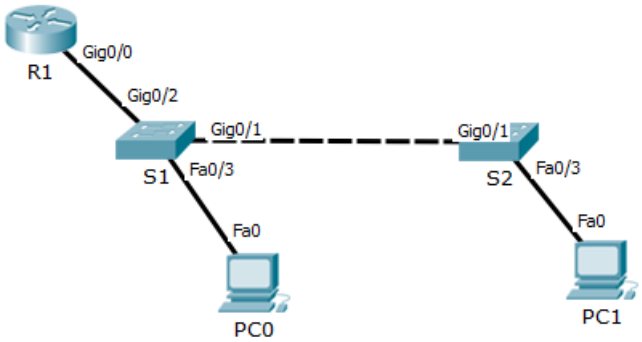
VLAN, STP, blokování floodingu

Virtuální lokální sítě

Zatím potřebujete dva (jakékoliv) switche (nazvěte je S1 a S2, propojte kříženým kabelem, na obou stranách g0/1). Na switchích nastavíme VLANy a nakonfigurujeme je. Nejdřív je vytvoříme (v obou switchích):

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



Porty: 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. Jak například mohou vypadat adresy pro připojené počítače? Mohou být taky ze sítě 10.99.0.0/16?

Pokud se na portu nedaří nastavit trunk, bývá (na multilayer switchi) důvodem to, že je encapsulation nastaveno na jiné než IEEE 802.1Q:

```
switchport trunk encap dot1q
do sh int … switchport    ; ověříme, jestli už se „zapouzdřuje“ dot1q (i.e. IEEE 802.1Q)
```

Ke každému switchi připojte jeden počítač (porty f0/3).

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

Router-on-a stick

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

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 zapojte router (na switchi g0/2, na routeru g0/0 či podobný). Ke switchi S2 připojte ještě jeden počítač (port f0/4). Na počítači nastavte IP adresu 10.20.0.4/16, bránu 10.20.0.1, na switchi S2 nastavte u portu f0/4 VLAN 20. Ověřte vzájemnou dostupnost počítačů 10.10.0.3 a 10.20.0.4.

Na switchi S2 nastavte trunk směrem k routeru:

```
int g0/2
      switchport mode trunk
      switchport trunk native vlan 1
      switchport trunk allowed vlan 10,20,99
```

Na routeru:

```
int g0/0.10
      encapsulation dot1q 10
      ip addr 10.10.0.1 255.255.0.0
int g0/0.20
      encapsulation dot1q 20
      ip addr 10.20.0.1 255.255.0.0
int g0/0.99
      encapsulation dot1q 99
      ip addr 10.99.0.1 255.255.0.0
int g0/0
      no shut
```

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.

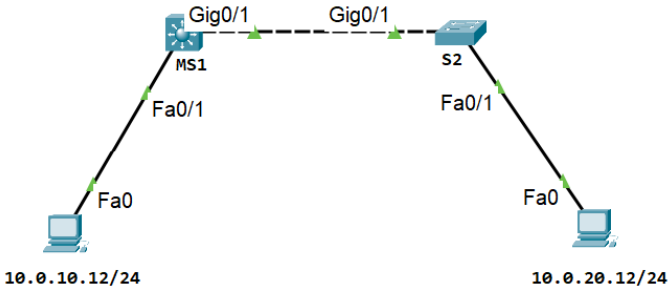
Co se stane, když odstraníte některou VLAN, například `no vlan 10`?

Multilayer switch

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.

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

Na prvním PC je IP adresa 10.0.10.12/24 a brána 10.0.10.1. Na druhém PC 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)#int g0/1
S2(config-if)#switchport mode trunk
```

Přesuneme se na multilayer switch. Vytvoříme VLANy a do první přiřadíme port:

```
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)#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
```

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

```
MS1(config-if)#exit
MS1(config)#ip routing

MS1(config)#do sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
* - candidate default, U - per-user static route, o - ODR
P - periodic downloaded static route

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 ; uvidíme obě virtuální rozhraní s nastavenými IP adresami
sh ip route  ; ve směrovací tabulce jsou obě virtuální připojené sítě
```

Blokování neznámých unicast/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
```

Pokud na takový port přijde rámec s neznámou cílovou adresou, bude zahozen. Co myslíte, jaký to má důsledek na provoz 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“.

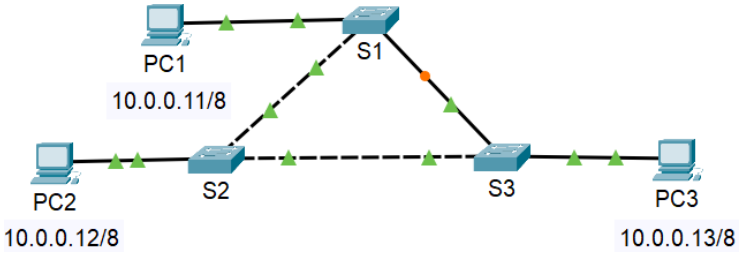
Jak si ověřit nastavení:

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

Nastavení blokování floodingu je nezávislé na VLANách.

Spanning-tree

Konfigurace STP



Vytvořte následující síť switchů s počítači:

Předpokládejme, že tedy jako první byl „vytvořen“ switch S1, atd. Který switch je root switch – i 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 záložní primary byl switch S2, provedeme na něm tento příkaz:

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

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.

Předpokládejme, že porty f0/5-20 nejsou připojeny k jinému switchi, 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:

```
spanning-tree portfast default
```

a pak na těch, 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
```

Bezpečnost

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.

Pro množinu portů:

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

Pro všechny portfast porty:

```
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-disabled, můžeme zkontrolovat jeho stav takto:

```
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):

```
show errdisable recovery        ; zjistěte, zda errordisable recovery je zapnutý
conf t
      errordisable recovery cause bpduguard        ; zapněte pro daný účel
      errordisable recovery interval 30            ; po 30 s se provede recovery
```

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
```

Pokuste se vyrobit „kravatu“, tj. dva switche propojené dvěma kabely (viz obrázek vlevo). Ověřte, jak se dohodly na nastavení spanning-tree.