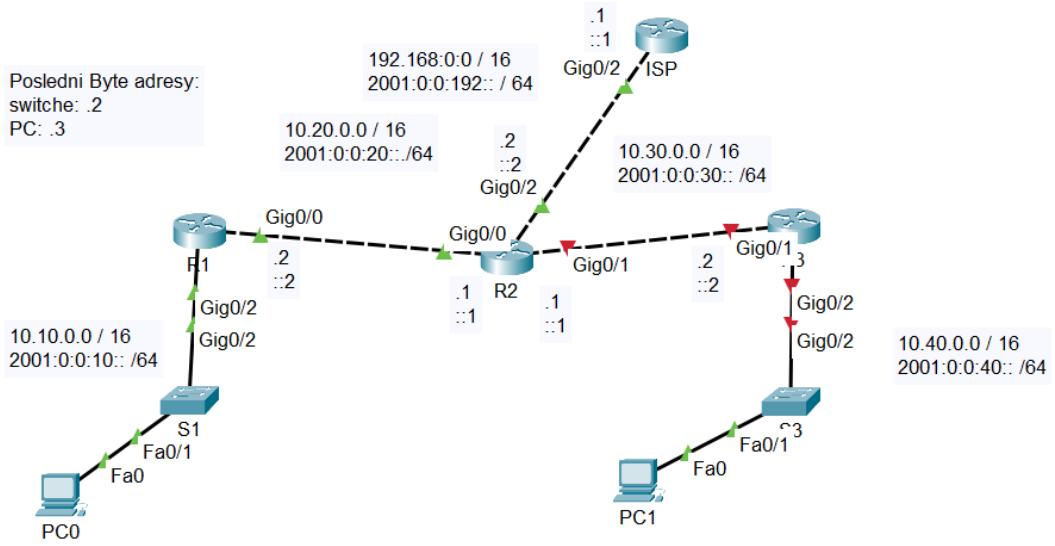
Pro RouterID existují tyto možnosti (v tomto pořadí, u první použitelné možnosti se zastaví):

1. nakonfigurujeme ručně, mělo by mít formu podobnou IPv4 adrese (čtyři 1B čísla oddělená tečkou),
2. zařízení použije nejvyšší IP adresu loopbacku (pokud nějaký existuje),
3. zařízení použije nejvyšší IP adresu „normálního“ rozhraní.

Pokud máme jen pár routerů a nepotřebujeme ID pro jiné účely, klidně můžeme ID určit ručně. Jinak je praktické použít loopback (u některých směrovacích protokolů, zejména BGP, se toto důrazně doporučuje). Třetí možnost není doporučovaná, protože znamená potenciální problémy při změnách v síti.

Příklad

Na webu je předpřipravný soubor `routingOSPF.pkt` s infrastrukturou z obrázku 3.6. Pokud soubor nemáte k dispozici, vytvořte tuto infrastrukturu ručně se zachováním použitých portů.



Obrázek 3.6: Připravená síť pro konfiguraci OSPF

Na spoji vedoucím do stub sítě (=sítě, která není průchozí) bude vždy první adresa z rozsahu. U ostatních sítí je poslední byte naznačen na obrázku. Switche mají adresu končící 2, počítače adresu končící 3.

Brána na počítačích i switchích je IP adresa routeru na rozhraní vedoucím do dané sítě, v případě IPv6 je brána pro počítače a switche vždy `FE80::1` (v obou stub sítích).

Link-local adresa: na rozhraních do stub sítí nastavte vždy `FE80::1`, na ostatních rozhraních použijte poslední byte stejný jako v jiných adresách téhož rozhraní.

Tabulka 3.3: IP adresy na portech v příkladu na OSPF

Zařízení	Rozhraní	IP adresy	Brána

R1 (OSPF)	`g0/0`	`10.20.0.2/16`
		`2001:0:0:20::2/64`
		`fe80::2`	

	`g0/2 (pasivní)`	`10.10.0.1/16`
		`2001:0:0:10::1/64`
		`fe80::1`	

Pokračování na další stránce

Tabulka 3.3: IP adresy na portech v příkladu na OSPF (Pokračování)

Zařízení	Rozhraní	IP adresy	Brána
R2 (OSPF)	g0/0	10.20.0.1/16 2001:0:0:20::1/64 fe80::1	
	g0/1	10.30.0.1/16 2001:0:0:30::1/64 fe80::1	
	g0/2 (pasivní)	192.168.0.2/16 2001:0:0:168::2/64 fe80::2	
R3 (OSPF)	g0/1	10.30.0.2/16 2001:0:0:30::2/64 fe80::2	
	g0/2 (pasivní)	10.40.0.1/16 2001:0:0:40::1/64 fe80::1	
ISP	g0/2	192.168.0.1/16 2001:0:0:168::1/64 fe80::1	
Sw1	vlan1	10.10.0.2/16	10.10.0.1
Sw2	vlan1	10.40.0.2/16	10.40.0.1
PC0		10.10.0.3/16 2001:0:0:10::3/64	10.10.0.1 fe80::1
PC1		10.40.0.3/16 2001:0:0:40::3/64	10.40.0.1 fe80::1

V předpřipraveném souboru je většina IP adres nasazená, kromě routeru R3 a PC1. Tam nakonfigurujte IPv4 a IPv6 adresy. Router ISP nám simuluje poskytovatele internetu, je tam vytvořena default route kvůli odpovědím na ping.

Nejdřív se zaměříme na router R2. Vytvoříme loopback 0, jehož adresa bude automaticky použita jako routerID:

```
int lo0
ip addr 2.2.2.2 255.255.255.255
```

Vytvoříme směrovací proces (přidělíme mu číslo 10) a zadáme do něj připojené sítě – pozor, je třeba použít wildcard masku podobně jako v ACL, a také označíme rozhraní vedoucí k ISP jako pasivní:

```
router ospf 10
    passive-interface g0/2
    network 10.20.0.0 0.0.255.255 area 0
    network 10.30.0.0 0.0.255.255 area 0
    default-information originate
```

Na pasivní rozhraní nebude proces OSPF posílat žádné informace o sítích a sousedech, protože našemu ISP není nic do struktury naší sítě. Svým sousedům R1 a R3 budeme ohlašovat uvedené sítě. Je připsána oblast 0, protože OSPF sice umožňuje rozdělit routery do několika oblastí, ale v našem případě to nemá moc smysl.

Poslední příkaz způsobí, že sousedům bude přeposílána (propagována) i informace o default route. Aha, ještě jsme defaultní cestu neurčili, tak to provedeme (v globálním konfiguračním módu):

```
ip route 0.0.0.0 0.0.0.0 g0/2
```

Na routerech R1 a R3 nastavte směrování OSPF sami. Nebudete tam řešit default route, jen vytvořte loopback s adresou `1.1.1.1` (u R1) a `3.3.3.3` (u R3) a ve směrovacím procesu určete připojené sítě s inverzní maskou v oblasti 1. Pasivní rozhraní budou ta, která vedou do stub sítě (není tam už žádný další OSPF router).

Pokud bychom chtěli RouterID určit radši ručně a nenechávat to na loopbacku, vložíme do subkonfiguračního módu pro směrování tento příkaz:

```
router ospf 10
    router-id 2.2.2.2
```

Až budete mít konfiguraci hotovou, vyzkoušejte tyto `show` příkazy:

```
sh ip route
sh ip route ospf
sh ip protocols
sh ip ospf
sh ip ospf neighbors
sh ip ospf int br
...
```

(ověřte si, jestli existují i další klíčová slova/podpříkazy u příkazu `sh ip ospf`).

Poznámka

Pokud z nějakého důvodu potřebujete změnit RouterID, pak po změně použijte příkaz

```
clear ip ospf process
```

Tím se restartuje proces OSPF a znovu si načte své vlastnosti včetně RouterID.

3.4.3 Single-area OSPF pro IPv6

Pro směrování IPv4 adres stačí OSPFv2, pro směrování IPv6 adres se používá OSPFv3, který má trochu jinou syntaxi. Adresy připojených sítí, které mají být propagovány sousedům, nepíšeme do směrovacího subkonfiguračního módu, ale přímo na jednotlivá rozhraní. Do subkonfiguračního módu zapisujeme RouterID (pokud ho chceme určit ručně), příkaz pro propagování default route a další.

Nejdřív ovšem zapneme IPv6 směrování:

```
ipv6 unicast-routing
```

Vytvoříme proces pro IPv6 směrování, můžeme mu přidělit totéž číslo jako pro IPv4:

```
ipv6 router ospf 10
    passive-interface g0/2              ! může být na různých routerech odlišné
    default-information originate       ! toto bude jen u routeru R2
    exit
```

Pak půjdeme na jednotlivá rozhraní a zapneme na nich propagování sítě, do které dané rozhraní vede (zde pro router R1):

```
int g0/0
    ipv6 ospf 10 area 0
    int g0/1
    ipv6 ospf 10 area 0
```

Zadává se protokol, číslo směrovacího procesu a oblast.

Pokud je rozhraní označeno jako pasivní, tak to ještě neznamená, že bychom na něm nezadávali poslední probírané příkazy. Pasivní rozhraní je takové, na které nebudeme posílat směrovací informace (protože je za ním ISP, nebo síť bez dalšího routeru, třeba switch nebo přímo koncové zařízení). Příkaz `ipv6 ospf xxx area xxx` však znamená, že síť, která je k tomuto rozhraní připojená, má být propagována sousedům.

Takže:

- pokud rozhraní vede k ISP, bude označeno jako pasivní a nepoužijeme na něm zmíněný příkaz,
- pokud rozhraní vede do naší sítě (ve které není další router), pak bude označeno jako pasivní, ale použijeme na něm zmíněný příkaz.

`Show` příkazy jsou podobné, jen všude bude místo `ip` uvedeno `ipv6`.

3.5 Směrování mezi autonomními systémy

Z přednášek víme, co je to autonomní systém, a že tedy souvisí s provozovatelem sítě. Zatím jsme řešili směrování uvnitř autonomního systému nebo dokonce uvnitř jeho části (protože pokud nemáme vlastní autonomní systém, jsme součástí autonomního systému svého ISP, což je zcela obvyklá situace). Pokud potřebujeme řešit směrování mezi autonomními systémy, jde o exterior routing, a používáme vnější směrovací protokol (typicky BGP nebo některou jeho variantu).

V konfiguraci budeme potřebovat ASN (Autonomous System Number), tedy číslo našeho autonomního systému.

3.5.1 eBGP pro IPv4

BGP je exterior směrovací protokol a zatímco u OSPF určujeme číslo procesu (které je lokální a míří směrem dovnitř routeru), u BGP určujeme číslo ASN. Zde budeme používat čísla z testovacího rozsahu.

Pro RouterID platí u BGP totéž co u OSPF: buď ho stanovíme ručně ve směrovacím procesu, nebo se použije nejvyšší adresa nasazená na některém loopbacku, nebo nejvyšší adresa nasazená na některém běžném rozhraní (priorita podle tohoto pořadí). Obvykle používáme loopback.

Pokud se použije adresa loopbacku, její role je identifikace daného routeru vzhledem k ostatním BGP peerům, ale ve skutečnosti se dá využít i pro posílání BGP komunikace (nastavení je mimo rámec tohoto

předmětu). Výhodou je, že loopback IP adresa je vždy aktivní a funkční, i když některé z fyzických rozhraní přestane fungovat, komunikace je tedy stabilnější.

Předpokládejme tedy, že RouterID řešíme pomocí loopbacku, který už máme vytvořený a nastavený. Nejdřív ve směrovacím procesu (určeném číslem ASN) definujeme sousedy a jejich ASN:

```
router bgp 64500
    neighbor 12.0.0.2 remote-as 64501
    ...
```

Postupně pro všechny BGP sousedy (nemusejí nutně mít adresu z téže podsítě). U každého uvádíme ASN, které logicky bude jiné než to, které jsme napsali k směrovacímu procesu (jsme zatím u eBGP, sousedi jsou z jiných autonomních systémů).

Sousedství si můžeme ověřit příkazem

```
sh ip bgp neighbors
```

Vrátíme se do směrovacího procesu a použijeme příkaz `network`. Ten má trochu jiný význam než u vnitřních protokolů: u vnitřího protokolu se zadaná síť porovná s adresami na připojených rozhraních a síť za rozhraním je propagována sousedům. U BGP se zadaná síť (včetně masky) porovnává s obsahem směrovací tabulky, a pokud je tam nalezena, bude propagována sousedům. Z toho vyplývá, že propagovat můžeme i takovou síť, která není přímo připojená (může být do směrovací tabulky umístěna jiným směrovacím protokolem). Syntaxe:

```
network 12.0.0.0
network 12.128.0.0 mask 255.255.0.0
```

Masku uvádíme, pokud neodpovídá třídě.

Shrňme si `show` příkazy pro protokol BGP:

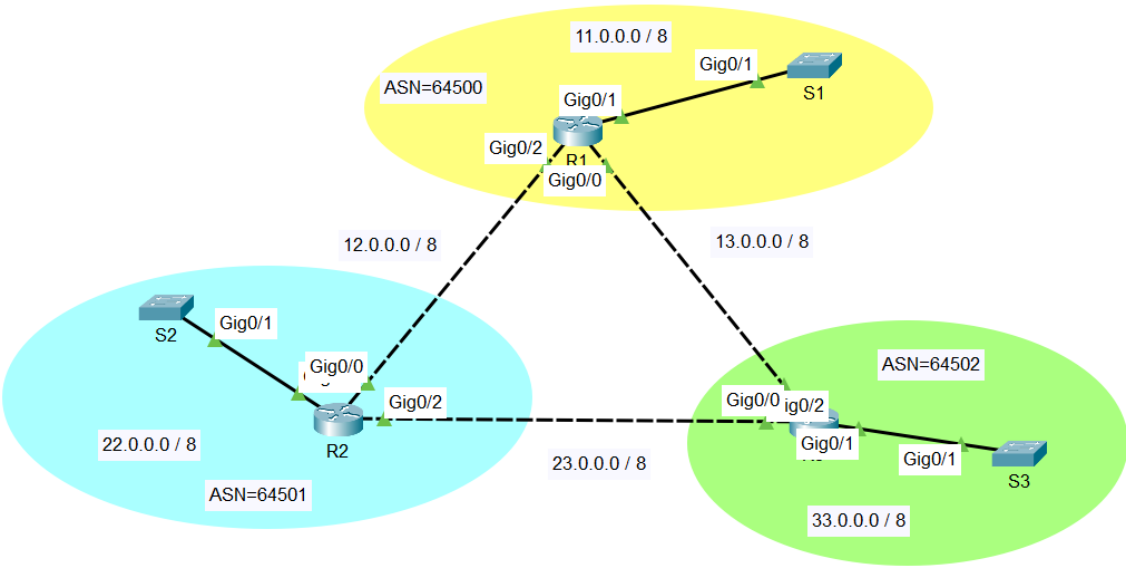
```
sh run
sh ip protocol
sh ip bgp             ! zobrazí BGP tabulku, ze které se určí položky do směrovací tabulky
sh ip bgp neighbors
sh ip bgp summary
sh ip route
debug ip bgp
no debug ip bgp
```

Poznámka

Protokol BGP má poměrně dlouhou dobu konvergence, do směrovacích tabulek se změny promítají někdy až po několika minutách.

Příklad

Na obrázku 3.7 je (poněkud nepravděpodobná) síť se třemi autonomními systémy, ke stažení je předpřipravený soubor `routing-eBGP.pkt`. Na všech rozhraních jsou nasazeny IP adresy, na routeru R2 je provedena celá základní konfigurace eBGP (peers a sítě), na routeru R1 je zatím částečná konfigurace (zjistěte, co chybí, dokončete), na routeru R3 je nutno nakonfigurovat BGP. Nezapomeňte ověřit, jestli existují loopbacky se správnými adresami.



Obrázek 3.7: První příklad na eBGP

Pro zjednodušení budeme chtít, aby všechny přítomné sítě byly propagovány přes BGP, třebaže to není nutné a ani to není zvykem. Po dokončení konfigurace ověřte dostupnost jednotlivých sítí a vyzkoušejte různé `show` příkazy.

3.5.2 Další možnosti eBGP

Protokol eBGP toho umí celkem hodně a navíc některé úlohy se v něm řeší poněkud specificky, podíváme se na jeho různé vlastnosti.

Sumarizace. V BGP je defaultně vypnuta sumarizace, ta se samozřejmě dá zapnout, sumarizuje se podle třídy. K tomu slouží příkaz `auto-summary` v subkonfiguračním směrovacím módu.

Také je možné zajistit agregaci jen určitých adresních rozsahů, k tomu slouží příkaz `aggregate-address` použitý ve směrovacím subkonfiguračním módu.

Tento příkaz má především jako parametr agregovaný rozsah adres, ale I další parametry, jejich vysvětlení najdeme například na https://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/5441-aggregation.html

Redistribuce směrovací informace jiného protokolu. Jak bylo výše uvedeno, BGP propaguje sítě převzaté z vlastní směrovací tabulky, nebere je z rozhraní. Takže není problém distribuovat peerům záznam ze směrovací tabulky pocházející od jiného směrovacího protokolu, například od OSPF. Ovšem mělo by se jednat o veřejné adresy, nikoliv soukromé. Můžeme do směrovacího procesu přímo zadat příkaz `network` s danou sítí (včetně masky, pokud je jiná než pro třídu), nebo se dá použít příkaz pro redistribuci adresních rozsahů (například `redistribute ospf 1 match external` distribuuje externí cesty protokolu OSPF).

Taktéž je možné určit adresy pro redistribuci přesněji, a to pomocí ACL, příklad najdeme například na https://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/6500-bgp-pix.html (asi tak v polovině stránky).

BGP umí také předávat default route. Nejdřív vytvoříme nulovou cestu a nasměrujeme do zařízení Null0, pak propagujeme přes BGP:

```
ip route 0.0.0.0 0.0.0.0 null0
router BGP xxxxx
    network 0.0.0.0
```

Použití loopbacku pro posílání směrovací informace a zvýšení TTL. Směrovací informace BGP může být posílána na IP adresu síťového rozhraní (například router R2 na obrázku výše bude posílat směrovací informace pro router R1 přes TCP spoj na adresu `12.0.0.1`), nebo se dá nakonfigurovat posílání na adresu loopbacku použitou jako RouterID (zde `1.1.1.1`). V Packet Traceru je s tím problém, ale na reálném zařízení bychom to provedli takto: nejdřív zprovoznit loopback na routeru R1:

```
int lo0
    ip addr 1.1.1.1 255.255.255.255
    exit
```

Pak na routeru R2 sdělíme, že i pro TCP spojení pro BGP informace se má používat loopback na R1:

```
router bgp 64501
    neighbor 1.1.1.1 remote-as 64500
    neighbor 1.1.1.1 update-source Loopback0
    neighbor 1.1.1.1 ebgp-multihop 2        ! číslo větší než 1, použije se jako TTL
```

Výhodou je větší odolnost komunikace proti výpadkům sítě, a také možnost použít tutéž IP adresu při přístupu do zařízení přes různé cesty. Ale pozor: eBGP standardně používá TTL=1, jenže pokud router přijme BGP segment přes fyzické síťové rozhraní a cílová adresa je jeho loopback, pak uvnitř routeru proběhne směrování, při kterém se TTL sníží o 1, a až potom se PDU dostane na loopback, pokud ovšem TTL >0. Proto je nutné použít příkaz `ebgp-multihop` a nastavit takto TTL na hodnotu vyšší než 1.

Problém s navýšením TTL také řešíme, pokud mezi dvěma eBGP sousedy je ještě jiné L3 zařízení. Také musíme nastavit hodnotu TTL na vyšší číslo. Protokol iBGP tento problém nemá, tam je defaultní hodnota TTL větší než 1.

Úkol

Na webu je předpřipravený soubor `routing-BGP2a.pkt`. Přímo v souboru jsou instrukce, co je třeba udělat. Nezapomeňte nakonfigurovat také propagování default route přes BGP. Všimněte si, jakým způsobem je z propagován rozsah pro veřejnou síť z AS 64501 a uvědomte si, proč je v tomto AS také protokol OSPF (které rozhraní je pasivní?). Zobrazte si také směrovací tabulky různých routerů.

Filtrování propagování sítí v BGP. Jak víme, BGP dokáže propagovat peerům cokoliv, co najde ve směrovací tabulce, ovšem ne vše z tabulky je vhodné poslat do světa. V předchozím příkladu jsme zajistili propagování jedné veřejné sítě v každém stub AS, kdežto soukromé rozsahy zůstaly soukromými, ale ve skutečnosti je několik možností jak vyfiltrovat sítě pro propagování:

- vytvoříme standardní nebo extended ACL a ve směrovacím procesu použijeme příkaz

```
neighbor soused distribute-list označení_ACL out       ! k sousedovi půjde pouze toto
neighbor soused distribute-list označení_ACL in        ! od souseda přijmu pouze toto
```

- použijeme příkaz `ip prefix-list název permit adresa_sítě` (při určení sítě použijeme CIDR zápis, tedy s délkou prefixu), pak ve směrovacím procesu použijeme příkaz

  ```
  neighbor soused prefix-list název in
  ```

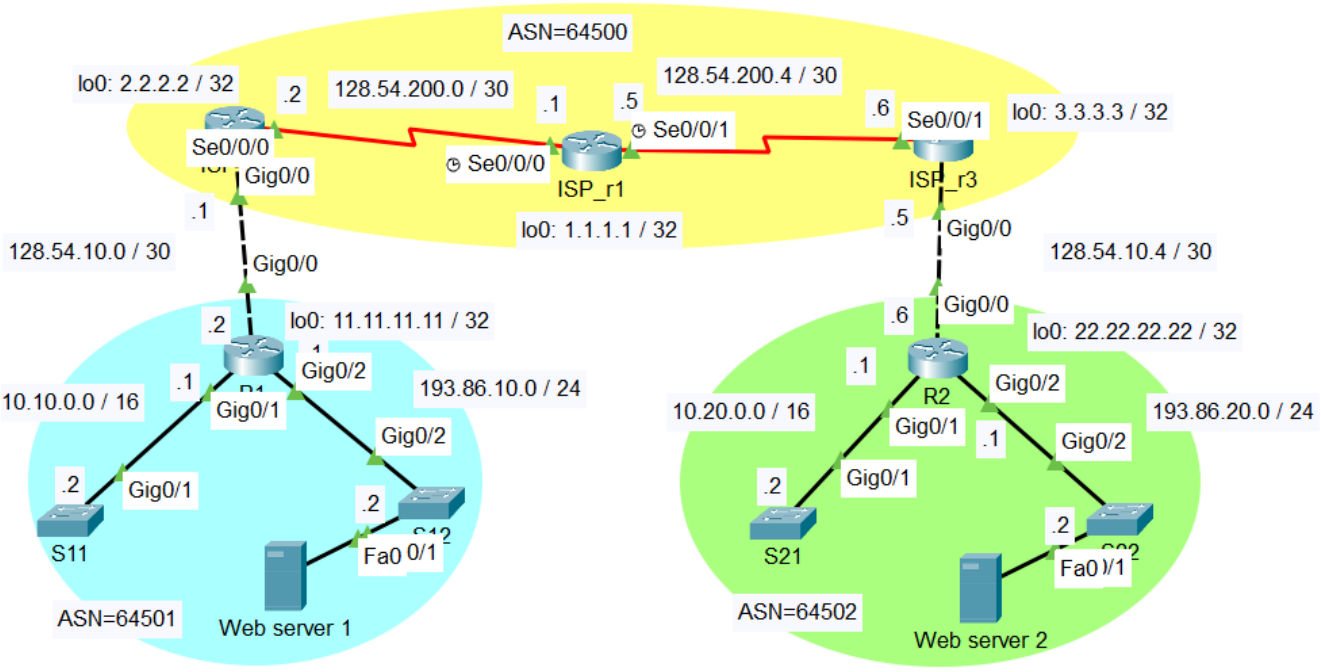
  (nebo `out`, podle toho, který směr konfigurujeme)
- další možností je využít příkaz `route-map`, případně další.

3.5.3 iBGP: směrování uvnitř AS

Zatímco eBGP předpokládá, že BGP sousedi mají různá čísla ASN, iBGP se používá tehdy, když BGP sousedi mají totéž číslo ASN (patří do téže autonomní oblasti). Rozdíl v konfiguraci není až tak velký, taktéž se používá příkaz `neighbor xxxx remote-as yyyy`, ale za klíčovým slovem `remote-as` je stejné číslo ASN jako u označení BGP procesu.

Příklad

Tento příklad budeme řešit jen v teoretické rovině, protože Packet Tracer nepodporuje iBGP...



Obrázek 3.8: Síť pro konfiguraci iBGP

Z obrázku je patrné, že ISP propojuje dvě firmy (nebo to mohou být dvě pobočky téže firmy), z nichž každá má svůj autonomní systém. Pokud chce ISP zprostředkovávat cesty mezi „podřízenými“ AS a posílat si je přes své routery, musí použít iBGP (nebo mít ve své síti OSPF a redistribuovat mezi BGP a OSPF). Na routeru ISP_r2 (v žluté zóně vlevo) by byla tato posloupnost příkazů:

```
int lo0
    2.2.2.2 255.255.255.255
    no shut
    exit
```

```
router bgp 64500
    neighbor 128.54.200.1 remote-as 64500      ! iBGP
    neighbor 128.54.10.2 remote-as 64501       ! eBGP
```

V případě použití iBGP se pravděpodobněji budou řešit různé možnosti filtrování směrovacích informací (určitým peerům budeme posílat jen částečné updaty směrovacích informací, ne celou tabulku, případně opačně – od určitých peerů nebudeme chtít vše, ale jen část směrovacích informací). Důvodem je jak bezpečnost (zejména v prvním případě) nebo třídění veřejných a soukromých sítí, ale také ochrana před zbytečným zahlcováním směrovacích tabulek, určování optimálnějších cest a ochrana před smyčkami.

Ve směrovací tabulce má iBGP 10× horší AD než eBGP (najděte tabulku ve skriptech pro přednášky a ověřte).

3.6 Programování socketů

O programování UDP socketů v Pythonu najdeme informace například na
https://wiki.python.org/moin/UdpCommunication

Souhrn kapitoly 3

V této kapitole jsme se zabývali zejména tím, co souvisí s routery a IP adresací. První sekce byla věnovaná protokolu DHCP pro IPv4 a IPv6, pak jsme se zaměřili na základy směrování (statické směrování a práci se směrovací tabulkou). Následovaly seznamy řízení přístupu ACL (standardní i extended varianta) včetně doporučení pro jejich používání. Další sekce byla věnována dynamickému směrování pomocí vnitřních směrovacích protokolů, soustředili jsme se zejména na protokol OSPF. Následovala sekce o směrování pomocí vnějších směrovacích protokolů na páteřních sítích, konkrétně jsme se zabývali protokolem BGP v jeho formách eBGP a iBGP.

Kapitola 4

Bezpečnost a správa

Rychlý náhled:

Klíčová slova:

Cíle studia:

4.1 Debugging

4.2 Překlady adres

4.2.1 NAT

4.2.2 DNS

Můžeme si přímo na routeru rozjet místní překlad názvů. Například: ip host webserver 192.168.25.4 Pak můžeme v různých příkazech (nejen v pingu) místo IP adresy používat tuto jmennou adresu. Ovšem tato vazba bude viditelná opravdu jen lokálně na tomto zařízení. Můžu také nastavit adresu jmenného serveru: ip name-server ip.adresa Pokud jsme si hráli s ACL, je třeba povolit UDP provoz na portu 53 (tedy „domain“): access-list 105 permit udp any any eq domain access-list 105 permit udp any eq domain any Z bezpečnostních důvodů se naopak často zakazuje DNS překlad: no ip domain-lookup Příklad: Pokud si vytvoříme název pro některý server a pak zakážeme domain lookup, bude fungovat ping na vytvořený název?

4.3 GRE tunely

4.4 IP SLA

v souboru s BGP

4.5 Port Mirroring

dto.

4.6 NTP

4.7 Konfigurace multicast komunikace