VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

MODELOVÁNÍ A ANALÝZA SPOLEHLIVOSTI POČÍTAČOVÉ SÍTĚ VUT

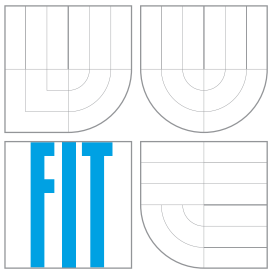
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

ZDENĚK KRAUS

BRNO 2011

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

MODELOVÁNÍ A ANALÝZA SPOLEHLIVOSTI POČÍTAČOVÉ SÍTĚ VUT

MODELLING AND RELIABILITY ANALYSIS OF CAMPUS NETWORK AT THE BUT

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

ZDENĚK KRAUS

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. VLADIMÍR VESELÝ

BRNO 2011

Abstrakt

Práce popisuje technologie a architekturu použitou na počítačové síti VUT. Vybrané technologie jsou v textu vysvětleny i s některými jejich variantami. V práci je navržen simulační model zařízení přepínače obsahující tyto technologie. Dále popisuje implementaci navrženého modelu, kde dále doplňuje detaily technologií spojené s implementací. Poté na sadě testovacích simulačních scénářů ověřuje správnost modelu vůči teoretickému základu. Model je také ověřen na reálném zapojení v laboratoři. Na závěr jsou navržena možná vylepšení modelu.

Abstract

This thesis informs about technologies and architecture of computer network at BUT. It describes chosen technologies and some of theirs variants. Further, it contains simulation model design of a network switch device. That is followed by the model implementation with detailed description of model parts. Then, the model is tested on a set of simulation scenarios and it is evaluated in comparison with theoretical basis and real devices. In conclusion, thesis suggests future model development and improvements.

Klíčová slova

VUT, počítačové sítě, simulace, OMNeT++, přepínač, MAC, VLAN, STP, RSTP, PVST, MSTP

Keywords

BUT, computer network, simulation, OMNeT++, switch, MAC, VLAN, STP, RSTP, PVST, MSTP

Citace

Zdeněk Kraus: Modelování a analýza spolehlivosti
počítačové sítě VUT, diplomová práce, Brno, FIT VUT v Brně, 2011

Modelování a analýza spolehlivosti počítačové sítě VUT

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana Ing. Vladimíra Veselého. Všechny zdroje, které jsem použil při vypracování této práce jsem svědomitě uvedl na závěr v seznamu literatury a průběžně jsem se v textu na ně odkazoval.

.....
Zdeněk Kraus
25. května 2011

Poděkování

Rád bych poděkoval vedoucímu své práce, který mě morálně podporoval, motivoval a podněcoval mou tvořivost. Dále bych chtěl poděkovat svému kolegovi Marku Černému za výbornou spolupráci při objevování „temných“ zákoutí simulačního nástroje OMNeT++. Velmi důležitá část mé vděčnosti patří mým rodičům za veškerou dostupnou podporu a pomoc, kterou mi vždy a hlavně v období dokončování této páce poskytli.

Obsah

1 Úvod

V dnešní velmi hektické době jsme zvyklí na rychle se rozšiřující a vyvíjející se technologie. Výjimkou nejsou ani počítačové sítě. Vzhledem k velkému množství složitých služeb a funkcí, které spolu často musí kooperovat, se neubráníme problémům. A to ani s použitím vysoce specializovaných metodologií.

Ideálním řešením by jistě byl automatický a plně verifikovaný návrh, který by síť analyzoval. Nyní bohužel máme omezené prostředky pro verifikaci natož automatizovaný návrh. Vzhledem ke složitosti a různorodosti technologií, zařízení a cílových prostředí je velmi efektivní použít simulaci. Výhodou simulace je take cena přístrojů, která se omezuje na cenu počítačů, na nichž se bude simulovat. Simulace je efektivní a flexibilní prostředek, díky níž není nutno zkoušet hypotézy na produkční síti.

Práce se zaměřuje na prostředí počítačové sítě VUT, a proto jsou v ní popsány vybrané technologie s ohledem na tuto síť. V rámci skupiny ANSA vznikl model směrovače s podporou směrovacích protokolů a filtrování ACL, vizte [21, 10, 22, 23] [NON VIDI]. Proto se také tato práce zaměřuje na sítě přepínané a v jejím rámci vznikl model zařízení přepínače s podporou technologií, které jsou v rámci VUT sítě použity.

Kapitola 2 popisuje technologie, architekturu a konfiguraci služeb sítě VUT.

V kapitole 3 je teoretický popis přepínaných sítí. Je zde vysvětleno základní přepínání a také přepínání ve *Virtuálních lokálních sítích*, jež jsou použity v počítačové síti VUT. Dále je uveden teoretický popis původního protokolu *Spanning Tree* a jeho variant, které byly postupem času vyvinuty. Vysvětluje rozdíly a vylepšení novějších variant, jež byly vyvinuty tvůrci původního STP – IEEE, ale také popisuje proprietární varianty vyvinuté firmou *Cisco Systems, Inc.* To je obsaženo v kapitole 4.

Následuje teoretický návrh modelu v kapitole 5, který vychází ze standardů těchto vybraných technologií. Jsou to IEEE 802.1D-1998 ([14]), IEEE 802.1D-2004 ([15]) a IEEE 802.1Q ([16]). Teoretický návrh je převzat a rozšířen ze semestrálního projektu. Zde jsou popsány funkce potřebné k přepínání na síti s VLAN a redundantní topologií.

Kapitola 6 popisuje architekturu a implementaci navrženého modelu. Vysvětluje proces přepínání rámců uvnitř VLAN, datové struktury a funkce pro uchování nastavení a informací o síti. Zde diplomová práce navazuje na semestrální projekt. Doplňuje detaily implementace významných částí STP a uvádí specifikaci konfiguračního souboru.

Následující kapitola 7 obsahuje specifikaci několika simulačních scénářů, které ověřují správnost implementovaného modelu. Scénáře sestávají ze simulace základního přepínání bez konfigurace, přepínání v rámci VLAN, ověření konvergence STP, funkce STP nad oddělenými VLAN a simulaci na topologii sítě VUT. Konvergence Spanning Tree protokolu byla navíc ověřena s reálnými zařízeními v laboratoři.

V kapitole 8 je navržen možný budoucí vývoj modelu. Jsou zde shrnuty části, které bude pro simulaci daných technologií ještě vhodné doplnit.

A kapitola 9 shrnuje dosažené výsledky a obsah práce.

2 Síť VUT

Tato kapitola je vypracována na základě informací z přednášky [13].

Páteřní počítačová síť je tvořena aktivními prvky firmy *HP*, *Extreme Networks* a *3Com*. Propojuje 18 budov, kde každá z nich je spojena nejméně dvěma nezávislými optickými kabely. Nejvýznamnějšími uzly jsou Antonínská, Technická a Kounicova. V uzlu Kounicova je síť VUT propojena směrovačem *Cisco CRS-1* s národní vysokorychlostní počítačovou síti *CESNET*.

Síť je postavena na technologii *Virtuálních Lokálních Sítí* – VLAN, které specifikuje standard IEEE 802.1Q ([16]). Prvky jsou zálohovány UPS a dvojím napájením. Přenosová kapacita páteřních linek je 10 Gb/s nebo 1 Gb/s.

V síti existuje okruh postavený čistě na linkové vrstvě (dále L2 okruh). Jedná se tedy o přepínanou síť. Jelikož L2 okruh poskytuje vzdáleným pracovištím zálohované spoje k centru, je redundantní a tedy obsahuje smyčky. Proto v rámci L2 okruhu pracuje Spanning Tree Protocol. Ten volí aktivní topologii a umožňuje komunikaci po přepínané redundantní síti. Dokonce v případě havárie linky nebo uzlu, umožňuje dynamickou změnu aktivní topologie. Jako další zálohování konektivity se používá VRRP – Virtual Router Redundancy Protocol. Ten umožňuje mít v síti redundantní zařízení, které se pro síť transparentně tváří jako jedno zařízení fyzické. Při výpadku je funkce „havarovaného“ zařízení automaticky nahrazena záložním. Pro více informací o VRRP vizte [9] [NON VIDI].

L2 okruh se používá pro WiFi Sítě, Eduroam, jež jsou rozprostřeny přes celou VUT síť. Dále přes tento okruh putuje provoz IPv6 a komunikace inteligentních budov.

Směrovaná část sítě (dále L3 síť) poskytuje IPv4 konektivitu a filtraci provozu pro jednotlivé budovy. Je směrována pomocí protokolů RIP a OSPF. Aktivní prvky, které se účastní směrování jsou zařízení *Cisco Systems*, *HP*, *Extreme networks* a počítače se programovým směrováním (např. gated nebo zebra).

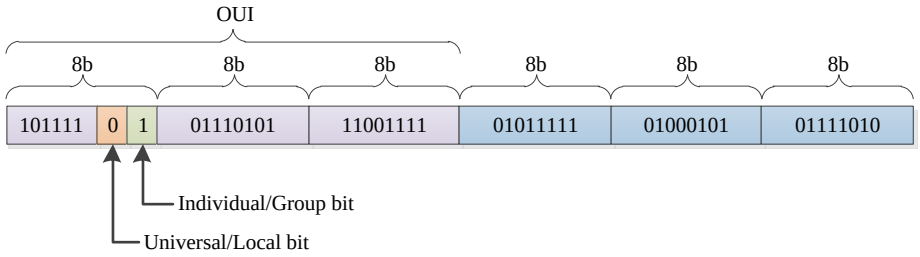
Výše zmíněná IPv6 konektivita je poskytována PC směrovačem, který je připojen tunelem do uzlu sítě CESNET2. Na této součásti sítě pracují technologie IEEE 802.1Q, OSPF v síti VUT a BPG4+.

3 Přepínané sítě

Informace o přepínaných sítích v této kapitole byly čerpány z následujících zdrojů: [20, 15, 16, 18, 17]

Obecně je síť množina uzlů schopných přepravovat informace, které jsou propojené množstvím datových spojů. Na okrajích potom leží připojeny koncové uzly, které pomocí dané sítě komunikují. Zasílání dat probíhá na základě určitých adres. V ethernetu jsou to adresy MAC o délce 48b znázorňuje 3.1. První část o délce 24b je přidělována organizací IEEE jednotlivým subjektům (např. výrobcům), nebo protokolů, pro jejich univerzální identifikaci dle adresy. Tato přidělená část se nazývá OUI – *Organizationally Unique Identifier*. Výjimka z OUI je *Individual/Group* bit, která určuje zadanou adresu jako skupinovou (I/G=1) nebo jako adresu jedné stanice (I/G=0). Speciálním případem je skupinová adresa, která adresuje všechny stanice na dané síti, obsahuje samé 1, tedy `FF-FF-FF-FF-FF-FF`, nazývá se *broadcast*. Dále se v OUI nachází *Universal/Local* bit, jež oznamuje, zda-li je adresa přidělena Univerzálně (U/L=0), nebo je přidělení lokální (U/L=1), jako příklad lokálního přidělení je také výše zmíněný broadcast.

Dalším příkladem přepínaných sítí jsou sítě telefonní – PSTN[1]. Zde jsou to pak telefonní čísla podle formátu mezinárodního telekomunikačního sdružení ITU E.164. Pomocí této adresy se vnitřní přenosové uzly rozhodují, kam dané informace přepínat.



Obrázek 3.1: Struktura MAC adresy, přepracováno z [20]

Přepínané sítě lze rozdělit na dva základní způsoby přenosu informací, a to sítě s přepínáním okruhů a přepínáním rámců. V síti s přepínáním okruhů se před vlastním přenosem informace ustaví "okruh" – cesta, přes kterou bude informace přepínána. Definice okruhů jsou buďto manuálně administrátorem, nebo dynamicky. Zástupcem jsou již zmíněné telefonní sítě – PSTN. Přepínání rámců naopak přepíná "in situ" podle cílové adresy a podle

[1] PSTN – Public Switched Telephone Network

momentálních znalostí o umístění jednotlivých cílových stanic. Při výpadku linek je v případě přepínání okruhů nutno navázat nový okruh, pokud je to možné. U přepínání rámců dojde k ustavení nové topologie a naučení nové polohy cíle.

3.1 Základní přepínání rámců

Předchůdci dnešních přepínačů byly síťové mosty a rozbočovače (Bridge[2], Hub), vykonávaly základní šíření zpráv mezi jednotlivými segmenty sítě. Zprávu, jež přijali na jednom rozhraní (portu), rozšířily na všechna ostatní připojená rozhraní.

Tento způsob šíření zpráv není příliš efektivní, a proto bylo k němu přidáno inteligentní přepínání, na základě postupně naučených adres. Tím vznikly tzv. "Learning" Bridge (Hub). Nakonec byl přidán Spanning Tree Protocol (vizte dále) a tím vznikl "Complete" Bridge – přepínač (Switch).

Přepínání tedy probíhá na základě znalosti polohy cíle vzhledem k danému uzlu. V případě neznámé cílové adresy je tato informace jednoduše šířena všemi směry – *broadcast*, kromě směru, ze kterého byla přijata. I v případě známé adresy uzel informaci nešíří směrem, z něhož ji přijal. Pokud by tam cíl opravdu byl, již informaci přijal (například sdíleným médiem).

Učení cílových adres ve vnitřním uzlu probíhá dle zdrojové adresy a linky (portu), na kterém byla zpráva přijata. Pokud tedy tímto portem přišla zpráva od koncového uzlu, lze stejným směrem odeslat data určená pro něj.

Názorně toto vysvětluje následující příklad na 3.2:

(a) Host A odesílá zprávu (rámec) pro Host B, vyplní tedy cílovou adresu Host B[3]a zdrojovou Host A a zprávu odešle.

(b) Tato zpráva dorazí na připojený přepínač, ten se podívá do své tabulky naučených adres, kde se nachází Host B. Adresu v tomto okamžiku nezná, proto danou zprávu šíří na všechna ostatní rozhraní.

(c) Při té příležitosti si poznamená, že Host A se nachází na portu 1 (ze zdrojové adresy a portu, kterým přijal danou zprávu).

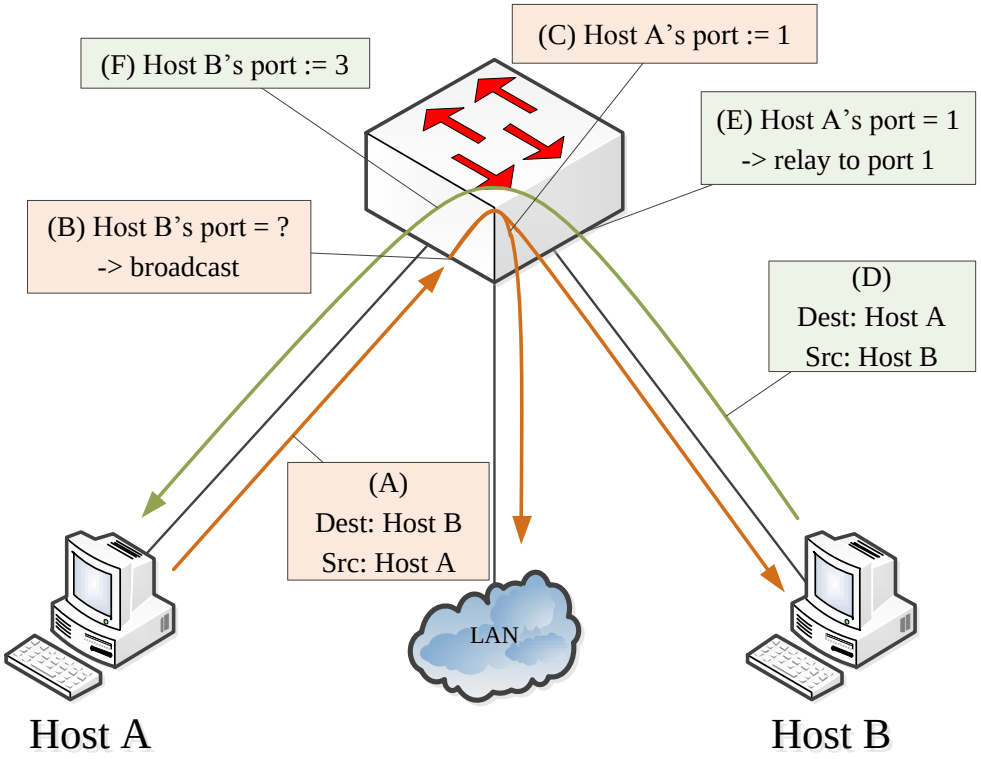
(d) Host B se rozhodne odpovědět na zprávu, odesílá zprávu s cílovou adresou Host A a zdrojovou Host B. Odpověď dorazí na přepínač, jež v tabulce adres má již záznam. Dle něj pošle zprávu jen na daný port 1, pro Host A.

(e) Dále si poznamená i polohu Host B a to na portu 3. Nyní i zprávy pro Host B, budou posílány jen na port 3.

Jednoduše lze pozorovat, že pokud je v topologii více přepínačů, každý se postupně učí ze zdrojových adres. Každý si tedy postupně poznamená port, kterým přijal zprávu některé ze stanic a tedy ví, kudy odeslat odpověď.

[2]Pojem *Bridge* se v terminologii standardů zachoval až dodnes, lze se s ním setkat právě i ve standartu [14, 15, 16], které popisují dnešní přepínače, ale z pohledu modelů funkčnosti jsou to stále mosty.

[3]Pro jednoduchost předpokládejme, že stanice znají navzájem své adresy.



Obrázek 3.2: Ilustrace průběhu přepínání rámců

3.2 Přepínání virtuálních lokálních sítí (VLAN)

Virtuální lokální síť – VLAN je logicky oddělená přepínaná síť linkové vrstvy ISO/OSI modelu, která koexistuje s ostatními VLAN sítěmi na stejné sdílené fyzické topologii. Dle standardu IEEE 802.1Q se tyto oddělené sítě označují identifikátorem – VID, které je v dané LAN unikátní a může nabývat hodnot 1-4094. Koncové uzly spadají právě do jedné VLAN dle portu, na kterém jsou připojeny, ale na přenosové sítí můžeme využít sdíleného vedení, je nutné rámec označit číslem VLAN, aby protější uzel věděl, do které VLAN daný rámec patří. Rozložení jednotlivých virtuálních sítí nastavuje administrátor.

Pro značení VLAN byl upraven formát rámce. Standardní Ethernet rámec je složen:

Preamble – slouží ke správné detekci začátku rámce, obsahuje střídající se 0 a 1.

Destination Address – MAC adresa cíle.

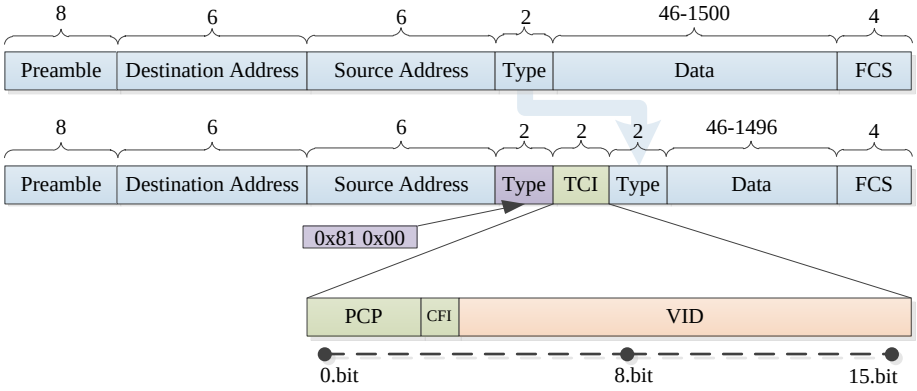
Source Address – MAC adresa zdroje.

Type – typ následujících dat.

Data – samotná data, dle velikosti MTU, pro ethernet až 1500.

FSC – Frame Check Sequence – kontrolní součet (CRC) o velikosti 4 Byte

Rámec označený dle IEEE 802.1Q, tzv. tagged, má vloženy 2 přídavné pole:



Obrázek 3.3: Struktura Ethernet rámce, bez (untagged) a s VLAN označením (tagged), vypracováno z [20, 16]

Type – na stejném místě jako v případě neznačeného rámce s hodnotou `0x81 0x00`, která oznamuje, že daný rámec je tagged.

TCI – Tag Control Information – samotné označení 802.1Q, skládá se z:

PCP – Priority Code Point – priorita, jež se regeneruje z původní priority rámce

CFI – Canonical Format Indicator – umožňuje použít jiné přístupové metody k médiu uvnitř jedné sítě.

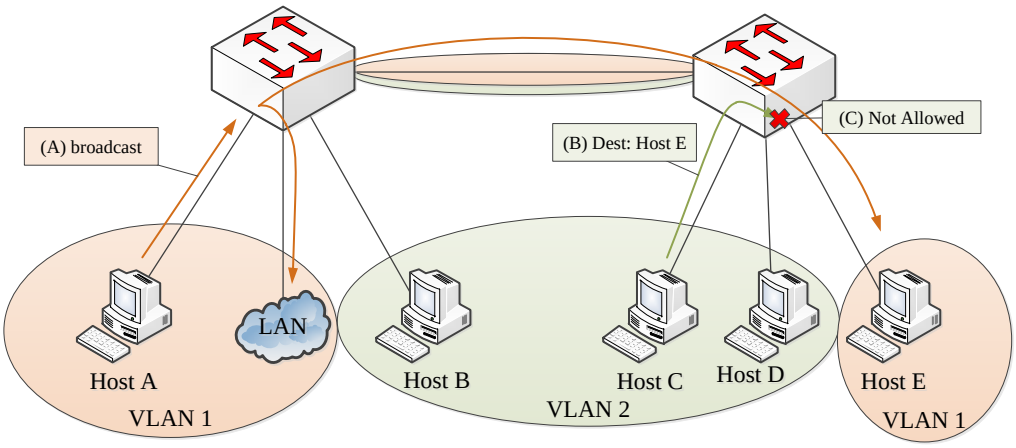
VID – VLAN Identifier – číslo dané virtuální sítě (1-4094), speciální hodnota 0, udává, že rámec je pouze *Priority-Tagged*, jelikož dle standardu není označení VLAN povinné. A dále hodnota 4095 je rezervovaná.

Původní Type a Data se posunou a kontrolní součet se přepočítá dle změněného rámce.

Přepínání v samotných VLAN pracuje na stejném principu jako základní přepínání rámců, rozsah přepínání je dán nastavením jednotlivých VLAN. Tedy komunikace mezi dvěma oddělenými sítěmi je možná jen za použití směrování. Použitím VLAN se zvyšuje segmentace sítě, to vede k lepší spravovatelnosti a škálovatelnosti. Díky rozdělení na logické celky se snižuje velikost broadcast domén a tedy i celkový rozsah provozu, například ARP dotazů[4]. Funkci přepínání rámců ve VLAN síti znázorňuje následující 3.4:

(a) Host A odesílá broadcast, tedy cílová adresa je nastavena na FF-FF-FF-FF-FF-FF. Zpráva se šíří do celé lokální sítě, dle základní fuknce přepínání. Ale vzhledem k tomu, že síť je rozdělena na Virtuální LAN sítě, šíří se jen ve své VLAN 1. Je to broadcast, je tedy odeslána na všechna dostupná rozhraní patřící do VLAN 1.

[4] ARP – Address Resolution Protocol je metoda získání odpovídající MAC adresy cíle, ke známe IP adrese



Obrázek 3.4: Příklad ilustrující přepínání rámců v síti s VLAN

(b) Host C zná MAC adresu Host E a rozhodne se mu poslat zprávu. Vyplní tedy cílovou adresu Host E. Poté, co zpráva dorazí na přepínač, vyhledá v tabulce záznamů cílových adres port, kterým má být zpráva odeslána – port 3.

(c) Poté zkontroluje příslušnost portu 3 ke zdrojové VLAN 2, což nesouhlasí, proto zprávu zamítne. Tím je zajištěno logické oddělení jednotlivých virtuálních sítí.

4 Redundatní topologie

Informace o základní variantě *Spanning Tree* Protokolu jsou vypracovány ze standardu [14, 15, 20], doplňující informace jsou čerpány z [3].

Mechanismus přepínání rámců neobsahuje žádnou metodu stárnutí rámců (např. TTL na síťové vrstvě) nebo jiný mechnismus pro předcházení smyčkám. Přepínané sítě jsou často stavěny jako redundantní topologie, abychom zvýšili odolnost vůči poruchám. Pokud v přepínané síti existují smyčky, může nastat stav zahlcení všesměrovými informacemi – *broadcast storm*. Ten nastává, pokud existuje v topologii cyklus, přes který se začné šířit všesměrová informace (broadcast). V tu chvíli ji uzly, ležící v cyklu, posílají na všechna ostatní rozhraní (viz sekce 3.1).

4.1 Spanning Tree Protocol

Proto potřebujeme efektivní mechanismus ovládání těchto topologií. Dle standardu [15], který mimo jiné specifikuje chování přepínaných sítí, dále definuje mechanismus obsluhy redundantních topologií. Tímto mechanismem je *Spanning Tree Protocol* – STP. Byl navržen v roce 1990 s první verzí IEEE 802.1D-1990 [NON VIDI]. Jeho základní funkcí je "naučit se" topologii dané lokální sítě, zvolit jeden z uzlů kořenem a od něj poté vytvořit aktivní topologii bez cyklů – kostru. Ta je poté použita k přepínání. Při výpadku linky nebo poruše některého zařízení, přehodnotí topologii a vytvoří novou, takže komunikace dále pokračuje po funkční části sítě.

4.1.1 Identifikátory a ceny

Volba kořene probíhá na základě idenfikátoru uzlu – *BridgeID*, ten je složen z MAC adresy zařízení a upravitelné priority. Uzel s nejlepším identifikátorem je vybrán jako kořen. Lepší identifikátor má nižší hodnotu.

Pokud do jednoho segmentu vede cesta přes dva uzly, rozhoduje se pomocí jejich *BridgeID* mezi nimi. Ten s lepší hodnotou přepne svůj port, jež je spojením do daného segmentu do role *Designated* (vizte sekce 4.1.2).

V případě, že do jednoho segmentu existují z daného uzlu dvě linky, existuje dodatečné uspořádání pomocí identifikátoru portu – *PortID*, jež se skládá z unikátního čísla portu v rámci zařízení a upravitelné priority. Obě zmíněné priority mají přednost před zbývající části identifikátoru.

Dalším parametrem je cena linky, kterou jsou spojena jednotlivá zařízení. Nejvýhodnější je mít kostru topologie složenou s nejlepších možných linek, jež se v síti vyskytují. Proto se kostra určuje od kořene, s přihlédnutím k ceně jednotlivých linek. Cena se určuje z přenosové rychlosti linky podle dané tabulky, která tyto ceny specifikuje. Každý další uzel dostane z okolí zprávu, jak „drahá“ je cesta přes sousedy ke kořeni a rozhodne se pro nejmenší.

Parametr	Implicitní hodnota	Dovolený rozsah
Priorita uzlu (*BridgeID*)	32768	0-65535
Priorita portu (*PortID*)	128	0-255

Tabulka 4.1: Tabulka udávající parametr priority uzlu a portu, zdroj [14, str. 109, tab 8-4]

V případě shodné ceny 2 linek se provádí dodatečné rozhodnutí pomoci *BridgeID* případně *PortID* uzlů, které tyto zprávu šíří. Pokud nelze ani pomocí těchto dodatečných hodnot rozhodnout, porovnává se lokální *PortID* portů, které zprávy přijaly. Uspořádání s použitím všech identifikátorů je úplné, protože *PortID* jsou unikátní v rámci zařízení. Dovolený rozsah cen linek, nezávisle na jejich rychlosti, je 1-65535.

Rychlost linky	Doporučená hodnota	Doporučený rozsah
4 Mb/s	250	100-1000
10 Mb/s	100	50-600
16 Mb/s	62	40-400
100 Mb/s	19	10-60
1 Gb/s	4	3-10
10 Gb/s	2	1-5

Tabulka 4.2: Tabulka určující ceny linek v závislosti na jejich rychlosti, zdroj [14, str. 109, tabulka 8-5]

4.1.2 Role a stavy portů

Spanning Tree protokol přiděluje jednotlivým portům stavy, které definují chování portu vzhledem k přepínání zpráv. Dále existují role portu, jež reprezentují úlohu portů v topologii, ačkoliv byly definovány až v RSTP (Uvádím je již zde). Ve standardu IEEE 802.1D-1998[1]je to jen pojmenování portů, které splňují následující podmínky, pro ilustraci jsou vyobrazeny na obrázku 4.1.

Disabled – port se neúčastní aktivní topologie.

Root – port s touto rolí obdržel v procesu ustavení stromu *Configuration BPDU* s nejlepším *Root Identifier* a cenou linky – *Root Path Cost* je spojen „nejlevnější“ cestou s kořenem.

Designated – tento port spojuje odlehlejší segmenty sítě se stromem, směřuje tedy od kořene ke koncovým uzlům. Když při ustavování stromu přijal horší hodnotu *Root Identifier*, je tedy připojen do okrajového segmentu. Je-li *Root Identifier* shodná s hodnotou na *Root* portu, byla přijata (od jiného uzlu, který není kořenem) s vyšší cenou cesty. *Root Bridge* má všechny porty v roli *Designated*, vzhledem k nejlepšímu *BridgeID*.

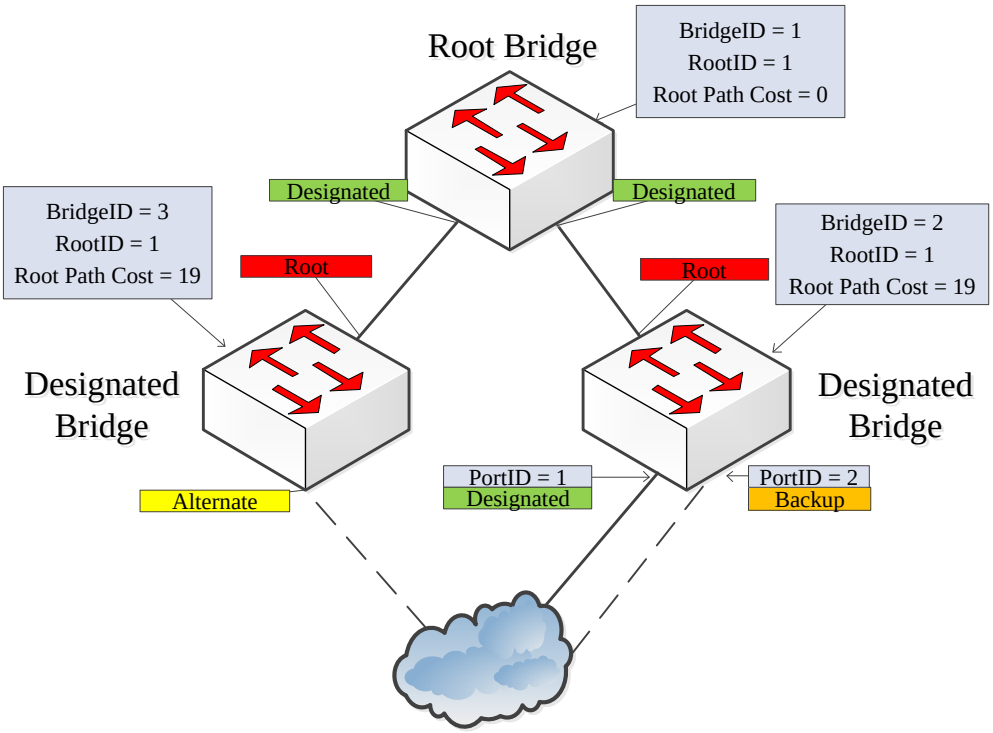
Alternate – poskytuje „horší“ spojení do segmentu, který je již spojen s aktivní topologií pomocí *Designated* portu na jiném uzlu. To bylo rozhodnuto porovnáním přijatých

[1] v STP byly použity jen Root a Designated

identifikátorů a ceny cesty ke kořeni, vizte sekce 4.1.1. *Alternate Port* identifikuje linky, které dle daného nastavení jsou hranami kružnic[2].

Backup – platí pro něj stejné podmínky jako pro *Alternate Port*, ale Designated Port, který spojuje daný segment je na stejném uzlu. *Backup Port* identifikuje linky, které dle daného nastavení jsou hranami smyček[3].

Poznámka: Pokud v dalším výkladu bude použitý výraz „kružnice“ nebo „smyčka“, je míněno obojí.



Obrázek 4.1: Znázornění určení rolí jednotlivých portů v závislosti na parametrech uzlů, přepracováno z [2]

Stavy portů se následně odvíjí od rolí portů. Role *Alternate* a *Backup* jsou ve stavu *Blocking*, kdežto role Root a Designated si projdou postupně všemi stavy, až se dopracují do stavu Forwarding tak, jak je naznačeno na obrázku 4.2.

Disabled port se neúčastní procesu přepínání rámců, není aktivní. Do tohoto stavu port přechází kdykoliv, je to administrátorem vynuceno – vypnutím portu nebo odpojením

[3] [19, str. 8] „Uzavřená cesta se nazývá kružnice. Orientovaná kružnice se nazývá cyklus.“
[3] [19, str. 6] „Hrana (u, u) se nazývá smyčka.“

kabelu. Pokud je možno detekovat závadu na portu, přechází port také do stavu disabled.

Blocking V tomto stavu se port také neúčastní procesu přepínání, aby mohly být potlačeny existující kružnice v topologii. Ale v tomto stavu jsou zpracovávány přijaté BPDU zprávy pomocí *Spanning Tree* protokolu. Tento stav je nastaven všem portům po inicializaci uzlu a také pokud je port administrátorem aktivován, přechází tedy ze stavu *Disabled*. Dále port může do stavu *Blocking* přejít ze zbylých stavů pomocí procesu STP, pokud přijal ze segmentu, do kterého je připojen, informaci o tom, že jiný ("lepší") uzel již spojuje daný segment se stromem. Pokud vyprší časovač na některém portu tohoto uzlu, který hlídá "živost" okolních uzlů, může port přejít do stavu *Listening*.

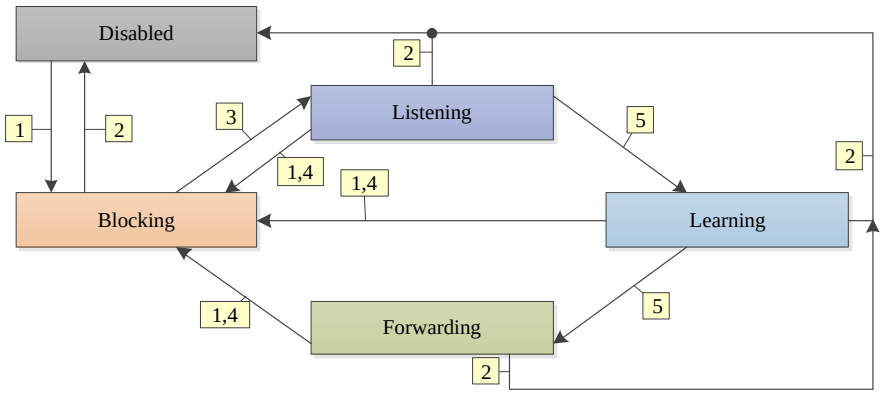
Listening Zde se port připravuje k účasti na procesu přepínání rámců, avšak samotné přepínání je stále dočasně neaktivní, aby se potlačily možné existující dočasné kružnice. Ty mohou vzniknout při přechodech portů do různých stavů v procesu ustavení aktivní topologie. Tímto je vyhrazen čas na ustálení ostatních portů na všech uzlech. Proces učení polohy stanic je neaktivní na tomto portu, protože aktivní topologie se právě mění. Tím by mohlo dojít k situaci, že nově naučené informace by po ustálení mohly být špatné. Přijaté BPDU zprávy jsou zpracovány procesem STP a port je zahrnut do výpočtu aktivní topologie. Portem ve stavu *Listening* je možné v případě potřeby odesílat BPDU zprávy. Do následujícího stavu, tedy *Learning*, se přechází po vypršení časovače v procesu STP. V případě, že uzel přijme zprávu, která hodnotami na jiném portu nahrazuje port v *Listening* stavu, může tento přejít zpět do stavu *Blocking*.

Learning i v tomto stavu se port připravuje k přepínání, ale to je stále neaktivní kvůli možné existenci dočasných kružnic v topologii. Přijaté BPDU jsou také zpracovávány stejně tak, jako mohou být odesílány. Ale proces učení polohy cílových stanic již aktivní je. Díky tomu je možné, ještě před samotným spuštěním aktivní topologie, naučit se mnoho informací o stanicích. To následně redukuje množství informací, jež by se musely poté přepínat všesměrově. Po vypršení časovače v procesu *Spanning Tree* tento port přechází do stavu *Forwarding*. Stejně tak, jako předchozí stav, může přejít do stavu *Blocking* nahrazením lepším portem a dále do stavu *Disabled*, jak je popsáno výše.

Forwarding zde se již port účastní přepínání rámců tak, jak je popsáno v předchozí kapitole. Také musí zpracovávat přijaté BPDU a odesílat aktualizované. Přechod do stavu Blocking a Disabled zůstává nezměněn.

Popis obrázku 4.2:

1. Port je aktivován, např. připojením kabelu.
2. Deaktivace daného portu, například vypnutí portu administrátorem.
3. Algoritmus vybral port jako *Designated* nebo *Root*
4. Algoritmus vybral port jako *Alternate/Backup*, byla přijata lepší hodnota z jiného uzlu/portu
5. Vypršení časovače *Forwarding Delay*, přechod do dalšího stavu (blíže k Forwarding)



Obrázek 4.2: Přechodový diagram pro změnu stavu portu, přepracováno z [14, str. 64, obrázek 8-3]

4.1.3 Časovače

Časovače jsou základním ovládacím prvkem *Spanning Tree* protokolu. Zajišťují správné přechody mezi stavy tak, aby nevznikaly dočasné smyčky. Díky nim je možno detekovat přerušení komunikace v aktivní topologii. A nakonec zabraňují nekonečnému přeposílání zastaralých informací.

Parametr	Doporučená hodnota	Dovolený rozsah
Hello Timer	2	1-10
Max Age	20	6-40
Forward Delay	15	4-30
Hold Time	1	1

Tabulka 4.3: Hodnoty časovačů pro Spanning Tree Protocol dle [14, str. 108, tabulka 8-3]

Hello Timer – interval, ve kterém uzel, jež se považuje za kořenový, odesílá konfigurační zprávy BPDU. V ustavené aktivní topologii zasílá konfigurační BPDU jen kořen, právě v intervalu *Hello Timer*.

Max Age – udává maximální stáří BPDU informací zjištěných portem, po kterém jsou zahazovány. Zajišťuje možnost rekonfigurace. Pokud do jeho vypršení na port nepřijde aktualizovaná informace, se sousedním uzlem něco není v pořádku a může převzít jeho roli[4]. Jeho hodnota je přijata od kořenového uzlu.

Forward Delay – Hodnota, způsobující přechody stavů portů, zajišťuje obranu proti dočasně vzniklým smyčkám vlivem rekonfigurace aktivní topologie. Hodnota je také přijata od kořenového uzlu.

Hold Time – Pevně zadaná hodnota. Je minimálním intervalem, v kterém mohou být odesílány BPDU zprávy. Slouží k zajištění nízké spotřeby přenosového pásma pro komunikaci *Spanning Tree* protokolu.

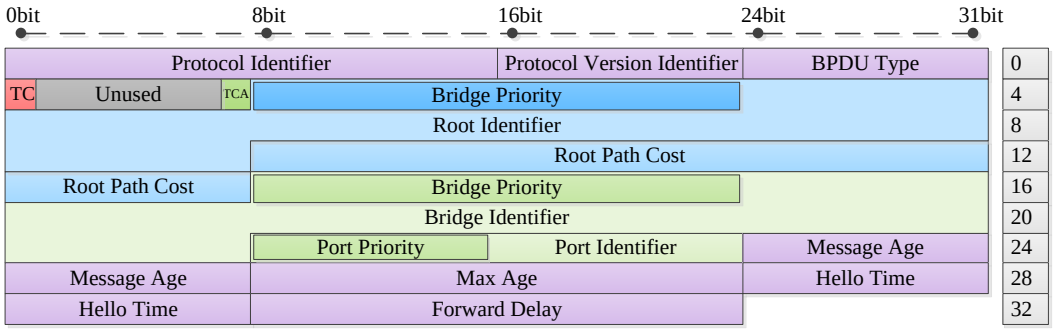
A navíc by měl uzel splňující standard IEEE 8021.D-1998 vyžadovat vztahy mezi časovači vyjádřené následujícími vzorci. Kde (4.1a) zjišťuje dostatečné zpoždění přechodů stavů portů (*Forward Delay*) nad vypršením starých informací na portech *MaxAge*, aby nedošlo k ustavení aktivní topologie, která by mohla obsahovat kružnice. Další nerovnice (4.1b) říká, že pravidelné vysílání BPDU musí být provedeno více než dvakrát v intervalu *MaxAge*. Mohlo byt dojít ke zbytečné rekonfiguraci po vypršení informací na portech, pokud by se některé BPDU při přenosu ztratily. Vzorce lze nalézt v [14, str.108].

$$2(\text{ForwardDelay} - 1.0\,s) \geq \text{MaxAge} \tag{4.1a}$$

$$\text{MaxAge} \geq 2(\text{HelloTime} + 1.0\,s) \tag{4.1b}$$

4.1.4 Šíření informací přepínačů

BPDU – *Bridge Protocol Data Unit* jsou zprávy určené pro šíření informací mezi přepínači IEEE 802.1D a dále. Pomocí nich se vyměňují informace *Spanning tree protokolu*, jako identifikátory nebo ceny linek. *Configuration BPDU* má následující formát:



Obrázek 4.3: Struktura *Configuration BPDU* rámce, přepracováno z [14, str. 112, obrázek 9-1]

Protocol identifier – obsahuje hodnotu 0 pro identifikaci STP.

Protocol Version Identifier – verze také obsahuje hodnotu 0.

BPDU Type – typ BPDU rámce, hodnota 0.

Flags:

Topology Change Flag – určuje změnu topologie

[4]Pokud je více takových, dohodnou se poté pomocí BPDU, jak je popsáno v sekci o šíření informací, sekce 4.1.4

Unused – není použito, hodnota 0

Topology Change Acknowledgement – potvrzuje provedení změny topologie

Root Identifier – identifikátor uzlu, který považuje odesílatel BPDU za kořen (dále také Root Bridge), tzn. nejlepší známý uzel. Po inicializaci je to uzel sám, který považuje se za kořen.

Root Path Cost – dosavadní akumulovaná cena k nejlepšímu známému kořenu, je uvedena ve zprávě BPDU. Pro kořenový uzel samozřejmě 0, každý další přičítá cenu linky, přes kterou bylo BPDU přijato.

Bridge Identifier – je to identifikátor zdroje BPDU zprávy. Tím se identifikují sousední uzly. Díky tomuto identifikátoru se také rozhoduje tzv. *Designated Bridge*[5]pro daný segment, pokud jsou do něj připojeny dva a více uzlů.

Port Identifier – je identifikátor portu zdroje BPDU zprávy, kterým byla BPDU odeslána. Umožňuje zvolit jeden z portů uzlu, jež jsou připojeny do shodného segmentu, který bude spojovat daný segment se stromem, tzv. *Designated*.

Message Age – určuje stáří BPDU zprávy. Slouží pro zamezení šíření příliš starých informací. Zvyšuje se na každém uzlu.

Max Age – je nejvyšší tolerované stáří zprávy a informací na portech. Po překročení se BPDU zahazuje. Hodnotu nastavuje kořenový uzel.

Hello Time – udává hodnotu časovače Hello Timer (viz dále), jež je nastavena na odesílajícím uzlu. Toto pole BPDU slouží pro detekci rozdílných a potenciálně chybných konfigurací časovačů.

Forward Delay – udává nastavení hodnoty na odesílajícím uzlu, a to časovače *Forward Delay*. Kořenový uzel posílá hodnotu Forward Delay, jež budou používat všechny uzly.

BPDU zprávy přepínač nepřeposílá, ale vždy zpracuje a své poznatky šíří dále. Zprávy mají odlišné hodnoty v závislosti na odesílacím portu (vizte *Port Identifier*).

Způsob šíření informací mezi uzly se odvíjí od jejich rolí a rolí jejich portů. Názorně popisuje obrázek 4.4 a jeho pravila jsou:

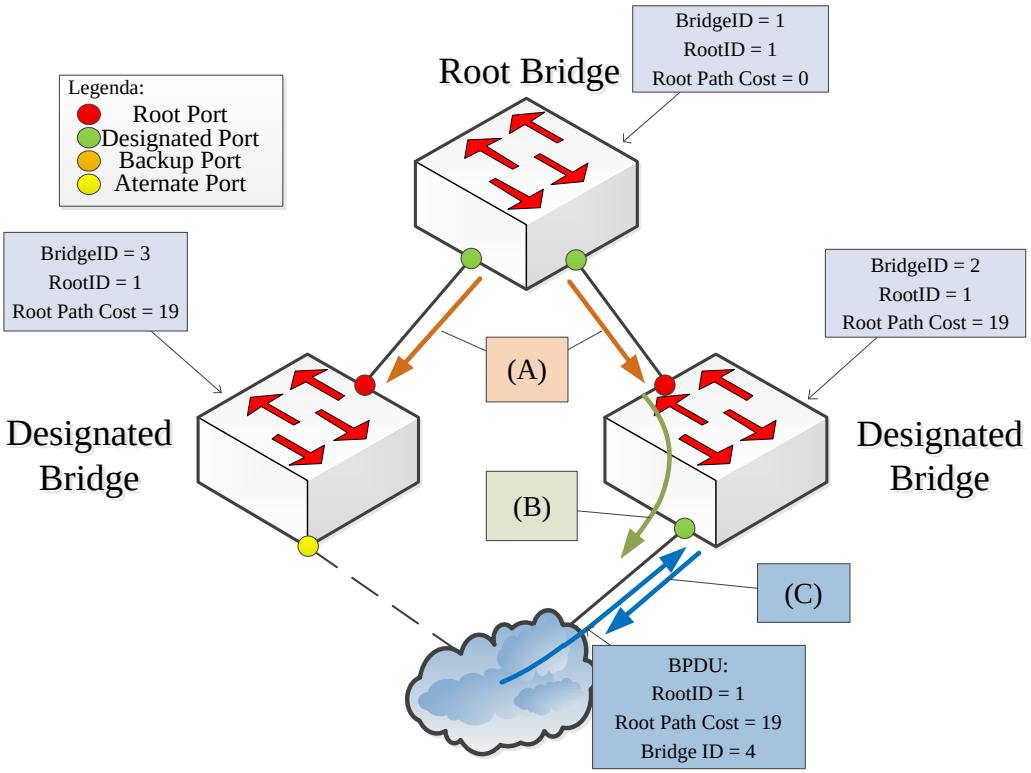
(a) Uzel, který se považuje za kořenový (při inicializaci každý), začíná šířit konfigurační BPDU zprávy do všech k němu připojených segmentů – všemi porty.

(b) Pokud uzel přijme konfigurační BPDU na svém Root portu a tato zpráva obsahuje nejlepší identifikátory (*Root Identifier*, *Root Path Cost*, *Bridge Identifier*, *Port Identifier*), postoupí zprávu do všech segmentů, kde věří, že zastává roli *Designated Bridge*. Tedy všemi aktuálními *Designated* porty.

(c) Nakonec, jestliže obdrží BPDU zprávu s horšími identifikátory na *Designated* portu, odpovídá svou lepší zprávou. Zprávu posílá všem uzlům, které jsou připojeny do daného segmentu, aby je uvědomil o tom, že je pro tento segment *Designated Bridge*.

BPDU zprávy se posílají v ethernetových rámcích s univerzální 48b adresou, nazývanou *Bridge Group Address*, která byla pro tento účel vyhrazena. Tabulka 4.4 udává vyhrané adresy dle standardu IEEE 802.1D-1998.

[5]Designated Bridge – uzel, který daný segment spojuje designated porty se stromem.



Obrázek 4.4: Příklad ilustrující šíření informací přepínačů pomocí STP

Přidělení	Adresa
Bridge Group Address	`01-80-C2-00-00-00`
IEEE Std. 802.3x Full Duplex PAUSE operation	`01-80-C2-00-00-01`
Reserved	`01-80-C2-00-00-02 ... 0F`

Tabulka 4.4: Rezervace MAC adres dle [14, str. 52, tabulka 7-9]

Při využití všech polí se jedná o *Configuration BPDU*. K ní STP používá navíc *Topology Change BPDU*. Konfigurační zpráva je základní jednotkou k vyjednání kořene, ustavení aktivní topologie a pro kontrolu funkce sousedních uzlů. Naproti tomu BPDU, informující o změně topologie, se používá, pokud se v již existující topologii něco změnilo a je třeba ji jen upravit.

Rozdílnou hodnotou (`0x40` = `1000 0000`) se plní pole *BPDU Type*, které říká, že se jedná právě o *Topology Change BPDU*.

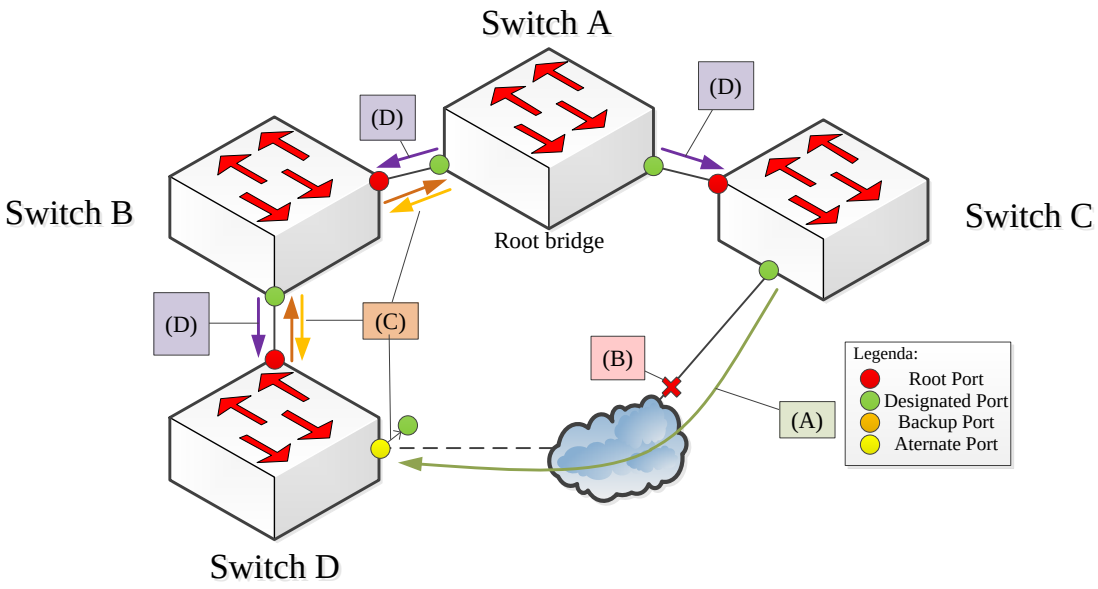
Změna topologie musí nastat v případě, že na některém z portů vypršel časovač *Max Age*. Což značí výpadek sousedního *Designated Bridge* pro daný segment. Při detekci výpadku se port pokusí převzít úlohu pro daný segment. Pokud je port, jež detekoval výpadek *Root Portem*, zvolí za něj jiný port, z přijatých BPDU a pravidel o volbě rolí. Pokud takové informace nedorazily odjinud, je pravděpodobně nejlepším následníkem a pokusí se převzít roli kořenového uzlu. V případě výpadku Root portu, je nutno přepočítat topologii.

Obrázek 4.5: Struktura *Topology Change BPDU* zprávy, přepracováno z [14, str. 113, obrázek 9-2]

Pokud se však jednalo o převzetí role *Desinated*, tento uzel vysílá svým root portem (tedy směrem ke kořeni) *Topology Change Notification BPDU* do té doby, něž mu následující uzel potvrdí přijetí. To učiní odesláním *Konfiguračním BPDU* s nastaveným parametrem *Topology Change Acknowledge*. Pokud některý z uzlů přijme *Topology Change Notification BPDU*, šíří ho dále ke kořenovému uzlu stejným způsobem. Tedy čeká na potvrzení.

Ve chvíli, kdy zprávu o změně topologie obdrží kořen, potvrdí přijetí a po určitou dobu vysílá *Konfigurační BPDU* s nastaveným parametrem *Topology Change*. Tím dává všem najevo, že někde došlo ke změně topologie. Může dojít ke změně „polohy“ stanice vzhledem k nové aktivní topologii. Proto se položky v tabulce adres stávají zastaralými. Uzly, když obdrží parametr *Topology Change* nastavují kratší dobu stárnutí v tabulce adres. Díky tomu rychleji zestárnou položky, jež se vlivem nové aktivní topologie *přesunuly*.

Názorně proces změny topologie ilustruje následující obrázek 4.6:



Obrázek 4.6: Ukázka průběhu změny topologie STP

(a) Stav, kdy je vše v pořádku – *Designated Bridge* pro daný segment přeposílá *Configuration BPDU* od kořene. Tím obnovuje časovač *Max Age* na ostatních přepínačích, jež sledují, zda je vše v pořádku.

(b) Zde nastal problém na lince, která spojuje daný přepínač se segmentem, ale ten si toho není vědom.

(c) Na sousední přepínač nebyly dlouho doručeny *Konfigurační BPDU*, a proto vypršel časovač. Přepínač se pokusí stát *Designated* pro tento segment. Jelikož je jediný (případně má nejlepší *BridgeID*), povede se mu to. Poté odesílá TCN BPDU a čeká na potvrzení od nadřazeného přepínače. To přijímá. Totéž provede každý další, až se zpráva dostane ke kořenovému přepínači.

(d) Ten reaguje na změnu topologie tak, že po určitou dobu rozesílá *Konfigurační BPDU* s nastaveným parametrem *Topology Change* := 1. Tím ostatní přepínače ví, že mají nechat stárnout položky v tabulce adres rychleji, aby aktualizovaly polohy cílových stanic.

4.2 Rapid Spanning Tree Protocol

Dle standardu IEEE 802.1D[6]z roku 2004 ([15]), je STP nahrazeno vylepšenou variantou s rychlejší dobou konvergence – Rapid Spanning Tree Protocol. Byl navržen tak, že ponechává mnoho parametrů nezměněných a funkčně vychází z původního STP. Díky tomu je schopen v případě potřeby spolupracovat i s přepínači, co neumí RSTP.

Informace k této sekci jsou čerpány přímo ze standardu a dále doplňovány z [2, 3].

4.2.1 Identifikátory a ceny

Identifikátory jednotlivých přepínačů a portů zůstávají zachovány. Nebylo je třeba měnit, uspořádání je dostatečné. Ale je důležité, že RSTP zavádí omezení v nastavení priorit uzlů a portů. Rozsahy jsou sníženy a každá z nastavených hodnot musí být dělitelná definovaným číslem, tedy je nastavována s takovým krokem. vizte tabulka 4.5:

Parametr	Implicitní hodnota	Dovolený rozsah
Priorita uzlu (*BridgeID*)	32768	0-61440, násobky 4096
Priorita portu (*PortID*)	128	0-240, násobky 16

Tabulka 4.5: Definice priorit uzlu a portu, dle [15, str. 153, tabulka 17-2]

Dále byly upraveny ceny linek. Jelikož technologie pokročily (přenosové rychlosti navýšily a jejich dostupnost je stále lepší), musely být provedeny jisté aktualizace hodnot. Vizte tabulka 4.6. Dovolený rozsah všech hodnot, nezávisle na rychlosti, je 1-200 000 000.

Vzhledem k vysokým hodnotám cen na nižších přenosových rychlostech není možné tyto hodnoty šířit na přepínače, jež podporují jen STP. Jejich hodnoty cen jsou omezeny v rozsahu 1-65535. Proto, z důvodu kompatibility, jsou pro komunikaci s STP použity hodnoty 65535, pro vyšší odpovídající hodnoty cen v RSTP. Případně je vhodné toto administrativně přenastavit, aby ceny odpovídaly poměrově přenosovým rychlostem na dané síti. Při použití hodnoty 65535 se totiž linka o rychlosti 1 MB/s a 100 MB/s chová v rámci STP naprosto stejně, což není vhodné.

[6]Verze z roku 1998 obsahuje dřívější verzi *Spanning Tree Protocol*, vizte [14]

Rychlost linky	Doporučená hodnota	Doporučený rozsah
≤ 100 Kb/s	200 000 000	20 000 000-200 000 000
1 Mb/s	20 000 000	2 000 000-200 000 000
10 Mb/s	2 000 000	200 000-20 000 000
100 Mb/s	200 000	20 000-2 000 000
1 Gb/s	20 000	2 000-200 000
10 Gb/s	2 000	200-20 000
100 Gb/s	200	20-2 000
1 Tb/s	20	2-200
10 Tb/s	2	1-2

Tabulka 4.6: Doporučné hodnoty a rozsahy cen linek v závislosti na jejich rychlosti, zdroj [15, str. 154, tabulka 17-3]

4.2.2 Role a stavy portů

Jak bylo zmíněno u rolí dřívějšího STP, byly byly definovány až s příchodem Rapid STP. Dříve to bylo v podstatě pojmenování portů podle toho, zda splnily podmínky (např. root směřuje ke kořeni, podle přijatých BridgeID ...). Role portů tedy zůstávají, jak jsou definovány výše (sekce 4.1.2).

Naproti tomu stavy portů prodělaly změnu. Z existujících pěti stavů byly tři sloučeny, vzhledem k velmi podobné funkci. Stavy Disabled, Blocking a Listening utvořily stav Discarding.

Discarding – procesu přepínání rámců se neúčastní a jak je naznačeno v názvu, rámce zahazuje. Je-li port aktivní (administrátor, připojený kabel), zpracovává přijaté BPDU, aby se v případě poruchy mohl začít účastnit aktivní topologie. Neaktivní je také učení polohy adres cílových stanic, protože port se nebude účastnit aktivní topologie, pokud nenastane změna a tedy by všechnu cílovou komunikaci zahazoval. Při inicializaci jsou všechny porty ve stavu *Discarding*. Do dalšího stavu *Learning* může přejít po změně role portu na *Root* nebo *Designated* po vypršení časovače *Forward Delay*. Tím se zabraňuje možným dočasným kružnicím (jako v STP). Nespornou výhodou však je exitence právě jednoho přechodu po *Forward Delay* časovači, kde v STP jsou právě dva.

Learning, Forwarding – tyto stavy zůstávají v podstatě stejné jako v STP, ale při změně přecházejí do stavu *Discarding*, aby předešly kružnícím v topologii. Přechod z *Learning* do *Forwarding* je také odpočítán časovačem *Forward Delay*.

4.2.3 Časovače

V Rapid STP je snaha zachovat časovače a jejich hodnoty z původního STP. Právě z důvodu kompatibility zůstavájí časovače *Hello Timer*, *Max Age*, *Forward Delay* a jejich hodnoty zmíněné výše.

Nově přibývá časovač *Migrate Time*, který slouží pro přechod k chování STP a tím zajišťuje spolupráci mezi přepínači bez RSTP. Dále slouží jako hodnota pro časovač, jenž detekuje *okrajové porty*. Ty nemají v připojeném segmentu další přepínač. Mohou tomu uspůsobit chování a vylepšit tak výkonnost protokolu. Hodnota časovače je v tabulce 4.7:

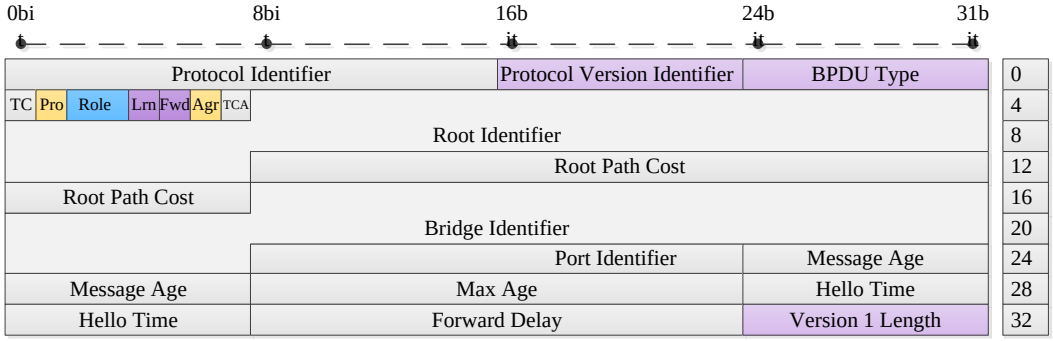
Parametr	Doporučená hodnota	Dovolený rozsah
Migrate Time	3	-

Tabulka 4.7: Definice nového časovače pro RSTP, dle [15, str. 153, tabulka 17-1]

Časovače v RSTP musí vyžadovat stejné vztahy hodnot mezi sebou jako v případě STP, vizte sekci 4.1.3, respektive vzorce 4.1.

4.2.4 Šíření informací

Z předchozí *Konfigurační BPDU* zprávy se vyvinulo RST BPDU. Byly přidány parametry do pole *Flags*, aby doplnily informace potřebné k novým metodám RSTP. Změnily se hodnoty verzí a typů zpráv, jež odráží provedené aktualizace kódování. A nakonec bylo přidáno pole, které udává délku informací předchozí verze. Změny ilustruje obrázek 4.7:



Obrázek 4.7: Struktura RST BPDU, přepracováno z [15]

Pole šedé barvy zůstávají zachována beze změn. A barevná pole jsou aktualizována takto:

Protocol Version Identifier – funkce zůstává, mění se verze na hodnotu 2.

BPDU Type – funkce také zachována, ale typ zprávy je hodnota 2, aby oznámil nové 36B kódování. STP přepínače přebytečná data ignorují.

Flags:

Proposal Flag – se nastavuje dle parametrů stavového automatu *Proposal Mechanismu*, tím se zajišťuje komunikace tohoto mechanismu mezi sousedícími uzly.

Port Role – role odesílajícího portu, hodnoty jsou definovány následovně:

- **00** – Neznámá role, tato hodnota by se při správné funkci neměla objevit. Ale přesto je přijímána a zpracována.
- **01** – *Alternate* nebo *Backup Port*.
- **02** – *Root Port*.
- **03** – *Designated Port*.

Learning Flag – udává, zda je na portu povolen proces učení adres stanic.

Forwarding Flag – udává, zda je na portu povolen proces přepínání rámců. Tento parametr společně s *Learning Flag* nepřímo oznamuje stav daného portu.

Agreement Flag – viz *Proposal Flag*.

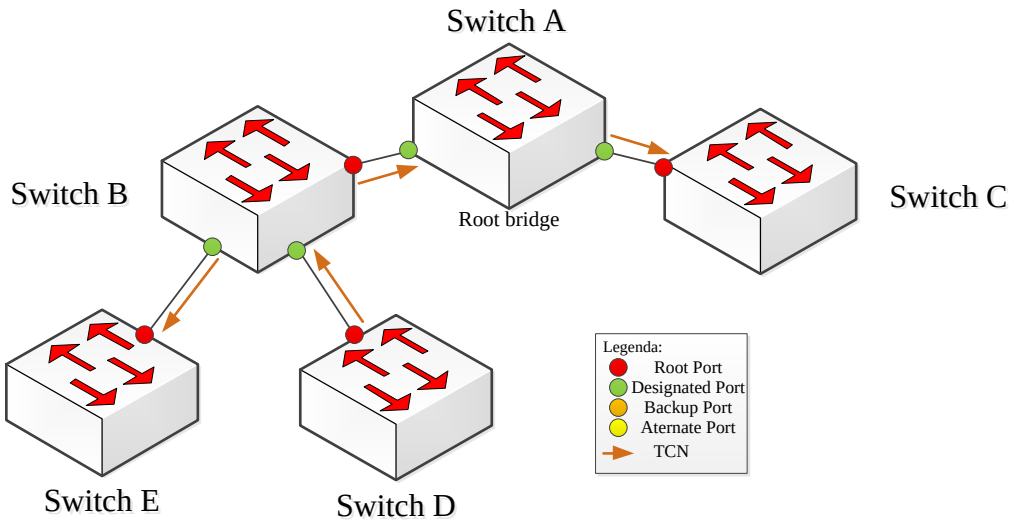
Version 1 Length – zde je vložena hodnota 0, která říká, že informace protokolu verze 1 nejsou přítomny.

Topology Change Notification BPDU zůstává zachována, vizte sekce 4.1.4 obrázek 4.5.

Stále jsou posílány jen dva druhy zpráv, a to *Configuration BPDU* a*Topology Change Notification BPDU*. V RSTP však může být *Configuration BPDU* posláno kódované jako původní nebo jako RST BPDU. A dále TCN BPDU může také být posláno jako původní nebo RST BPDU s nastaveným parametrem *Topology Change* (TC).

Pravidelné rozesílání prodělalo velikou změnu. Již neposílá pravidelné *Konfigurační BPDU* jen kořenový uzel, ale všechny. Při přijetí této zprávy ji nyní nešíří dále, protože každý uzel rozesílá vlastní. Díky této inovaci je mezi uzly možno použít obousměrný *keep-alive* mechanismus. Tedy, pokud některý z přepínačů nedostane tři po sobě jdoucí BPDU, považuje sousední přepínač za nedosažitelný a nastává úprava aktivní topologie. Tím je možné detekovat výpadky mnohem rychleji.

Topology Change Notification BPDU stále rozesílá přepínač, jěž změnil topologii. To se však nešíří postupně ke kořenovému uzlu a poté do zbytku sítě, ale postupně jej všechny uzly šíří v topologi. Tím se distribuuje TCN BPDU rychleji (může být kódováno jako RST BPDU s nastaveným TC). Ilustruje obrázek 4.8:



Obrázek 4.8: Příklad znázorňující funkci Topology Change v RSTP

4.2.5 Proposal/Agreement mechanismus

Rychlost konvergence může být ještě urychlena nezávisle na časovačích pomocí *Proposal/Agreement* mechanismu. Na portu, jež byl vybrán jako *Designated Port*, ale je ve stavu

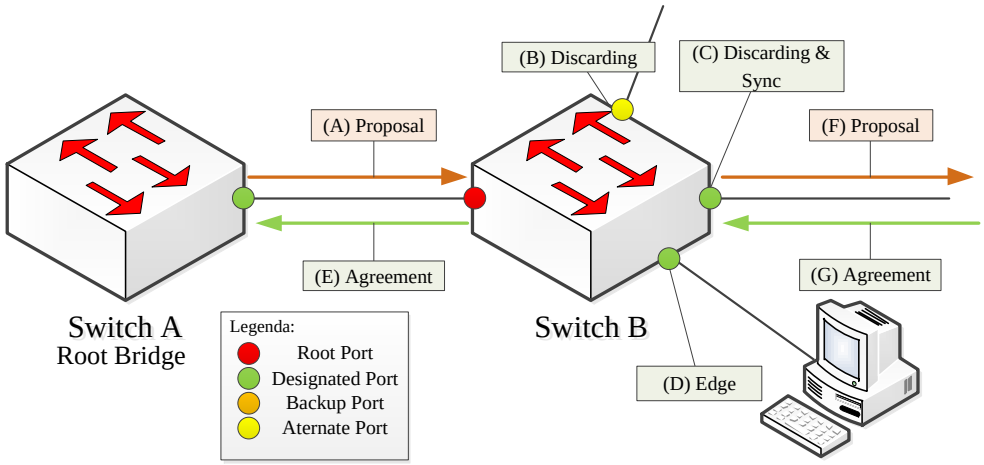
Discarding nebo *Learning*, se uzel pokouší o rychlý přechod pomocí tohoto mechanismu. Nastaví parametr *Proposal Flag* a odesílá RST BPDU. Poté, co mu přijde jako odpověď RST BPDU s parametrem *Agreement Flag*, může tento port urychleně přepnout do *Forwarding state*, protože je ujištěn, že sousední uzel se postará o to, aby nevznikly smyčky.

Přepínač, po přijetí žádosti – *Proposal* započne proceduru `Sync`. Ta zajišťuje, aby všechny porty měly povědomí o nových informacích , tj. od uzlu,který poslal žádost. Port je synchronizovaný pokud:

- je port blokovaný, tzn. je ve stavu *Discarding* ve stabilní topologii
- nebo je okrajovým portem, tj. nemá připojené další přepínače.

Pokud některý z portů není synchronizovaný, přepínač jej zablokuje a tím zesynchronizuje. Poté může odpovědet potrvzením – *Agreement* na žádost. Dále zvolí *Designated* porty (mohou zůstat stejné) a aplikuje mechanismus *Proposal/Agreement*, protože tyto jsou *Designated* a *Blocking/Learning*.

Tímto se urychleně ustaví topologie nezávisle na časovačích. Blíže ilustruje obrázek 4.9:



Obrázek 4.9: Znázornění funkce mechanismu *Proposal/Agreement*, přejato a upraveno z [2]

(a) *Root bridge* zasílá pomocí Designated portu žádost – *Agreement*.

(b) *Alternate port* je ve stavu *Discarding*, je proto synchronizován.

(c) *Designated port* ve stavu *Forwarding* se zablokuje (přejde do stavu *Discarding*) a započne na něm jeho *proposal/agreement*. Avšak pro tento předchozí *proposal* je již synchronní.

(d) *Okrajový port* připojený jen ke stanici, je také synchronizován, není ho třeba ani blokovat.

(e) Všechny dostupné aktivní porty jsou synchronní a odesílá se potvrzení.

(f) Odesílá se žádost pro *Designated port*, stejně jako v předchozím případě.

(g) Odpověd.

Pokud po žádosti není obdrženo potvrzení, port přechází do stavu *Forwarding* obvyklým způsobem. To může nastat v případě, že jeho sousední přepínač nerozumí RST BPDU.

4.2.6 Kompatibilita

Jak bylo zmíněno v úvodu, RSTP je kompatibilní s jeho předchůdcem STP. Kompatibilita se děje na bázi portu (*per Port behavior*). Pokud některý port detekuje existenci STP uzlu, přepne své chování do kompatibilního módu a pracuje dle předchozí verze. Tím se však zbavuje veškerých výhod. Například: rychlejší přechody mezi stavy, *keep-alive*, *proposal/agreement* mechanismus. Musí dodržet správnou konverzi ceny linek (viz 4.2.1) a posílat jen STP *konfigurační BPDU* a TCN BPDU, aby komunikaci rozuměl i protější přepínač. Detekce probíhá tak, že STP přepínač novější verze BPDU zpráv zahazuje a míto nich posílá STP BPDU dle své normální funkce. K tomu slouží *Migration Timer*, kde změna verze protokolu na portu může proběhnout, až když vyprší časovač. Díky tomu je omezeno časté přepínání verzí kompatibility.

4.3 Spanning Tree a Virtual LAN

Se zavedením virtuálních lokálních sítí vznikly nové problémy a požadavky na aktivní topologie. Problém nastává, pokud síť používající např. STP, je rozdělena pomocí VLAN na segmenty, ale tyto nepokrývají celou síť (resp. všechny sdílené linky). To může být z důvodu bezpečnosti. Nemusí být vhodné šířit citlivá data přes zařízení, které je například ve správě jiného subjektu připojeného do sítě (zaměstnanecká a studentská část sítě). Tuto bezpečnost je vhodné zajistit za normálního provozu, ale i při výpadku určitých linek poté, co se rekonfiguruje aktivní topologie. Příkladem takové topologie, kde problém klasického STP nastává, je ilustrován obrázkem 4.10. Rozdělení VLAN také může být dáno lokalitou. Není vhodné šířit *broadcast* provoz přes celou síť, když v odlehlé části sítě do dané VLAN nikdo nepatří. Tímto by se zbytečně vytěžovaly linky.

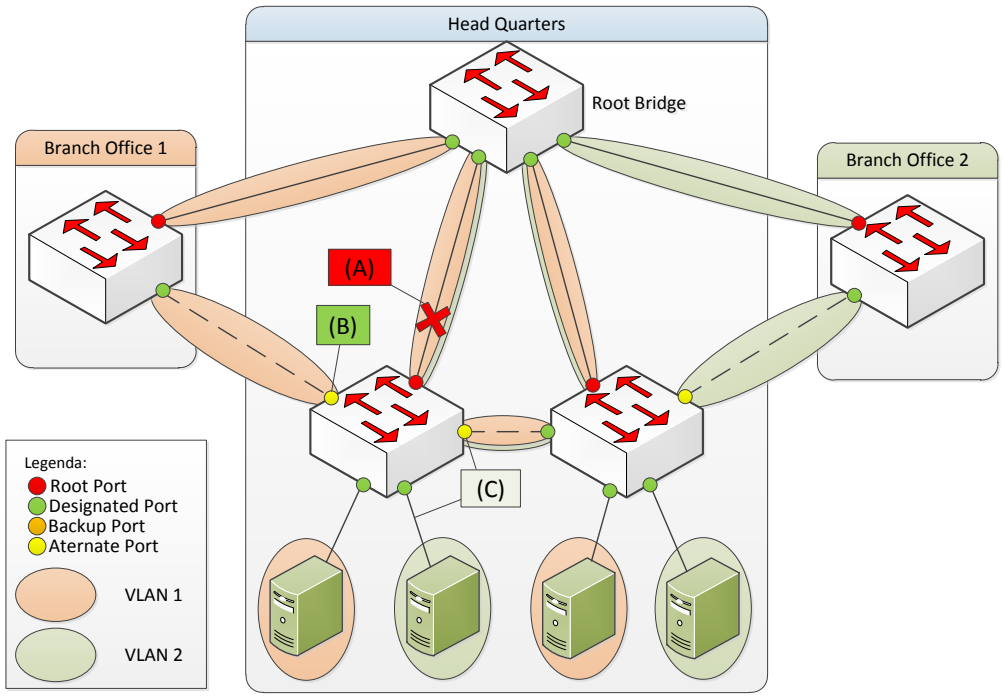
Další nevýhodou klasického STP je existence jedné aktivní topologie, vzhledem k *Virtuálním sítím*. U jedné přepínané sítě toto nelze nijak ovlivnit, ale v případě rozdělení dané sítě pomocí VLAN a použitím *Spanning Tree* s jednou aktivní topologií, budou existovat nevyužité linky. Proto mohu rozdělit redundantní linky, např. mezi budovami, do různých VLAN. Tímto se zajistí vyvažování zátěže, avšak na úkor redundance. Při pádu jedné z linek bude ztracena konektivita mezi segmenty patřící do těchto VLAN. Dalším problémem je funkce STP, jež jednu z těchto redundantních linek „odřízne“, protože nepočítá s existencí VLAN, natož s jejich oddělením.

Příklad na obrázku popisuje topologii a nastavení sítě, na kterém je ilustrován problém klasického STP na VLAN topologii. Předpokládejme konvergenci aktivní topologie, jak je naznačena pomocí rolí portů.

(a) Linka, která spojuje segment obsahující servery s kořenem aktivní topologie, se porouchá.

(b) Začne rekonfigurace. Na naznačeném portu se změní stav z *Alternate* na *Root*, čímž se ustálí aktivní topologie a z pohledu STP je vše v pořádku.

(c) Avšak z pohledu VLAN 2, je zde přerušena konektivita k jednomu ze serverů. Naznačený port, propojující centrální redundantní segment sítě se servery, je ve stavu



Obrázek 4.10: Ilustrace problému použití rozdělených VLAN s klasickým STP

Alternate, tedy nepřepíná provoz. Tímto nastává výše zmíněný problém, jelikož STP nebere virtuální sítě v potaz.

Na tomto ukázkovém příkladu je řešení zřejmé. Stačí zvýšit prioritu portům v centrálním redundantním segmentu. Ale v rozsáhlejších sítích toto zřejmé být nemusí. Proto je nutné použít některý z pokročilých *Spanning Tree* protokolů.

4.4 Cisco Spanning Tree protokoly

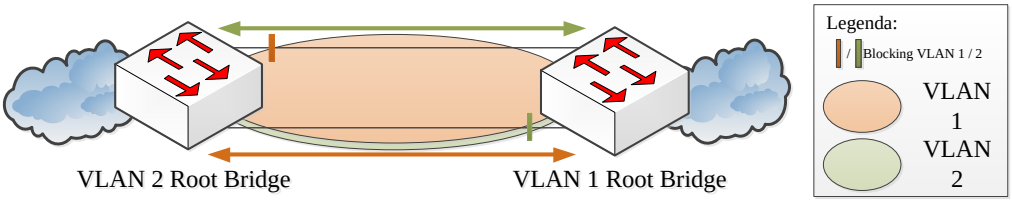
Firma Cisco vyvinula řadu proprietárních rozšíření, které se snaží vyřešit dané problémy a požadavky. Tato řešení tedy nejsou kompatibilní se standardními protokoly, a proto je nutné používat zařízení firmy Cisco.

4.4.1 Per-VLAN Spanning Tree (PVST) a jeho varianty

Informace čerpány z [7, 6, 3, 4, 1].

Přímočarým řešením výše popsaného problému je vytvářet oddělené aktivní topologie pro každou z VLAN. Proto je vytvořeno množství STP procesů a každý z nich operuje právě v jedné VLAN. Všechny procesy dle stejných pravidel jako klasické STP zvolí kořen a od něj pomocí šíření BPDU „roste“ *Spanning Tree* v rámci jednotlivých VLAN. Je tedy zajištěno, že jedna aktivní topologie nezablokuje linku, která je jediným spojením některé virtuální sítě. Přepínání tedy operuje se stavem portu dle kontextu VLAN, v jakém se daný rámec nachází.

Nastavení priorit uzlů a portů, nastavení časovačů a cen jednotlivých linek lze provádět pro každý STP proces zvlášť (*Per-VLAN*). Díky, tomu je řešitelná i druhá část problému. Při vhodném nastavení je možné vyvažovat provoz na síti pomocí rozdílných aktivních topologií. Situaci ilustruje obrázek 4.11.



Obrázek 4.11: Vyvažování zátěže pomocí Cisco PVST

Jednotlivé varianty *Per-VLAN Spanning Tree* protokolu firmy Cisco se liší verzí STP, která je použitá v *per-VLAN* procesech. V případě PVST je to verze STP dle standardu IEEE 802.1D-1998, kde Cisco přidalo své proprietární vylepšení *UplinkFast*, *BackboneFast*, *PortFast*, *BPDU Guard* a další (vizte [4]). Naproti tomu varianta *Rapid*-PVST používá *Rapid Spanning Tree Protocol*.

Další varianty jsou nazvány "plus", tedy PVST+ a *Rapid*-PVST+. Základní varianty Per-VLAN STP jsou uzpůsobeny pro komunikaci po linkách pomocí Cisco proprietárního mechanizmu pro přenos dat ve virtuálních sítích. Tento mechanizmus se nazývá *Inter Switch Link* (ISL), který používá odlišné zapouzdření než IEEE802.1Q a zcela mění tvar *Ethernet* rámce. Pro více informací o ISL mechanizmu vizte [1]. Varianta Per-VLAN protokolů je naopak uzpůsobena pro práci se standardizovanými formáty rámců. Porovnání přehledně vyobrazuje tabulka 4.8:

Varianta protokolu	Formát rámců	Verze Spanning Tree
PVST	ISL (Cisco)	STP
PVST+	IEEE 802.1Q	STP
Rapid-PVST	ISL (Cisco)	RSTP
Rapid-PVST+	IEEE 802.1Q	RSTP

Tabulka 4.8: Porovnonání variant Cisco PVST

4.4.2 Multiple Instance Spanning Tree Protocol (MISTP)

Předchozí varianty Spanning Tree Protokolu jednoduše řeší problém STP a oddělených VLAN. Jejich nevýhodou je to, že přidávají veliké nároky na výpočetní výkon procesoru na zařízení, jež by se mohl použít jinak. Dále zvyšují datové toky spotřebované vícenásobným rozesíláním BPDU pro každou VLAN zvlášť. Tento problém se firma Cisco snažila vyřešit *Instancemi STP* protokolu, které by oproti Per-VLAN obsluhovaly více VLAN najednou. Přičemž se velmi snižují nároky na výkon i na přenosové pásmo. Sdružené VLAN pod jednu instanci musí dodržet shodné rozložení po topologii, jinak by nastal problém STP na oddělených VLAN, kvůli němuž byly tyto varianty zavedeny.

Informace o pravém MISTP jsou velmi těžce dohledatelné. Kratičký popis této varianty se dá nalézt v [3]. Zde je dokonce popsána konfigurace pro přechod z PVST+ na MISTP pomocí MISTP-PVST+ módu, který zajišťuje bezešvý přechod při změnách konfigurace. Dokonce na určitých místech je varianta MISTP (Cisco) aktivně zaměňována za MSTP (IEEE 802.1Q), vizte [5].

4.5 Multiple Spanning Tree Protocol (MSTP)

Standardizovaným řešením aktivních topologií nad virtuálními lokálními sítěmi je MSTP, který specifikuje standard IEEE 802.1Q-1998 ([16], z něhož bylo převážně čerpáno pro tuto kapitolu. Další informace poskytly zdroje [3, 8].

4.5.1 MST Regiony

Základní idea je rozdělení jednotlivých uzlů do tzv. regionů (*MST Region*) spojených nadřazeným stromem *Common Spanning Tree* (CST). Takové regiony se potom chovají jako uzavřené[7] části sítě, které lze z pohledu modelu nahradit jedním uzlem. To je zajištěno *Internal Spanning Tree* (CST), který navazuje na CST a vytváří propojení všech uzlů v regionu s vnějškem. Uvnitř takového regionu jsou poté sestaveny aktivní topologie pro všechny virtuální sítě, jež se nalézají v daném regionu. Tyto vnitřní aktivní topologie jsou nazývány *Multiple Spanning Tree Instance* (MSTI). Jedna instance může operovat v rámci více VLAN, podobně jako CISCO MISTP.

Nad jednotlivými *MST Regiony* je poté sestaven CST, který je spojen s IST a tím spojuje všechny uzly v síti do *Common and Internal Spanning Tree* – CIST. Rozdělení topologie do regionů ilustrují následující obrázky: obrázek 4.12 – fyzická topologie, číslo udává *BridgeID* a případná informace oddělená čárkou je *identifikátorem nastavení MST* vysvětleného níže. Obrázek 4.13 znázorňuje rozdělení do regionu po výměně informací pomocí BPDU. CST na obrázku je topologie umístěna mimo vyznačené regiony.

4.5.2 MST Identifikátor konfigurace

Regiony se rozpoznávají podle identifikátoru nastavení daného uzlu, jež sestává z:

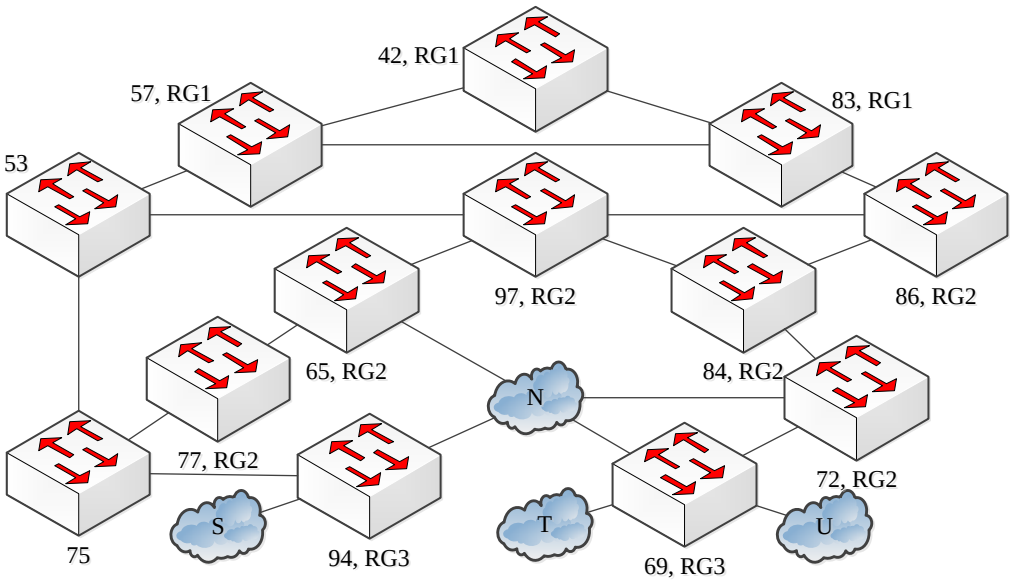
Configuration Identifier Format Selector - toto pole určuje formát identifikátoru a jeho hodnota je určena standardem na 0 a velikost pole je 1B.

Configuration Name - název dané konfigurace, kódovaný jako text proměnné délky, může dosahovat až 32B. Kratší název je ukončen znakem `NUL`. Více informací lze nalézt v [12, definice `SnmpAdminString`] [NON VIDI].

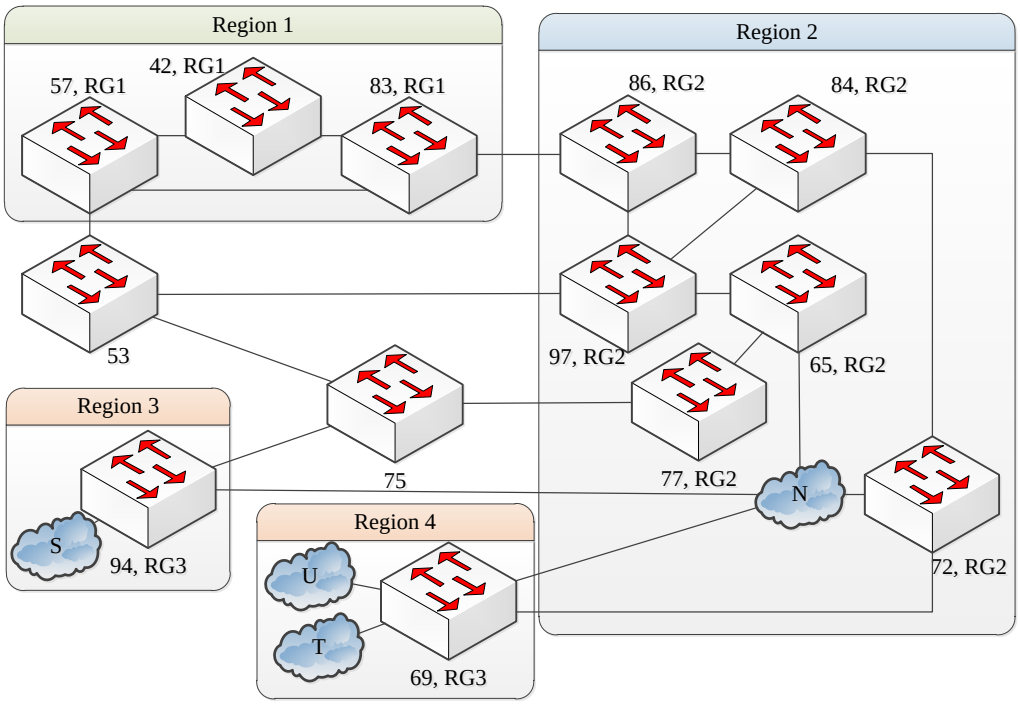
Revision Level - udává číslo revize dané konfigurace a kóduje se do pole velkého 2B.

Configuration Digest - je 16B podepsaný *hash* (klíč je uveden v tabulce 4.9) *mapování Instancí MST* na jednotlivé VLAN. *Hash* je proveden funkcí HMAC-MD5 (pro více informací vizte [11] [NON VIDI]) nad tabulkou MST konfigurace. Ta obsahuje 4096 polí velikosti 1B. První pole má fixní hodnotu 0. Každé další potom obsahuje jednu VLAN. Hodnota pole této VLAN je *instance MST*, která danou (respektive dané) VLAN obsluhuje. Poslední pole je také naplněno hodnotou 0. VLAN, na které není

[7] Uzavřené zde neznamená, že nedovolují komunikaci s okolím



Obrázek 4.12: Příklad fyzické topologie s nastaveným MSTP, přepracováno z [16]



Obrázek 4.13: Stejná topologie jako v obrázku 4.12 s ustavenými regiony, přepracováno z [16]

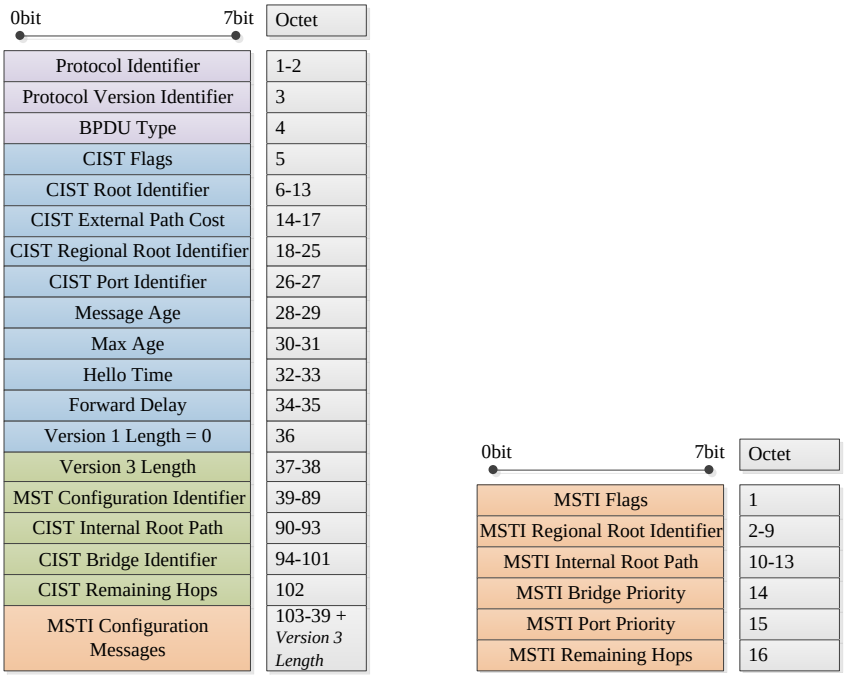
mapována žádná *MST instance*, obsahují také hodnotu 0. Speciální případ je tabulka obsahující jen nuly a ta říká, že všechny VLAN jsou přiřazeny ke stromu vyšší úrovně (CIST, viz dále). Tato konfigurace by měla být revize číslo nula. Uzly s touto speciální konfigurací nebudou patřit do stejného regionu.

Parametr	Hodnota
Configuration Digest Signature Key	`0x13AC06A62E47FD51F95D2BA243CD0346`

Tabulka 4.9: Definice klíče, jež se používá při výpočtu *Hash funkce* z *MST konfigurace*, zdroj [16]

4.5.3 MSTP BPDU

Vylepšeno bylo BPDU. Aby nezatěžovalo linku, proto je posíláno jen jedno MST BPDU pro všechny informace daného uzlu. Informace všech instancí jsou obsaženy v jednotlivých sekcích, kde si je daná instance převezme. Na původních pozicích RSTP BPDU jsou uloženy informace k externí části CIST (tedy CST) a *BPDU Type*, ten obsahuje hodnotu 2 (RSTP). Tímto jednoduchým krokem je zajištěna kompatibilita s předchozími verzemi, jež si nejsou vědomy existence vícenásobných instancí. Avšak jsou schopny pracovat s *Common Spanning Tree* a mohou se tedy zapojit do aktivní topologie bez jakýchkoliv problémů. Zbytek BPDU budou jednoduše ignorovat. Rozložení informací v BPDU ilustruje následující obrázek 4.14:



(a) MSTP BPDU (b) MST Configuration Messages

Obrázek 4.14: Struktura zpráv BPDU používaných v MSTP, přepracováno z [16]

Protocol Version Identifier pro MST BPDU je stanoven na hodnotu 3. Po části shodné s RSTP BPDU následuje:

Version 3 Length – udává počet následných instancí a jejich informací (*MSTI Configuration Messages*)

MST Configuration Identifier – *Identifikátor konfigurace MST* pro detekci regionů, vizte sekci 4.5.2.

CIST Internal Root Path – cena cesty ke kořenovému uzlu v IST.

CIST Bridge Identifier – *Bridge Identifier* uzlu v IST, který BPDU odeslal.

CIST Remaining Hops – určuje maximální velikost topologie v CIST.

MSTI Configuration Messages

MSTI Flags – parametry shodné jako v RSTP (sekce 4.2.4), určené pro danou *instanci MST*.

MSTI Regional Root Identifier – *Bridge Identifier* kořene pro danou instanci.

MSTI Internal Root Path – cena cesty ke kořeni dané instance.

MSTI Bridge Priority – priorita uzlu v dané MSTI, jež odeslal BPDU.

MSTI Port Priority – priorita portu v dané MSTI, přes který bylo BPDU odesláno.

MSTI Remaining Hops – určuje maximální velikost topologie určené danou instancí v rámci regionu.

MSTP samozřejmě podporuje i předchozí verze kódování BPDU. I v případě přijetí BPDU s hodnotou *Protocol Version Identifier* = 3, která má délku menší než 103 oktetů, zachází jako s RSTP BPDU.

4.5.4 Kompatibilita

Spanning Tree Protocol řeší problém STP a rozdělených VLAN. Dále poskytuje nastavení pro vyvažování zátěže. To umožňuje díky instancím, na které lze mapovat různé počty VLAN. Je tedy možné „dopilovat“ rozložení VLAN mezi instance pomocí priorit uzlů a linek, pro nejlepší aktivní topologii v rámci všech regionů. Tím je zpráva STP na rozsáhlé síti velmi přehledně segmentována do spravovatelných celků.

Navíc ke všem výhodám oproti Cisco PVST/MISTP, poskytuje MSTP zpětnou kompatibilitu. Ta je transparentní pro „starší“ zařízení. Umožňuje to díky sdílenému stromu CIST, který se sestavuje nad jednotlivými regiony. Ty z pohledu CST lze funkčně nahradit za jedno zařízení.

5 Návrh modelu

Popis modelu sleduje linii popisu ze standardů IEEE 802.1D [14, 15] a IEEE 802.1Q [16], ze kterých je taky čerpána převážná část informací.

Model obsahuje čtyři základní části, které jsou nutné pro funkci zařízení na síti druhé vrstvy s podporou VLAN a redundantních linek. Jsou to *Forwarding process*, který popisuje způsob, jakým se přepínají rámce, *Learning Process* popisující způsob učení rozložení stanic na síti a její topologii, *Filtering Base*, což je datová struktura obsahující informace z učícího procesu a informace načtené z konfigurace. A nakonec *Active Topology Enforcement*, jež zajišťuje správnou funkci zařízení a tím i sítě v prostředí obsahující smyčky.

5.1 Forwarding process

Tento proces řídí rozhodování v přepínání rámců založené na pozicích cílových stanic. Přepíná v oddělených virtuálních sítích, které mají vyhrazené linky nebo je mohou sdílet. Zajišťuje rozdělení *broadcast domén*[1] vzhledem k nastavení VLAN.

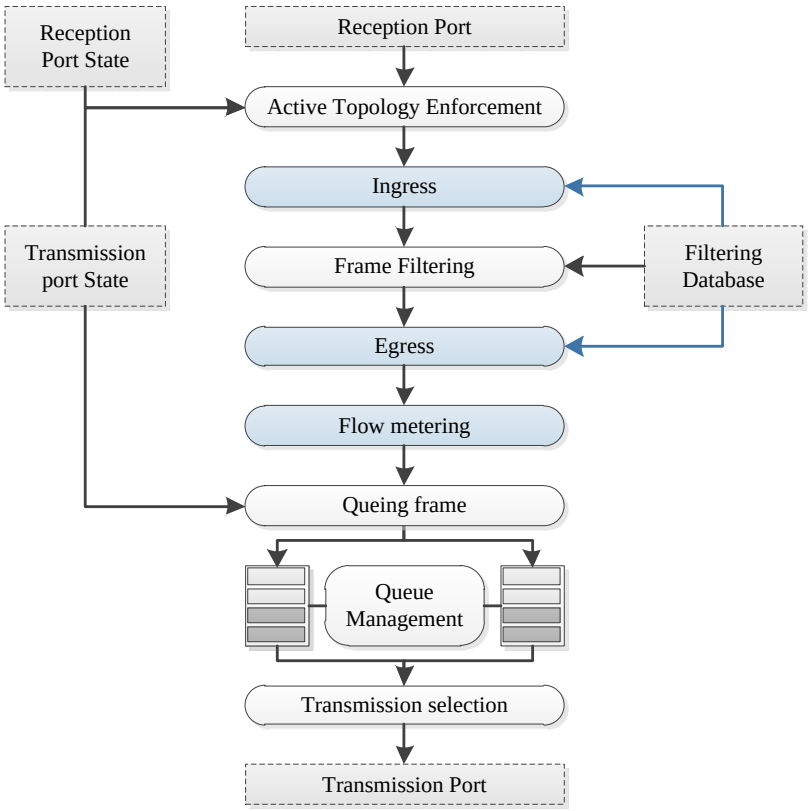
Obrázek 5.1 přehledně zobrazuje propojení jednotlivých částí procesu *Forwarding process*. Dále pak je na něm vyznačena spolupráce s ostatními moduly a barevně jsou vyznačeny části, které jsou navíc kvůli potřebě VLAN, oproti původní verzi ze standardu IEEE 802.1D.

Proces pracuje následovně:

- přijme rámec na přijímacím portu port_s
- *Active topology enforcement* přijme nebo zamítne rámec vzhledem k danému stavu portu (vizte 5.4)
- aplikuje se *Ingress* pravidlo a je přijat nebo zamítnut v případě, že nepatří do dané VLAN vzhledem k port_s
- vytvoří se množina cílových portů $\mathbf{Port}_d$, která se pomocí *Frame filtering* funkce redukuje na základě známých cílů
- poté jsou aplikována pravidla *Egress*, které redukují množinu $\mathbf{Port}_d$ na základě příslušnosti k cílové VLAN
- nakonec je rámec duplikován na všechna aktivní rozhraní $p \in \mathbf{Port}_d$ podle stavu a jsou zařazeny do jednotlivých front k odeslání

Pro další výklad použiji zápis zprávy $m = (s, d, v, m^+)$, kde s je adresa zdroje, d je adresa cíle nebo adresa broadcast $d = \mathbf{b}$, v je označení náležitosti k dané virtuální síti

[1] Množina uzlů sítě $\mathbf{BD} = \{u \mid u \in V \wedge deliver(u, m_b)\}$, kde m_b je zpráva s adresou broadcast a funkce $deliver(um, m_b)$ říká, že uzlu u bude zpráva m_b doručena.



Obrázek 5.1: Schéma ze standardu IEEE 802.1Q [16] s vyznačenými rozdíly oproti IEEE 802.1D [15].

– VLAN nebo ϵ pokud označení nemá a m^+ je zpráva vyšší vrstvy, tedy užitečná data vzhledem k tomu, že se zajímáme o vrstvu druhou.

Dále pak předpokládejme existenci dvojic $(a, p) \in \mathbf{MT}$, kde a je existující adresa v síti a p je port, na kterém je tato adresa dosažitelná a $\mathbf{MT}$ představuje množinu těchto asociací. Jedná se o MAC Tabulkou.

Předpokládejme existenci dvojice $(v, \mathbf{P}) \in \mathbf{VT}_v$, kde v představuje označení VLAN (neseno ve zprávě m), $\mathbf{P}$ je množina portů, které náleží do této VLAN a $\mathbf{VT}_v$ je množina těchto asociací VLAN-porty.

A nakonec definuji dvojici $(p, v) \in \mathbf{VT}_p$, kde p je port a v je jeho asociace k VLAN, $\mathbf{VT_p}$ je množina těchto asociací. Množiny $\mathbf{VT}_v$ a $\mathbf{VT}_p$ jsou VLAN tabulkou.

Množiny $\mathbf{MT}, \mathbf{VT}_v, \mathbf{VT}_p$ jsou součástí modulu *Filtering database* a budou detailněji vysvětleny v sekci 5.3

Ingress

V případě zprávy, která je označena hodnotou v, pravidlo *Ingress* povoluje zprávy:

$$m_0 \ : \ v_0 \neq \epsilon \wedge \exists (v, \mathbf{P}) \in \mathbf{VT}_v \ . \ (v \ = v_0, \ \text{port}_s \in \mathbf{P}) \tag{5.1}$$

Naopak, pokud zpráva m_0 není označena $v_0 = \epsilon$, pak $\exists(p, v) \in \mathbf{VT}_p \,.\, p = \text{port}_s$ a zpráva je považována, že patří do VLAN v.

Filtering Process

Na počátku se vytvoří množina cílových portů:

$$\mathbf{Port}_d = \{p_i \mid 0 \leq i \ < n \wedge p_i \neq \text{port}_s\}, \tag{5.2}$$

kde n je počet portů zařízení.

Tato množina obsahuje všechny výstupní porty kromě portu, na kterém byla zpráva přijata.

Pokud je zpráva m_0 adresována jako *broadcast*, tedy $d = \mathbf{b}$, je množina vytvořená tímto krokem korektní. V případě, že $d \neq \mathbf{b}$ je nutné zprávu odeslat jen portem, kde se nalézá cíl. Pokud známe výstupní port cíle:

$$\exists(a, p_u) \in \mathbf{MT} \,.\, a \ = d \wedge a \ \neq \mathbf{b}, \tag{5.3}$$

pak redukujeme množinu $\mathbf{Port}_d = p_u$ a zpráva m_0 bude odesílána jen portem p_u.

Egress

Pravidla výstupního filtrování *Egress* pokračují v redukci množiny $\mathbf{Port}_d$ nehledě na to, jestli je v množině jeden port, či portů více. Redukce se provádí na základě asociace VLAN a portů. Mějme tedy přijatou zprávu m_0 a vygenerovanou množinu $\mathbf{Port}_d$. Provedeme redukci podle znalosti v_0 dané zprávy nebo dle zjištěného v pokud $v_0 = \epsilon$, viz. 5.1, položme tedy `iff` $v_0 \neq \epsilon$ `then` $v \ \leftarrow v_0$.

$$\mathbf{Port}_d \leftarrow \mathbf{Port}_d \setminus \{p \mid p \in \mathbf{Port}_d \wedge \nexists(v, \mathbf{P}) \,.\, p \in \mathbf{P}\} \tag{5.4}$$

Queuing Frame

Nakonec *Forwarding process* je třeba ještě doplnit drobnost. Před duplikací rámce do výstupních front jednotlivých portů se kontroluje, zda-li je port aktivní, více v 5.4.

5.2 Learning Process

Zajišťuje funkci učení informací o síti, podle kterých se zpřesňuje proces přepínání rámců. Proces učení se využívá v případě:

- přijatá zpráva m_0 je korektně přijata – není vyloučena pravidlem *Ingress*,
- port, na kterém byla zpráva přijata, má povoleno učení procesem *Active Topology Enforcement*,
- zdrojová adresa je specifickou adresou stanice (tj. adresa není skupinová nebo broadcast),
- adresa není ve *Filtering Database* vedena jako statická položka (např. nastavená administrátorem)
- a přidáním položky se nesmí překročit kapacita databáze.

Funkce *filtering procesu* je zakládat a aktualizovat znalosti o umístění cílů v síti vzhledem k VLAN. Znalost je pouze lokální, tedy známe informaci o tom, kterým lokálním portem existuje cesta k cíli. Pomocí sloučení znalostí všech zařízení v síti, je možno sestavit přehled o umístění cílů, což ale zařízení nepotřebuje a nedělá. Učení znalostí jen vzhledem k VLAN je zajištěno díky aktivaci učení až po průchodu vstupními pravidly. Pokud by informace o zprávě nebyly v daném kontextu relevantní, tj. například nepatří do dané VLAN na portu, nebyla by zpráva přijata a proces učení aktivován.

Pokud by mělo vkládání překročit kapacitu databáze, je možno odstranit dynamické položky, aby se uvolnilo místo pro nové. Je vhodné odstraňovat nejstarší záznamy, pokud je bude v budoucnu potřeba, vloží se jako položky nové.

Nyní je třeba zmínit, že v množině **MT** jsou uloženy trojice (a, p, t_i), kde t_i je čas vytvoření nebo aktualizace daného záznamu. Hodnota t_i je pro *Forwarding process* nezajímavá, proto jsou ve výkladu použity dvojce (a, p). Dále v 5.3 uvidíme jak se s hodnotou t_i pracuje.

Proces tedy zakládá informace a aktualizuje existující. Pro existující je to komplikovanější, musí je nejprve nalézt a poté změnit. Nechť je zpráva $m_0 = (s_0, d_0, v_0, m^+) \wedge d_0 \neq \mathbf{b}$, a platí že $\exists (a, p, t_i) \in \mathbf{MT}.a \ = d_0$, pak aktualizujeme existující položku takto:

$$\mathbf{MT} \leftarrow \mathbf{MT} \setminus (a, p, t_i) \tag{5.5a}$$

$$\mathbf{MT} \leftarrow \mathbf{MT} \cup (a, p', t_i') \ . \ a \ = d_0 \wedge p' = \mathrm{port}_s \wedge t_0' = t_{\mathrm{now}} \tag{5.5b}$$

Pokud položka existuje, vytvoří se jako v případě vložení nového při aktualizaci.

5.3 Filtering Database

Je modulem, který poskytuje znalostní bázi pro správné rozhodování ostatních procesů. Znalosti jsou vkládány a aktualizovány výše popsaným *Learning process* nebo jsou předdefinovány konfigurací pro danou topologii.

Filtering Database v podstatě obsahuje množiny **MT**,$\mathbf{VT_v}$ a $\mathbf{VT_p}$, se kterými pracuje. Vyhledává v nich informace potřebné pro *Forwarding process* a předává je ve formě potřebné pro tento proces. Není třeba nutné předávat výše dodatečně zmíněný čas t_i.

5.3.1 MAC table

Tabulka asociací MAC[2] adres na porty, je dynamickou tabulkou, kterou v teoretickém výkladu reprezentuji množinou **MT**.

Tabulka je částečně dynamická. Dynamickou část tabulky obhospodařuje *Learning process*, který ji vytváří a aktualizuje. V dynamické části dále *MAC tabulka* sama kontroluje položky s hodnotou $t_i + t_{\mathrm{now}} \geq t_{\mathrm{aging}}$, kde t_{aging} je hodnota stárnutí položek. Pokud tuto dobu překročí, jsou považovány za zastaralé a jsou smazány.

Dále pak existují statické položky předdefinované standardem. Je to například skupinová adresa všech přepínačů na daném segmentu. Na tyto se nevztahuje kontrola t_i.

Hlavní funkcí MAC tabulky je předávání hodnot pro *Forwarding Process*. Předává ale jen relevantní položky, takže musí existovat a nesmí přesáhnout stáří. Pokud přesáhnou stáří v době dotazu, MAC tabulka tyto položky eliminuje. Tabulka předává hodnoty (a, p) takto:

[2] MAC – Media Access Control

$$p = MT(a) = \begin{cases} p & \text{pokud } \exists(a, p, t_i) \in \mathbf{MT} \wedge t_i + t_{\text{now}} < t_{\text{aging}} \\ \epsilon & \text{pokud } t_i + t_{\text{now}} \geq t_{\text{aging}} \Rightarrow \mathbf{MT} \leftarrow \mathbf{MT} \setminus (a, p, t_i) \\ \epsilon & \text{jinak} \end{cases} \tag{5.6}$$

Forwarding process používá k rozhodování hodnotu p, proto není potřeba předávat veškeré informace.

5.3.2 VLAN table

Tabulka VLAN je složená ze dvou částí. Část, kde se asociují množiny portů na jednotlivé VLAN, a část, kde je každému portu přiřazena právě jedna VLAN. Jsou to právě množiny $\mathbf{VT}_v$ a $\mathbf{VT}_p$.

VLAN tabulka poskytuje informace pro pravidla *Ingress* a *Egress*, na základě známého Port_s a známého nebo zjištěného v.

První důležitá funkce je právě zjišťování příslušnosti k VLAN neoznačené zprávy, tj. $m_0 \ . \ v_0 = \epsilon$. Tato informace se zjišťuje takto:

$$VT_p(\text{Port}_s) = v \ . \ (p, v) \in \mathbf{VT}_p \tag{5.7}$$

Tímto získáme zařazení zprávy do VLAN, která nebyla označená. Nadále lze se zprávou m pracovat tak, že náleží do v.

Další důležitou funkcí je zjišťování existence mapování konkrétního v na konkrétní Port_s, což se využívá v *ingress* při filtrování zpráv nepatřící do dané VLAN.

$$\texttt{Allow}(v', \text{Port}_s) = \begin{cases} \text{povol} & \text{pokud } \exists(v, \mathbf{P}) \in \mathbf{VT}_v \ . \ v \ = v' \wedge \text{Port}_s \in \mathbf{P} \\ \text{zamítni} & \text{jinak} \end{cases} \tag{5.8}$$

A poslední funkce je předání množiny cílových výstupních portů pro danou VLAN v případě, že zpráva m je adresována všesměrově, tj. $d = \mathbf{b}$. Vyhledá se tedy celá množina $\mathbf{P}$ na základě v.

$$VT_v(v') = \mathbf{P} \ : \ \exists(v, \mathbf{P}) \ . \ v \ = v' \tag{5.9}$$

5.4 Active topology enforcement

Tento proces zajišťuje správnou funkci sítě i v případě, že obsahuje smyčky. Toho je dosaženo pomocí STP – *Spanning Tree Protocol*. STP nalezne v síti podle parametrů jednotlivých vrcholů kořen a podle parametrů linek si každý prvek vytvoří minimální kostru grafu. Podle kostry pak přiřadí portům role a dle nich platné stavy, aby mohly přijímat/zamítat zprávy.

Definuji port jako trojici (p, s, r), kde p je port definovaný výše, s je stav daného portu, ve kterém se nachází $s \in \{\text{Disabled}, \text{Discarding}, \text{Learning}, \text{Forwarding}\}$ a r je role portu v procesu *Active Topology Enforcement* $r \in \{\text{Disabled}, \text{Alternate}, \text{Root Port}, \text{Designated}\}$, tedy v STP. Kořenový přepínač – *Root Bridge* má všechny porty role *Designated*.

Popíši nyní role portů podle parametrů, jak se v STP přiřazují:

Disabled – port není fyzicky aktivní, STP s ním nepracuje.

Backup – je záložním portem, který je připojen do stejného segmentu jako jiný aktivní port na stejném uzlu. Poskytuje záložní přístup do segmentu ze stejného uzlu.

Alternate – je alternativní port, který udržuje informace o stejném kořeni jako jiný aktivní port, ale je oddělen jiným přepínačem (tedy informace, jež udržuje, jsou z jiného přepínače). Poskytuje alternativní cestu k danému segmentu přes jiný uzel. Přímo v protokolu se nerozlišuje mezi *Backup* a *Alternate*. Z pohledu funkce jsou shodné.

Root Port – port směřující po minimální kostře ke kořeni.

Designated – port směřující po minimální kostře ke koncovým uzlům – stanicím.

Příklad přiřazení rolí je na obrázku 4.1 v sekci 4.1.2.

Dle rolí se následně přiřazují stavy dle stavového automatu (vizte 4.2). Ten prochází stavy s = Discarding, kde se učí parametry pro tvorbu kostry. Poté s = Learning, kde si zaznamenává pomocí *Learning Process* asociace cílových adres a portů. Nakonec stav s = Forwarding, díky tomu je zajištěn provoz na síti s redundantní topologií a následně i automatická rekonfigurace při pádu linek, pokud existují záložní (resp. alternativní).

Přechody ze stavu *Discarding* jsou dovoleny jen portům s rolí *Root* nebo *Designated*. Ty jsou totiž součástí aktivní topologie a budou poskytovat přepínání rámců. Role *Disabled* a *Alternate* zůstávají ve stavu *Discarding*, aby nevznikaly kružnice v topologii. Pro roli *Alternate* není dovoleno ani učení adres, protože by docházelo k přepisování položek ve *Filtering Database* špatnými daty a přepínání by nefungovalo správně.

Na portu je nutné uchovávat informace, jež daný port obdržel ze sítě pomocí *Configuration BPDU*. Je nutné z přijatých informací vybrat nejlepší, tedy všechny horší BPDU se zahazují. Poté je dle informací na všech aktivních portech ověřena existence kořene. Pokud je lokální informace lepší než všechny přijaté ze sítě, je daný uzel zvolen za kořenový a od něj se rozpíná aktivní topologie. Pokud na některém z portu existuje výhodnější informace, je tento (nejlepší z kandidátů) prohlášen za *Root Port*, jelikož tím směrem se nachází kořen.

Redundantní porty patřící do stejných segmentů (resp. poskytují alternativní cestu ke kořeni) jsou zvoleny jako *Backup* (resp. *Alternate*). To lze detekovat pomocí shodných identifikátorů *RootID* a *BridgeID*. Rozhodujícím parametrem je cena cesty ke kořenovému uzlu – *Root Path Cost*. Pokud se i cena shoduje, je dále rozhodnuto dle *BridgeID*, *PortID* následníka a případně dle *PortID* lokálního uzlu (to je v rámci místního rozhodnutí unikátní, protože se jedná i o číslo portu na zařízení).

Po zvolení rolí jednotlivých portů je spuštěn časovač *Forwarding Delay*, který udává přechody stavů automatu (viz obrázek 4.2).

Při přijetí informací je také spuštěn časovač *Message Age*, který zajišťuje vypršení platnosti naučených informací o topologii a tím i detekci výpadků. Při pravidelném rozesílání BPDU kořenovým uzlem a jejich přeposíláním je zajištěno obnovování těchto informací. Při detekci výpadku je nutné dle zbývajících informací přepočítat topologii (tj. ověřit platnost kořene, zvolit role ...). Následně je pak vysláno upozornění o změně topologie buďto ve formě *Topology Change Notification* nebo nastavením parametru *Topology Change* ve zprávě BPDU, pokud došlo ke změně kořene, případně pokud výpadek detekoval právě kořen.

Při přijetí tohoto oznámení je pak nastaven čas stárnutí ve *Filtering Database* na krátkou dobu. Tím budou špatné položky, které změnily přepočítáním topologie svou polohu vzhledem k daným uzlům, odstraněny. Proces přepínání nebude šířit rámce špatným směrem nebo rovnou zahazovat na základě špatných informací. Poté je obnoven předchozí čas stárnutí a funkce *Filtering Database* je v původním stavu.

Pro správnou funkci v prostředí VLAN je nezbytné k topologii přistupovat v rámci jednotlivých VLAN. Je tedy nutné veškerou funkci provádět pro každou VLAN zvlášť. Implementace standardizovaného MSTP je velmi rozsáhlá a komplikovaná, proto bylo navrženo řešení funkčně podobné *Per-VLAN Spanning Tree*. Pro každou VLAN budou tedy STP informace (identifikátory, role, stavy apod.) uchovávány a zpracovávány odděleně. BPDU zprávy budou obsahovat informaci o „instanci“ *Spanning Tree*, ve které se nacházejí a chování tedy bude řešeno odděleně. Tím se dosáhne správné aktivní topologie pro každou VLAN.

Porty koncových stanic nejsou změnami portů kostry zasaženy, jelikož informace na nich se nemění. Jejich funkce je sice zpožděna počáteční konvergencí (přechodem mezi stavy), což ale nevadí, protože v té době není kostra ještě ustavena a přijaté informace by byly samozřejmě zahozeny.

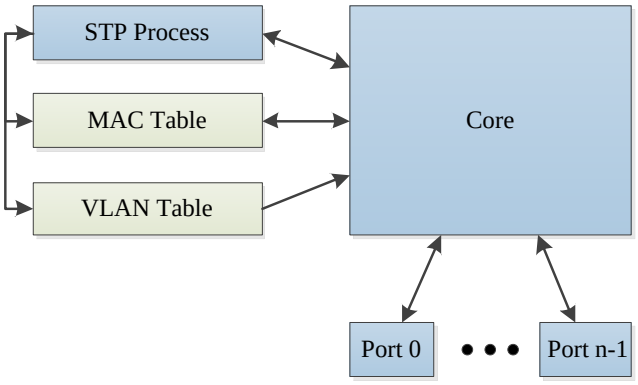
6 Implementace modelu

6.1 Architektura

Architektura zařízení se odvíjí od zařízení vyvinutého v rámci projektu *ANSA – AnsaRouter*. Předpokládejme, že upravím architekturu *AnsaRouter* vložením jádra druhé vrstvy mezi existující funkční moduly a výstupní rozhraní. Dále pak přidám moduly *Filtering Base – MAC table*, *VLAN table* a moduly pro *Active Topology Enforcement – STP process* a *Port Table*.

Jádro, které řídí funkci druhé vrstvy s pomocí *STP*, bude napojeno na vyšší vrstvu. Té bude předávat zprávy (resp. pakety) m^+, jež jsou určeny právě tomuto zařízení, nebo které musí, vzhledem k topologii a nastavení sítě, zpracovat vyšší vrstva.

Návrh architektury zobrazuje obrázek 6.1.



Obrázek 6.1: Architektura modelu zařízení

6.2 OMNeT++

Model zařízení bude implementován v jazyce C/C++ pro simulátor OMNeT++ [24].

Autorem je András Varga. OMNeT++ je simulační prostředí s kalendářem diskrétních událostí. Jeho primární oblastí působnosti je simulace síťové komunikace. Ale díky velmi flexibilní architektuře je používán i v jiných odvětvích jako simulace front, hardwarové architektury a také agentní systémy.

OMNeT++ poskytuje architekturu pro modely, jejichž jednotlivé komponenty jsou vytvářeny v jazyce C++. Poté jsou spojeny do větších celků a modelů. Ty používají vyšší

jazyk NED[1], díky němuž jsou modely znovupoužitelné i bez drastických úprav.

OMNeT++ má rozsáhlou podporu grafických rozhraní a díky modulární architektuře lze simulační jádro jednoduše vložit do vlastní aplikace.

Samotný OMNeT++ není síťovým simulátorem, je spíše platformou pro síťové simulace a díky silné komunitě existují doplňující balíky, které poskytují základní moduly pro simulaci sítí.

Dokumentace je velmi příjemně zpracována, jsou v ní veškeré základní informace pro vytváření simulací. Bohužel, pokud je potřeba vytvářet speciality jako velmi pěkný sekvenční diagram s popisem stavů modulů, je velmi těžké nalézt nějaké související informace, o návodu nemluvě.

6.3 Filtering Database

6.3.1 MAC Table

Je implementována jako mapa s vyhledávacím klíčem `MACAddress` (a). Do mapy jsou vkládány struktury, které obsahují výstupní port (p), čas vložení (t_i) a druh záznamu (statický, dynamický, skupinový[2]).

Nad mapou jsou implementovány funkce vkládání, aktualizace, odstranění zastaralých záznamů, smazání dynamických záznamů a smazání celé mapy. Díky použití mapy je časová složitost vyhledání ve třídě $O(\log n)$. Odstranění zastaralých záznamů je vzhledem k nutnosti projít celou mapu ve třídě $O(n)$, což se provádí jen na vyžádání, například při nedostatku volných položek. Odstraňování zastaralých položek se provádí také při vyhledávání tak, jak bylo popsáno v 5.3.1.

6.3.2 VLAN Table

Obsahuje 2 části, obě jsou implementovány jako vektor s pevnou délkou[3], do kterého se vkládají struktury představující jednotlivé záznamy.

Pro mapování VLAN-Porty struktura obsahuje seznam portů náležící k dané VLAN ($(p, v) \in \mathbf{VT_P}$), a navíc údaj o tom zda daným portem má zpráva být značena VID[4], tzv. *tagged* nebo neoznačena *untagged* ($p \in \mathbf{P}$ je v podstatě struktura obsahující port a příznak, zda se má označit).

Naproti tomu asociace portu k dané VLAN je vytvořeno jako struktura obsahující port a danou VLAN.

Nad vektory jsou implementovány funkce vyhledání, přidání, odstranění a modifikace záznamů. Pro přidávání portů k jednotlivým VLAN je implementována funkce s poměrně vyšší časovou složitostí. Vzhledem k tomu, že se tato funkce volá jen při prvotní konfiguraci a zároveň jen pro velmi malý počet prvků, je vyšší třída složitosti zanedbatelná. Díky tomu je přehlednější kód, jež obstarává konfiguraci databáze.

Operace vyhledání (potažmo i ostatní funkce) má časovou složitost $O(1)$, jelikož jako klíč při hledání je buďto číslo VLAN nebo číslo portu.

[1] NEtwork Descriptor

[2] tato oblast není momentálně v zájmu řešení

[3] pro VLAN-Porty je to 4096, což je rozsah číslování VLAN, pro Port-VLAN je to počet portů, údaj se čte z konfigurace

[4] VID – VLAN IDentifier

6.4 Forwarding Process

Funkcionalita *Forwarding Process* je implementována v jádře zařízení a je rozdělena podobně jako je tomu v teoretickém úvodu.

Jakmile je rámec přijat pravidlem *ingress*, které je zahrnuto v implementační části *reception* (zobrazuje obrázek 6.3), je vytvořena interní reprezentace zprávy, která ji provází celým procesem přepínání. V této reprezentaci jsou uloženy informace zjištěné přímo ze zprávy nebo na základě přijímacího portu ze znalostní databáze (*Filtering Database*).

Tyto údaje jsou:

source – zdrojová MAC adresa

destination – cílová MAC adresa

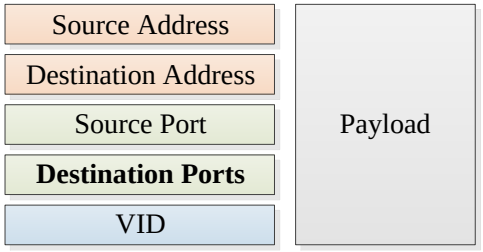
Port$_s$ – port, na kterém byl rámec přijat

Port$_d$ – seznam potenciálních výstupních portů

VID – označení VLAN, v jejímž kontextu je zpráva zpracovávána

Payload – užitečná data vyšší vrstvy

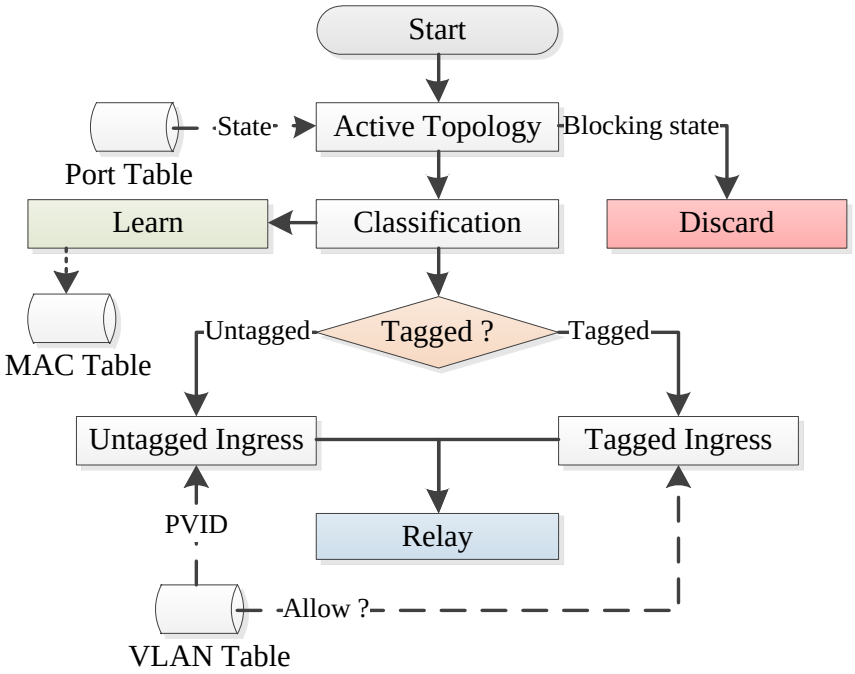
Názorně zobrazuje obrázek 6.2.



Obrázek 6.2: Zobrazení interní reprezentace rámce a informací, které jsou potřeba k plnému vyhodnocení.

Funkce přijímání rámce, zobrazená na obrázku 6.3, je implementována dle diagramu.

- funkce *discard* zajišťuje filtraci rámců na základě stavu portů.
- *klasifikace* spočívá v určení, zda-li je rámec označen pomocí VID (jedná se o jiný datový typ).
- *ingress* filtrace funguje dle teoretického popisu, ale je implementována zvlášť pro zprávy s VID a bez.
- *relay* je komplexní funkce vlastního přepínání rámců (vizte dále).



Obrázek 6.3: Zjednodušené schéma procesu příjmu rámců.

Funkci *Relay* vyobrazuje diagram 6.4 a je implementovaná dle něj. Nejprve je zjištěno, zda-li je zpráva adresována jako *broadcast* $d = \mathbf{b}$ a podle toho se se zprávou manipuluje.

Pokud je adresována jako *broadcast*, *Relay* zjistí cílové porty v dané VLAN a poznačí je do interní reprezentace zprávy. Není-li adresována jako *broadcast*, pokusí se zjistit z *MAC tabulky* známý výstupní port cíle. Pokud tento není znám, je zpráva zpracována jako *broadcast*.

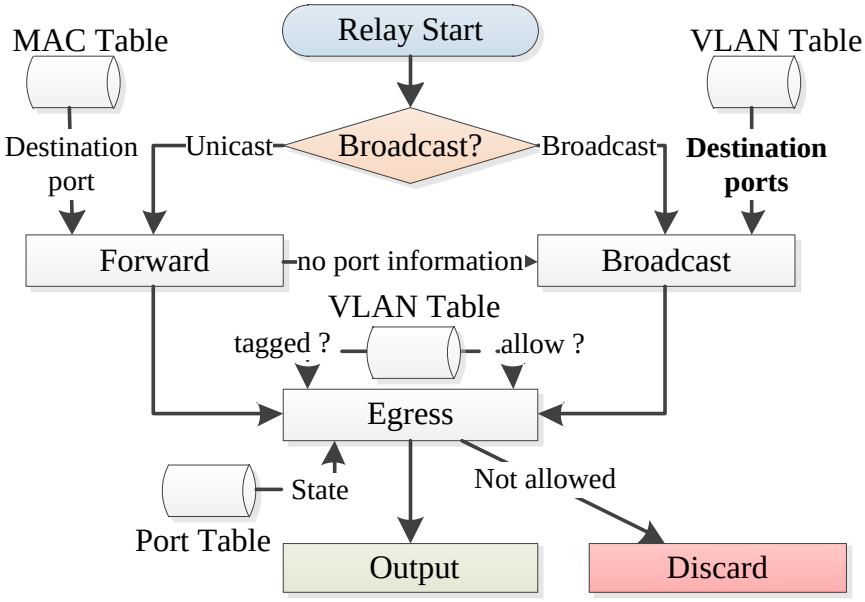
Dále je postoupena výstupnímu pravidlu *egress*, které omezí seznam výstupních portů dle informací z báze znalostí. Pokud existuje výstupní port, na který bude zpráva odeslána, jsou informace postoupeny *learning* funkci. Ta je implementována, jak je naznačeno v teoretickém popisu.

Poté je zpráva duplikována na všechny výstupní porty uvedené v seznamu Port_d.

6.5 Active Topology Enforcement

Pro funkci *Spanning Tree* protokolu byly vytvořeny zprávy `STPBPDU` a `STPTCN`, které obsahují potřebné informace dle standardizovaného STP, vizte obrázky 4.3 a 4.5. Dále pro funkci nad oddělenými VLAN byly opatřeny parametrem pro identifikaci VLAN, ve které se nacházejí. Tím je zajištěno oddělení instancí STP.

Identifikace portu, na němž byla BPDU zpráva přijata, je zajištěna vícenásobnými spoji



Obrázek 6.4: Detail přepínání rámců na výstupní porty.

STP modulu s modulem jádra. Počet těchto spojení je shodný s počtem portů, a tedy jednotlivé spoje odpovídají portům celého zařízení.

Byly navrženy struktury pro uchovávání informací na portech, pomocí nichž se následně vytváří topologie. K nim byly definovány metody vykonávající porovnání veškerých identifikátorů, které případně aktualizují informace na portech. Po přijetí informací je ověřeno, zda-li lokální uzel nesplňuje podmínky kořenového uzlu (*Root Eligibility*). Takový uzel si automaticky nastaví všechny aktivní porty do role *Designated* a snaží se tuto informaci šířit do okolí. Je-li v síti lepší uzel, po přijetí lepší zprávy lokální uzel rezignuje na pozici kořene a volí *Root Port*. Z ostatních jsou pak zvoleny další role.

Časovače jsou řešeny pomocí čítačů na portech a jsou aktualizovány a kontrolovány pomocí lokální zprávy `TICK`. Ta je plánována každou vteřinu, a tedy poskytuje mechanismus efektivní na prostředky simulace. Při vypršení některého časovače (resp. přetečení čítače) je aktivována metoda na řešení situace (přechod portu do následujícího stavu nebo detekce výpadku).

Metody porovnávání parametrů jednotlivých uzlů jsou shodné s porovnáváním parametrů lokálního uzlu. Struktura portu však obsahuje pro samotný uzel nepotřebné položky, jako roli a stav. Průběh porovnávání by tedy musel mít více specializovaných metod se shodnou funkcí. Je proto optimalizováno porovnání pro lokální uzel tak, že jeho informace se dočasně převedou do struktury portu. Potom jsou porovnány a podle výsledků je provedeno rozhodnutí.

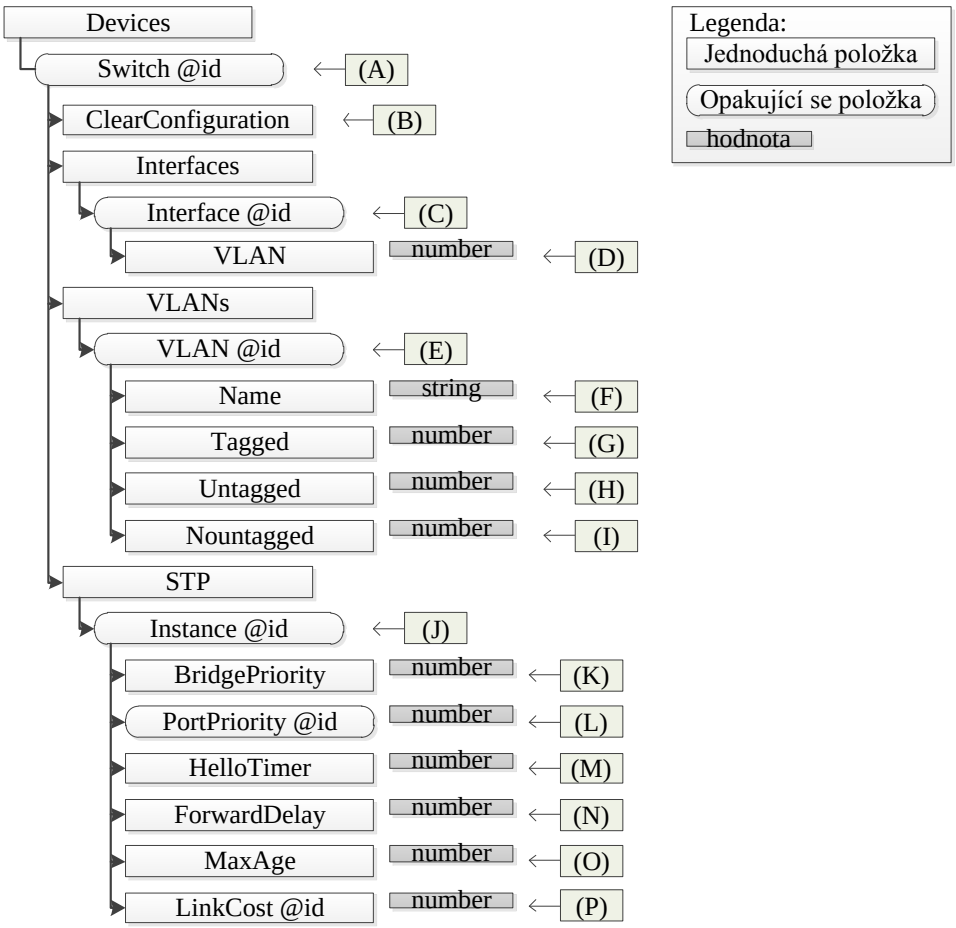
Vzhledem k instancím *Spanning Tree* jsou informace uloženy ve třídě pro danou instanci STP. Metody byly následně upraveny tak, aby pracovaly s jednotlivými strukturami

odděleně v závislosti na kontextu VLAN. Instance jsou uloženy ve vektoru, kde poskytují rychlý přístup. Minimální paměťové nároky jsou zajištěny vektorem registrovaných instancí s indexy na ně. Jeden STP modul tedy obsluhuje všechny instance STP.

6.6 Konfigurace

Konfigurace simulace probíhá po vzoru zařízení *AnsaRouter*. Simulace se nastaví modulem, který čte konfigurační parametry ze souboru ve formátu *XML*. Tento soubor obsahuje nastavení všech zařízení a modul vybere podle deskriptoru sítě (soubor `.ned`) část, která odpovídá právě konfigurovanému zařízení. Tuto část projde a vybere z ní údaje potřebné pro simulaci.

Strukturu konfigurace popisuje obrázek 6.5.



Obrázek 6.5: Struktura konfigurace switche v XML

(a) Definuje sekci pro dané zařízení, `id` je určující řetězec, jež sekci identifikuje.

(b) Příkaz, který zabrání vytvoření implicitní konfigurace. (vizte níže)

(c) Definice rozhraní, `id` určuje index portu ve vektoru portů. Při zápisu topologie je proto vhodné ručně určovat indexy připojených portů.

(d) Přiřazení *PortVID*. Tedy VLAN kontextu pro přijaté neoznačené rámce.

(e) Definice VLAN, `id` určuje číslo VLAN. Nepoužité VLAN dostanou hodnotu 0. Paměťový prostor se omezí na nejvyšší nutný, dle nejvyšší zadané VLAN.

(f) Nepovinné jméno VLAN.

(g) Přiřazení portů, na kterých je daná VLAN povolena. Také udává chování při odesílání rámců v kontextu této VLAN pomocí těchto portů – tedy označit.

(h) Shodná funkce jako předchozí s tím rozdílem, že rámce se právě neoznačují. Také se zde neurčuje povolení neoznačeného rámce. To zajišťuje *PortVID* na rozhraní.

(i) Odebírá přiřazení portů bez označování k dané VLAN. Používá se pro odebírání implicitního nastavení.

(j) Definice instancí STP. `id` je číslo instance odpovídající VLAN, nad kterou instance pracuje.

(k) Priorita v *BridgeID* pro danou instanci.

(l) Priorita na daný port pro *PortID*, `id` udává index portu.

(m) Nastavení časovače *Hello Timer*, pro pravidelné odesílání BPDU.

(n) Časovač *Forward Timer*, specifikuje rychlost přechodů mezi stavy portů.

(o) Hodnota časovače *MaxAge*, udává nejvyšší možné stáří informací.

(p) Cena linky připojené k danému portu. `id` je index portu.

Během studia simulačního nástroje a již vytvořených simulačních modulů byla zjištěna nepříjemná vlastnost. Pokud neexistuje konfigurační soubor, který by nastavil simulaci, program oznámí chybu, že nenalezl konfiguraci a simulace se ukončí. Toho jsem se vyvaroval implicitní konfigurací, do které je zařízení uvedeno při jejím načtení. Díky tomu je možné pracovat s konfiguracemi, které jsou postaveny na základním nastavení a předpokládají jisté původní parametry. Dále je díky tomu možné toto zařízení zkoumat bez extrémní znalosti implementace a nastavení simulace. Proto, pokud někdo bude pokračovat ve vývoji dalších funkcí na zařízení, které vytvářím, nesetká se s tímto popsaným problémem.

Naopak, při ručním nastavování simulace je nutné s tímto chováním počítat. Pokud chci přenastavit některé nevýluční parametry s implicitními, musím tyto nevýluční zrušit. Jedná se například o přiřazení *untagged* VLAN na rozhraní. Implicitně je na všechna rozhraní aplikována VLAN 1 jako *untagged*. Záměr je nastavit jen VLAN 2 *untagged* na rozhraní 1-3, proto je do konfigurace nezbytné přidat příkazy pro zrušení přiřazení VLAN 1 na těchto rozhraních.

Z příkladu v tabulce 6.1 je vidět, že pro ruční konfiguraci není tento způsob načítání příliš pohodlný. A pokud si této vlastnosti nejsme vědomi, v simulaci se pravděpodobně vyskytne neočekávané chování.

Proto existuje příkaz `ClearConfig`, který zabraňuje aplikaci, implicitní konfigurace. Je však nutné nakonfigurovat všechny potřebné parametry, aby simulace mohla běžet správně.

Ukázka konfigurace

```
<VLAN id=1>
    <Nountagged>1</Nountagged>
    <Nountagged>2</Nountagged>
    <Nountagged>3</Nountagged>
</VLAN>
<VLAN id=2>
    <Untagged>1</Untagged>
    <Untagged>2</Untagged>
    <Untagged>3</Untagged>
</VLAN>
```

Tabulka 6.1: Ukázka přenastavení implicitních parametrů

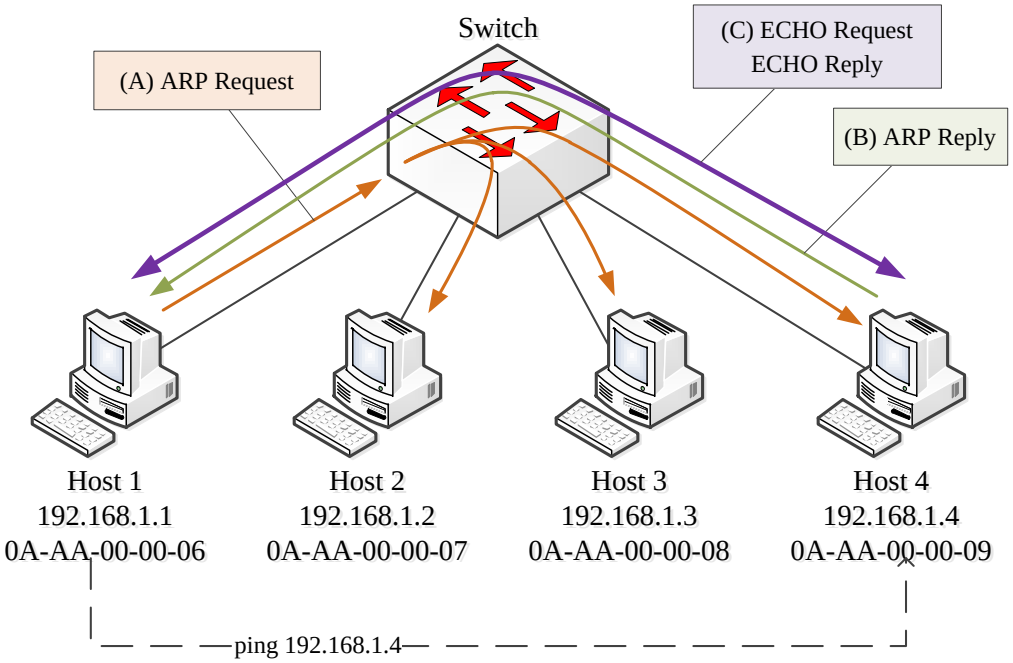
Dále je z příkladu 6.1 patrné, že zadávání rozhraní/VLAN po jednom by bylo mnoho psaní. Proto je lze při větším výskytu zadat jako seznam. Například `<Tagged>1,2,3,4</Tagged>`.

7 Simulace

7.1 Přepínač bez konfigurace

Při použití zařízení *AnsaSwitch* bez konfigurace funguje jako reálné zařízení. Přepínač se nastaví implicitní konfigurací, která poskytuje základní přepínání v rámci VLAN 1. To však nedovoluje označovat rámce (*tagged*). Funkce přepínání se proto jeví, jako kdyby nebyla prováděna v rámci žádné VLAN.

Na obrázku 3.2 je naznačen průběh simulace.



Obrázek 7.1: Průběh simulace základního přepínání dle 3.1

(a) Host 1 provádí příkaz `ping 192.168.1.4`, což je IP adresa Host4. Proto odesílá `ARP Request`, který je adresován jako *broadcast*, aby zjistil MAC adresu svého cíle. Po přijetí `ARP Request` přepínačem je šířen všesměrově podle funkce základního přepínání. Naučí se adresu Host 1.

(b) Host 4 odpovídá na přijatý `ARP Request` svým `ARP Reply` se svou zadanou *MAC*

adresou. `ARP Request` je adresován přímo pro Host 1. Proto ho přepínač předá na `Port 0` podle předchozí naučené informace. Naučí se adresu Host 4.

(c) Nyní už obě stanice znají navzájem své MAC adresy a přepínač má v MAC Tabulce naučeny informace o jejich polohách. Další přímá komunikace tedy probíhá jen mezi porty `0` a `3`. Touto „cestou“ proběhne příkaz `ping 192.168.1.4` ve formě zpráv `ECHO Request` a `ECHO Reply`.

Čas [s]	Událost
0	Po inicializaci je přepínač kořenovým uzlem, spouští časovače. Všechny porty jsou role *Designated*
15	Porty přecházejí ze stavu *Discarding* do *Learning*
18	Neúspěšný pokus o `ARP Request`. Porty nemají povoleno přepínat
30	Porty přecházejí ze stavu *Learning* do *Forwarding*
54	Host 1 úspěšný průběh `ARP Request/Reply` a `ECHO Request/Reply`, jak bylo popsáno v úvodu simulace

Tabulka 7.1: Zjednodušený průběh simulace základního směrování.

Průběh simulace odpovídá teoretickému popisu základního přepínání rámců dle 3.1.

7.2 Komunikace ve VLAN topologii

Simulace topologie s nastavenými VLAN, která ověří funkci popsanou v teoretickém popisu přepínání rámců (sekce 3.2). Komunikace bude probíhat v obou VLAN. Bude simulována jako příkaz `ping` mezi určitými koncovými stanicemi.

Přepínání by mělo zajistit logické oddělení komunikací, které pocházejí z jiných VLAN. Prvotní zprávy `ARP Request` s cílovou adresou broadcast budou šířeny všesměrově jen v rámci VLAN, ze které byly odeslány. Následná přímá komunikace již bude probíhat podle naučených adres, protože ty jsou naučeny v souladu s nastavením VLAN, a proto budou povoleny.

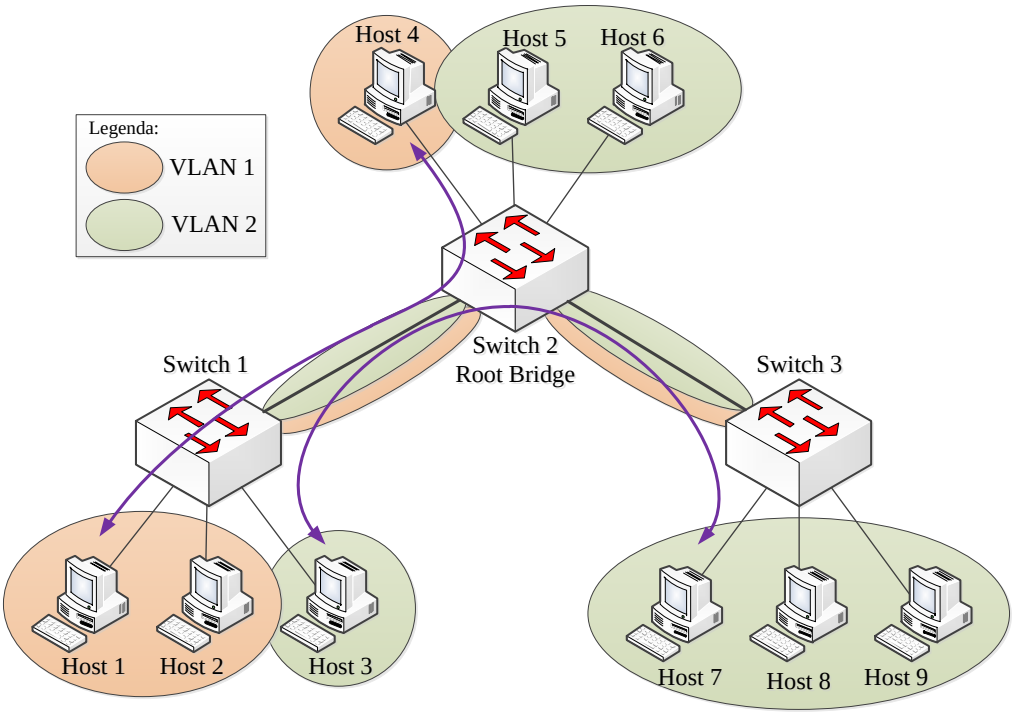
Obrázek 7.2 ukazuje topologii sítě a nastavení VLAN. V tabulce 7.2 je zapsán průběh simulace. Konfiguraci simulace lze nalézt v příloze B.

Simulace přepínání v rámci oddělených VLAN odpovídá popisu v teoretickém úvodu práce, sekce 3.2.

7.3 Konvergence a rekonfigurace STP

V předchozích simulacích již byl naznačen průběh konvergence implementovaného *Spanning Tree* protokolu. Tato simulace ověří konvergenci protokolu s pozměněnými parametry. Dále ověřuje průběh rekonfigurace po pádu linky. Rychlost konvergence a správnost průběhu *Topology Change Notification*. Nakonec ověří rekonfiguraci zpět po obnovení porouchané linky.

Na počátku proběhne volba kořenového uzlu a ustavení aktivní topologie. Přepínače si vymění BPDU zprávy se svými identifikátory. Jejich porovnáním zjistí kořen a ustaví topologii – zvolí jednotlivým portům jejich role podle přijatých identifikátorů (sekce 4.1).



Obrázek 7.2: Ilustrace topologie a nastavení simulace přepínání ve Virtuálních LAN dle 3.2

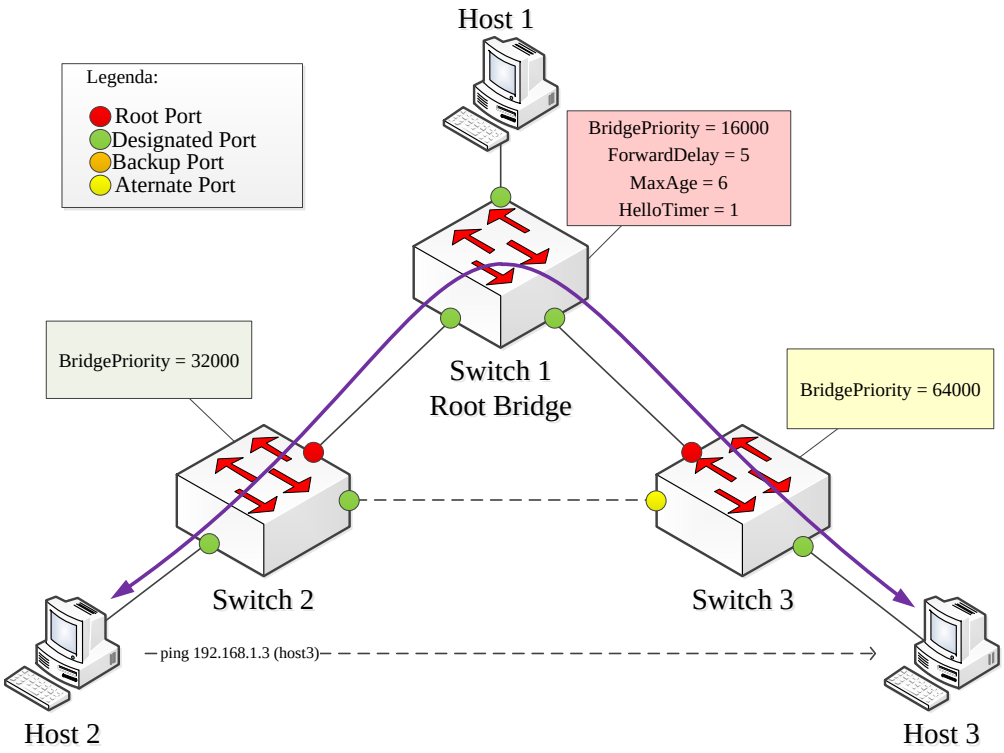
Čas [s]	Událost
0	Po inicializaci jsou všechny přepínače přesvědčeny o tom, že jsou kořeny.
2	Je ustavena aktivní topologie, časovače na portech jsou spuštěny.
15	Porty přecházejí ze stavu *Discarding* do *Learning*
30	Porty přecházejí ze stavu *Learning* do *Forwarding*
54	Host 1 provádí `ping` na Host 4. Tomu předchází `ARP Request`, který se šíří všesměrově v rámci své VLAN 1. Následný ping poté probíhá přímo na trase (A). Shodně probíhá `ping` z Host 7 → Host 3. Samotný ping putuje po trase (B).

Tabulka 7.2: Zjednodušený průběh simulace přepínání v rámci VLAN, dle obrázku 7.2

Po ustavení topologie se spustí časovače a postupně *Root* a *Designated* porty přejdou do stavu *Forwarding*. Pro kontrolu zde nastane komunikace mezi stanicemi. Volba rolí, aktivní topologie a testovací komunikace je naznačena na obrázku 7.3.

Nato je „poškozena“ linka mezi přepínači. Tuto poruchu však ani jeden přepínač nedetekuje přímo, proto musí dojít k vypršení časovače. Tato porucha je horší, než přerušení elektrického spojení kabelu. To je totiž reálné zařízení schopno detekovat ihned.

Po vypršení časovače je na odlehlém přepínači zvolen nový *Root Port*, přes který je odesláno oznámení o změně topologie. Oznámení je postupně potvrzeno všemi přepínači na cestě ke kořeni. Nově změněné porty přecházejí do stavu *Forwarding*. Oznámení je přijato



Obrázek 7.3: Topologie a nastavení simulace pro ověření konvergence aktivní topologie.

kořenovým uzlem. Ten dává zprávu o změně topologie do celé sítě pomocí BPDU s nastaveným parametrem *Topology Change*. Po dokončení rekonfigurace je pro kontrolu provedena komunikace mezi stanicemi. Nová topologie a testovací komunikace je na obrázku 7.4.

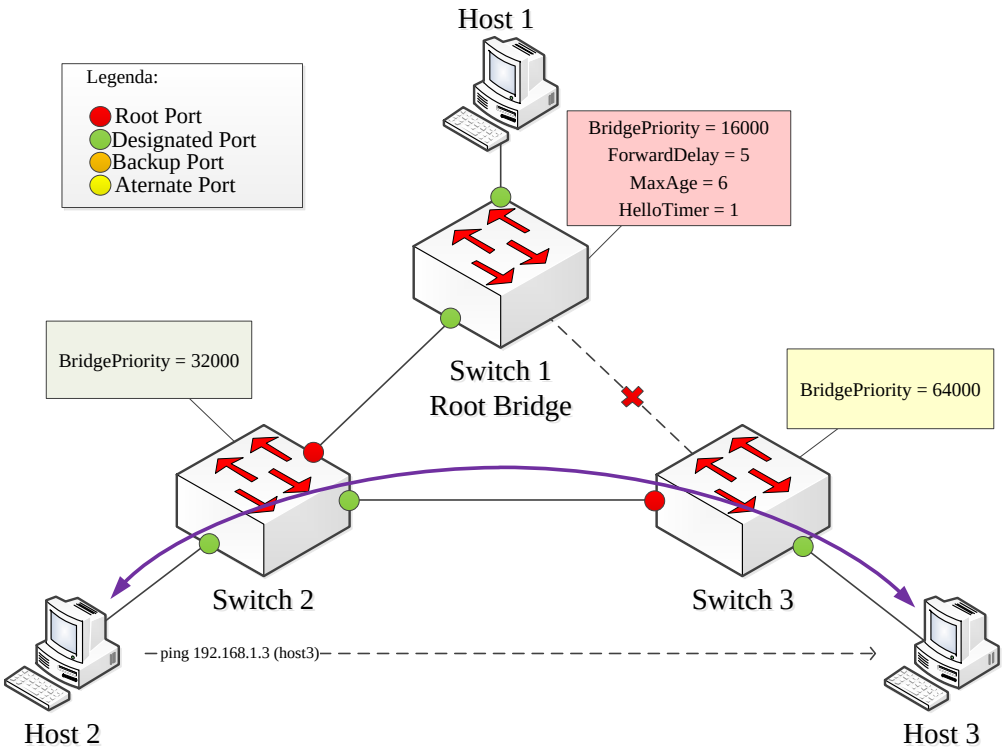
Linka je poté znovu obnovena. Odlehlý přepínač dostává BPDU zprávu od kořene předchozí linkou a posílá oznámení o změně topologie obnoveným *Root* portem. To má stejný průběh jako v předchozím případě. Změní role portů a ty postupně přejdou do stavu *Forwarding*. Nakonec je pro kontrolu provedena komunikace mezi stanicemi. Topologie odpovídá té výchozí.

Změna parametrů je naznačena na obrázku 7.3 případně v tabulce 7.3. Konfiguraci simulace lze nalézt v příloze C.

Parametr	Switch 1	Switch 2	Switch 3
Bridge Priority	16000	32000	64000
Forward Delay	5	–[1]	–
Max Age	6	–	–
Hello Time	1	–	–

Tabulka 7.3: Parametry simulace konvergence STP.

[1] – označuje doporučenou hodnotu dle standardu IEEE 802.1D-1998 [14, str. 64, tabulka 8-4]



Obrázek 7.4: Ilustrace průběhu rekonfigurace aktivní topologie po pádu linky.

Konvergence a rekonfigurace aktivní topologie probíhá dle předpokladu. Rychlejší přechody stavů jsou dány konfigurací a také úpravou implementace, která používá stavy portů z *Rapid-STP*. Parametry definované v kořenovém uzlu se úspěšně rozšířily k ostatním, přičemž ty si uložily své hodnoty pro případ výpadku kořenového uzlu.

Funkce STP v modelu byla dále ověřena na reálných zařízeních. Byla použita zařízení *Cisco Catalyst 2960*. Topologie a nastavení zůstaly zachovány dle simulace. Praktický test ověřoval správnou volbu aktivní topologie při funkční síti a při výpadku. Při konvergenci se dále sledovaly přechody stavů na portech.

Zápis průběhu přechodů na reálném zařízení udává tabulka 7.5. Časy odpovídají hodinám v zařízení. Jelikož model přepínače používá stavy z *Rapid-STP*, je konvergence rychlejší o jeden přechod mezi nimi. V reálném testu trvá konvergence o 5 s déle, což je nastavená hodnota *Forwarding Timer*.

Topologie na funkční síti a po pádu linky, jež přepínače nemohly detekovat přímo, souhlasí se simulací. Obrázek 7.5 ukazuje výpis protokolu *Spanning Tree* v simulaci a výpisy protokolu na reálných zařízeních jsou následující:

Čas [s]	Událost
0	Inicializace
2	Výměna BPDU, volba kořene, nastavení rolí portů a ustavení topologie
7	Porty přecházejí ze stavu *Discarding* do *Learning*
12	Porty přecházejí ze stavu *Learning* do *Forwarding*
20	Úspěšný `ping` Host 2 → Host 3. Komunikace podle obrázku 7.3.
21	Poškození linky mezi Switch 1 ↔ Switch 3.
26	Switch 3 detekuje problém a mění topologii. Při tom odesílá TCN novým *Root* portem na Switch 2.
27	Switch 2 přeposílá TCN ke kořeni a posílá potvrzení pro Switch 1.
28	Switch 1 potvrzuje přijetí TCN.
29	Switch 1 posílá TC do sítě. Všechny přepínače povolují urychlené stárnutí záznamů v MAC tabulce.
31	Porty na Switch 3 přecházejí mezi stavy *Discarding*→*Learning*.
34	Switch 1 ukončuje vysílání TC a přepínače obnovují dobu stárnutí.
36	Porty na Switch 3 přecházejí mezi stavy *Learning*→*Forwarding*.
55	Probíhá komunikace dle obrázku 7.4.
57	Poškozená linka je obnovena.
58	Ustavení předchozí topologie. Switch 3 posílá TCN → Switch 1.
59	Switch 1 posílá potvrzení pro Switch 3.
60	Switch 1 rozesílá TC. Nastává rychlé stárnutí.
63	Porty na Switch 3 přecházejí mezi stavy *Discarding*→*Learning*.
66	Rychlé stárnutí ukončeno.
68	Porty na Switch 3 přecházejí mezi stavy *Learning*→*Forwarding*.
90	Testovací komunikace po původní trase.

Tabulka 7.4: Průběh simulace konvergence a rekonfigurace STP po pádu linky.

Čas [hh:mm:ss]	Událost
00:19:57	Spuštění protokolu
00:20:02	Porty přecházejí ze stavu *Blocking* do *Listening*.
00:20:07	Přechod do stavu *Learning*.
00:20:12	Přechod do stavu *Forwarding*.

Tabulka 7.5: Průběh přechodů stavů na portů na kořenovém přepínači při ověřování laboratoři.

Switch 1

```
SW1# show spanning-tree

VLAN0001
  Spanning tree enabled protocol ieee
  Root ID    Priority    16385
             Address     f4ac.c1c0.8f00
             This bridge is the root.
```

```
            Hello Time   1 sec Max Age 6 sec Forward Delay 5 sec

  Bridge ID Priority    61441 (priority 61440 sys-id-ext 1)
            Address     0021.1c3c.5280
            Hello Time   1 sec Max Age 6 sec Forward Delay 5 sec
            Aging Time  300 sec

Interface          Role Sts Cost   Prio.Nbr Type
------------------ ---- --- ------ -------- ------------------------
Fa0/1              Desg FWD 19     128.1    P2p
Fa0/2              Desg FWD 19     128.2    P2p
Fa0/3              Desg FWD 19     128.3    P2p
```

Switch 2

```
SW2# show spanning-tree

VLAN0001
  Spanning tree enabled protocol ieee
  Root ID   Priority    16385
            Address     f4ac.c1c0.8f00
            Cost        19
            Port        1 (FastEthernet0/1)
            Hello Time   1 sec Max Age 6 sec Forward Delay 5 sec

  Bridge ID Priority    32769 (priority 32768 sys-id-ext 1)
            Address     001c.f9a8.ea80
            Hello Time   2 sec Max Age 20 sec Forward Delay 15 sec
            Aging Time  300 sec

Interface          Role Sts Cost   Prio.Nbr Type
------------------ ---- --- ------ -------- ------------------------
Fa0/1              Root FWD 19     128.1    P2p
Fa0/2              Desg FWD 19     128.2    P2p
Fa0/3              Desg FWD 19     128.3    P2p
```

Switch 3

```
SW3# show spanning-tree

VLAN0001
  Spanning tree enabled protocol ieee
  Root ID   Priority    16385
            Address     f4ac.c1c0.8f00
            Cost        19
            Port        1 (FastEthernet0/1)
            Hello Time   1 sec Max Age 6 sec Forward Delay 5 sec

  Bridge ID Priority    61441 (priority 61440 sys-id-ext 1)
```

```
            Address     0021.1c3c.5280
            Hello Time   2 sec Max Age 20 sec Forward Delay 15 sec
            Aging Time  300 sec

Interface          Role Sts Cost   Prio.Nbr Type
------------------ ---- --- ------ -------- --------------------------
Fa0/1              Root FWD 19     128.1    P2p
Fa0/2              Altn BLK 19     128.2    P2p
Fa0/3              Desg FWD 19     128.3    P2p
```

```
VLAN 1
RootID Priority: 16000
  Address: 0A-AA-00-00-00-01
  This bridge is the Root.
  Hello Time: 1
  Max Age: 6
  Forward Delay: 5
BridgeID Priority: 16000
  Address: 0A-AA-00-00-00-01
  Hello Time: 1
  Max Age: 6
  Forward Delay: 5
Port Flag Role State Cost Priority
----------------------------------------
0  [LF] Desg FWD 19 128
1  [LF] Desg FWD 19 128
2  [LF] Desg FWD 19 128
```

(a) Switch 1

```
VLAN 1
RootID Priority: 16000
  Address: 0A-AA-00-00-00-01
  Cost: 19
  Port: 0
  Hello Time: 1
  Max Age: 6
  Forward Delay: 5
BridgeID Priority: 32000
  Address: 0A-AA-00-00-00-05
  Hello Time: 2
  Max Age: 20
  Forward Delay: 15
Port Flag Role State Cost Priority
----------------------------------------
0  [LF] Root FWD 19 128
1  [LF] Desg FWD 19 128
2  [LF] Desg FWD 19 128
```

(b) Switch 2

```
VLAN 1
RootID Priority: 16000
  Address: 0A-AA-00-00-00-01
  Cost: 19
  Port: 0
  Hello Time: 1
  Max Age: 6
  Forward Delay: 5
BridgeID Priority: 64000
  Address: 0A-AA-00-00-00-09
  Hello Time: 2
  Max Age: 20
  Forward Delay: 15
Port Flag Role State Cost Priority
----------------------------------------
0  [LF] Root FWD 19 128
1  [__] Altr DIS 19 128
2  [LF] Desg FWD 19 128
```

(c) Switch 3

Obrázek 7.5: Výpis protokolu *Spanning Tree* v simulaci jeho konvergence na funkční topologii.

To je stav po konvergenci aktivní topologie na funkční síti. Po přerušení linky, kterého si však přepínače nevšimnou, dojde k vypršení platnosti informací na *Root* portu Switch 3, jež zahájí změnu topologie výběrem nového *Root* portu a odesláním *Topology Change Notification*. Obrázek 7.6 udává výpis *Spanning Tree* protokolu ze simulace a výpisy z laboratoře jsou následující:

Switch 1

```
SW1# show spanning-tree

VLAN0001
  Spanning tree enabled protocol ieee
  Root ID   Priority    16385
            Address     f4ac.c1c0.8f00
            This bridge is the root.
            Hello Time   1 sec Max Age 6 sec Forward Delay 5 sec

  Bridge ID Priority    61441 (priority 61440 sys-id-ext 1)
            Address     0021.1c3c.5280
            Hello Time   1 sec Max Age 6 sec Forward Delay 5 sec
            Aging Time  15 sec
```

```
Interface          Role Sts Cost   Prio.Nbr Type
------------------ ---- --- ------ -------- ------------------------
Fa0/1              Desg FWD 19     128.1    P2p
Fa0/2              Desg FWD 19     128.2    P2p
Fa0/3              Desg FWD 19     128.3    P2p
```

Switch 2

```
SW2# show spanning-tree

VLAN0001
  Spanning tree enabled protocol ieee
  Root ID   Priority    16385
            Address     f4ac.c1c0.8f00
            Cost        19
            Port        1 (FastEthernet0/1)
            Hello Time   1 sec Max Age 6 sec Forward Delay 5 sec

  Bridge ID Priority    32769 (priority 32768 sys-id-ext 1)
            Address     001c.f9a8.ea80
            Hello Time   2 sec Max Age 20 sec Forward Delay 15 sec
            Aging Time  15 sec

Interface          Role Sts Cost   Prio.Nbr Type
------------------ ---- --- ------ -------- ------------------------
Fa0/1              Root FWD 19     128.1    P2p
Fa0/2              Desg FWD 19     128.2    P2p
Fa0/3              Desg FWD 19     128.3    P2p
```

Switch 3

```
SW3# show spanning-tree

VLAN0001
  Spanning tree enabled protocol ieee
  Root ID   Priority    16385
            Address     f4ac.c1c0.8f00
            Cost        38
            Port        2 (FastEthernet0/2)
            Hello Time   1 sec Max Age 6 sec Forward Delay 5 sec

  Bridge ID Priority    61441 (priority 61440 sys-id-ext 1)
            Address     0021.1c3c.5280
            Hello Time   2 sec Max Age 20 sec Forward Delay 15 sec
            Aging Time  15 sec

Interface          Role Sts Cost   Prio.Nbr Type
------------------ ---- --- ------ -------- ------------------------
```

```
Fa0/1              Desg FWD 19     128.1    P2p
Fa0/2              Root FWD 19     128.2    P2p
Fa0/3              Desg FWD 19     128.3    P2p
```

```
VLAN 1
RootID Priority: 16000
  Address: 0A-AA-00-00-00-01
  This bridge is the Root.
  Hello Time: 1
  Max Age: 6
  Forward Delay: 5
BridgeID Priority: 16000
  Address: 0A-AA-00-00-00-01
  Hello Time: 1
  Max Age: 6
  Forward Delay: 5
Port Flag Role State Cost Priority
----------------------------------------
0  [LF] Desg FWD 19 128
1  [LF] Desg FWD 19 128
2  [LF] Desg FWD 19 128
```

(a) Switch 1

```
VLAN 1
RootID Priority: 16000
  Address: 0A-AA-00-00-00-01
  Cost: 19
  Port: 0
  Hello Time: 1
  Max Age: 6
  Forward Delay: 5
BridgeID Priority: 32000
  Address: 0A-AA-00-00-00-05
  Hello Time: 2
  Max Age: 20
  Forward Delay: 15
Port Flag Role State Cost Priority
----------------------------------------
0  [LF] Root FWD 19 128
1  [LF] Desg FWD 19 128
2  [LF] Desg FWD 19 128
```

(b) Switch 2

```
VLAN 1
RootID Priority: 16000
  Address: 0A-AA-00-00-00-01
  Cost: 38
  Port: 1
  Hello Time: 1
  Max Age: 6
  Forward Delay: 5
BridgeID Priority: 64000
  Address: 0A-AA-00-00-00-09
  Hello Time: 2
  Max Age: 20
  Forward Delay: 15
Port Flag Role State Cost Priority
----------------------------------------
0  [LF] Desg FWD 19 128
1  [LF] Root FWD 19 128
2  [LF] Desg FWD 19 128
```

(c) Switch 3

Obrázek 7.6: Výpis protokolu *Spanning Tree* v simulaci jeho konvergence na „poškozené“ topologii.

Ověření v laboratoři ukázalo, že model funkčně odpovídá reálným zařízením. Aktivní topologie byla shodná jak v simulaci, tak v laboratoři. Šíření parametrů protokolu z kořenového uzlu taktéž pracuje shodně. Z porovnání událostí je patrné, že přechody mezi stavy simulace model úspěšně dodržuje ve správném čase.

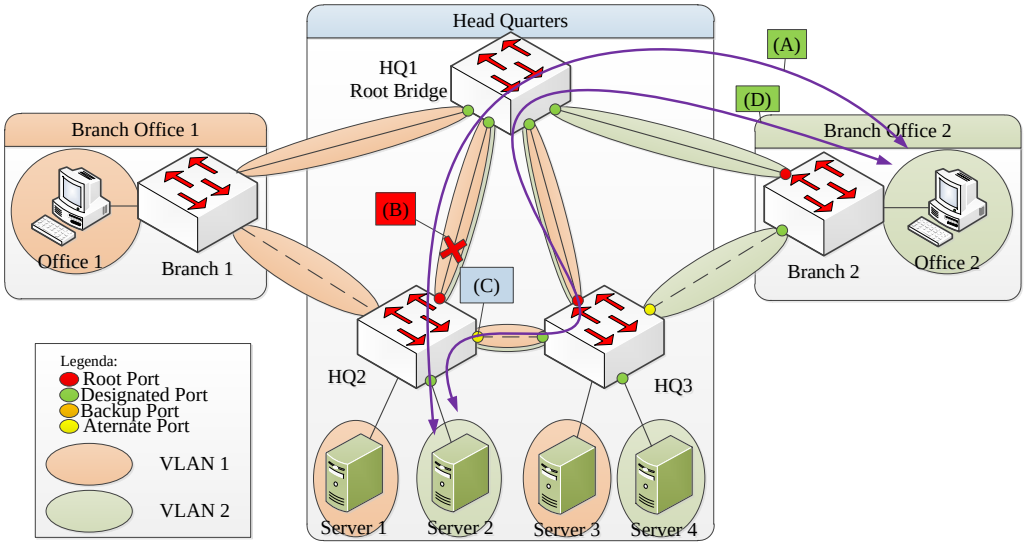
7.4 STP v síti s oddělenými VLAN

Zbývá ještě ověřit model, zda pracuje správně protokol *Spanning Tree* v oddělených virtuálních sítích. Podle problému klasického STP, popsaném v teoretickém popisu (sekce 4.3), byla vytvořena topologie a nastavení na obrázku 7.7. Ta ověří funkčnost oddělených instancí STP podle jednotlivých VLAN.

Konfiguraci této simulace lze nalézt v příloze D.

Jako v předchozích simulacích proběhne konvergence topologie. Následuje průběh zobrazený na obrázku 7.7 s popisem:

- Je provedena kontrolní komunikace mezi Office 2 a Server 2 (v této simulaci, dále jen kontrolní komunikace). Ta proběhne podle naznačené trasy (A) v obrázku.
- Poté dojde k přerušení linky HQ 1 ↔ HQ 2, (B).
- Po vypršení časovače dojde k rekonfiguraci a oznámení změny topologie. Naznačený port (C) se stane novým *Root* portem.
- Po ustálení nové aktivní topologie proběhne kontrolní komunikace po trase (D).



Obrázek 7.7: Ilustrace průběhu komunikace na síti s oddělenými VLAN, podle problému naznačeného v sekci 4.3.

Čas [s]	Událost
0	Inicializace
2	Ustavení topologie
18	Přechod většiny aktivních portů do stavu *Learning*.
19	Přechod zbývajících *Designated* portů do stavu *Learning*. Opožděné porty byly zvoleny později.
32	Přechod většiny do stavu *Forwarding*.
33	Opožděné do stavu *Forwarding*.
62	Kontrolní komunikace trasa (A).
63	Pád linky *HQ 1*↔*HQ 2*, (B).
84	Detekce výpadku, rekonfigurace (C), TCN.
85	Opakování TCN.
86	*HQ 1* potvrzuje TCN a odesílá TC, urychlené stárnutí.
99	Změněné porty přecházejí do stavu *Learning*.
100	Obnovení rychlosti stárnutí položek. Dlouhá doba stárnutí je dána opakovaným přenosem TCN, na které bylo odpovězeno později.
100	Kontrolní komunikace proběhla neúspěšně, protože změněné porty jsou v tento okamžik stále ve stavu *Learning*.
114	Změněné porty přecházejí do stavu *Forwarding*.
135	Kontrolní komunikace po trase (D), nyní již úspěšná.

Tabulka 7.6: Průběh ověření funkce instancí STP na oddělených VLAN.

Simulace ukazuje, že instance protokolu STP pracují dle předpokladu. To bylo ověřeno na příkladu z teoretického úvodu (sekce 4.3).

7.5 Komunikace a Aktivní Topologie na síti VUT

Model sítě VUT, který sestává z nastavení (`.xml`) a zapojení (`.ned`), měla poskytnou práce Analýza toků sítě VUT. Z vyjádření kolegy, jež se prací na modelu sítě VUT zabývá, je patrné, že vytvořit zapojení v modelu nelze bezpečně určit.

> *Analýza konfigurací zařízení byla na vybraných vstupních datech provedena v pořádku a v simulačním modelu jsou k dispozici všechna zařízení a jejich konfigurace. Při spojování jednotlivých zařízení, které již je nutné provést manuálně, jsme však narazili na problém s nedostatkem informací pro správné určení přiřazení linek k daným portům na zařízeních.*
>
> *Spojení zařízení lze provést na základě společných či podobných informací – IP adresy, číslo VLAN, OSPF oblast a především určení druhé strany linky v popisu rozhraní. S popisem jednotlivých linek se setkáváme především u významných uzlů, jako jsou např.* ***`hp-ant`****,* ***`hp-fme`****,* ***`hp-vev`****,* ***`hp-kou`*** *atd. V popisu bývá uvedena informace, jež blíže určí cíl zapojení. Je to název zařízení, ke kterému linka vede, nebo poukáže na to, že se jedná o spojení mimo páteřní síť.*
>
> *Problém však nastává právě u zařízení, která můžeme označit jako koncová. Ta jsou na rozhraní páteřní sítě a jejích jednotlivých podsítí. Na těchto zařízeních již nejsou často vůbec daná rozhraní popsána a určení portu pro příchozí spojení tak není možné. Např. port A18 na zařízení hp-ant má v popisu* ***`hp-ant4(strizna)`****. Na tomto portu je připojena linka k zařízení* ***`hp-ant4`****. Na odlehlém zařízení se však žádný popisek odkazující zpět na* ***`hp-ant`*** *nevyskytuje.*
>
> *Dalším způsobem určení portu může být společné číslo VLAN na portech. Zde se bohužel setkáváme často s tím, že daná VLAN se vyskytuje na více portech. Opět je těžké určit, který z portů v rozsahu je ten správný. Využití IP adres v tomto případě také selhává, protože se jedná o přepínače. Na nich jsou uvedeny IP adresy na portech jen zřídka a u VLAN je nemůžeme spolehlivě využít ze stejného důvodu jako čísla VLAN.*
>
> *V síti tedy lze nalézt určitá spojení, ale jejich počet není dostatečný k tomu, aby byl vytvořen spolehlivý model sítě VUT. Pro získání takového modelu je nutné mít k daným zařízením seznam portů a k nim přiřazené koncové body.*

Vzhledem k výše uvedenému důvodu byla provedena analýza nastavení STP v síti dle dodaných konfiguračních souborů. Bylo zjištěno nastavení priorit sdružených linek (Trunk), které zvýhodňuje jejich použití v aktivní topologii. Za velmi vhodné nastavení považuji vypnutí protokolu MST na některých zařízeních. Vzhledem k existenci kompatibility MSTP-RSTP bude funkce aktivní topologie zachována a zároveň se dosáhne úspory výpočetních prostředků.

Dalším zjištěným nastavením je *bpdu-filter*, který chrání aktivní topologii před příjmem BPDU z okrajových portů. Tím zabraňuje útoku na STP.

V nastavení však není uvedena optimalizace časovačů, která by zajistila rychlejší konvergenci a rekonfiguraci sítě. V případě, že by z nějakého důvodu nefungoval mechanismus *Proposal/Agreement*, probíhalo by ustavení topologie v závislosti na časovači *Forwarding Timer*. Bylo by vhodné zvážit snížení hodnot časovačů tak, aby nedocházelo ke zbytečným rekonfiguracím a zároveň se urychlila konvergence.

8 Budoucí vývoj

Aktuální implementace již pomalu směřuje k *Rapid Spanning Tree* protokolu. Avšak ke kompletnímu RSTP vede dlouhá cesta. Bude nutné upravit architekturu stávajícího STP, která by podporovala chovaní na oddělených portech a díky tomu kompatibilitu s případným STP. Následně implementovat chování všech automatů, které jsou v rámci RSTP potřeba. Nedílnou součástí RSTP je Proposal/Agreement, který výrazně urychluje konvergenci. Pro více informací vizte [15, kap. 17].

Dále je vhodné implementovat základní třídu pro rodinu *Spanning Tree* protokolů, ze které by jednotlivé varianty vycházely pomocí dědičnosti. To by díky unifikovanému rozhraní této základní třídy sloužilo k rozložení instancí protokolů jak na jednotlivé porty, tak na celé VLAN.

Po dokončení *Rapid Spanning Tree* protokolu by již neměl být problém vytvořit chování v oddělených VLAN po vzoru *Rapid*-PVST. Což by zajistilo plnou funkčnost na síti s VLAN a zároveň by mělo všechny vlastnosti novějšího protokolu. Pro plnou podporu standardu IEEE 802.1Q je však nutné RSTP zapojit do komplexního *Multiple Spanning Tree* protokolu. Bude vhodné navrhnout mechanismus rozdělení do regionů. Pro účely simulace se klíčovaný HASH příliš nehodí. Dále se vytvoří nové zprávy pro podporu MST protokolu. Tedy hlavní kostry a také jednotlivých instancí, kterých se v jedné zprávě může objevit více. Musí být vytvořeny funkce obsluhy hlavní sdílené kostry a také těch lokálních v rámci regionů.

Pro optimalizaci simulace je vhodné vytvořit pro model přepínače modul `Device Configurator`, který zajistí čtení konfigurace pro dané zařízení a následně spuštění konfigurátorů registrovaných modulů.

Nakonec je nejdůležitějším vylepšením připojení síťové vrstvy, která obsahuje moduly *směrování*, *filtrování provozu* pomocí ACL apod.. Tím se z přepínače stane *Vícevrstvý přepínač*. Příprava na propojení je již v zařízení implementovaná. Avšak pro plnou funkčnost a jednoduché propojení je potřeba vybrat veškerý provoz, který je nutné předávat vyšším vrstvám. Následně se musí upravit *Forwarding Process* a *MAC Table*, aby vybraný provoz přepínaly na vyšší vrstvu. Naopak přijatý provoz z vyšší vrstvy musí být správně zasazen do kontextu VLAN a odeslán dle stávajících pravidel přepínání.

9 Závěr

Díky přednášce pana Ing. Iva Hažmuka jsem si vytvořil představu o síti VUT. Podle toho byly vybrány prvky pro implementaci do simulačního modelu.

Studiem standardů technologií spojených s linkovou vrstvou a přepínanými sítěmi jsem získal množství velmi cenných znalostí. Prohloubil jsem povědomí o Virtuálních lokálních sítích, jejich funkci a nastavení. Další cenné informace se vztahují k protokolu Spanning Tree, jež má několik variant. Ty se v některých zdrojích pletou. Proto byly v této práci varianty protokolu popsány z více zdrojů a byl vypracován jejich korektní přehled.

Práce byla vytvořena v rámci výzkumné skupiny ANSA, kde jsem načerpal mnoho cenných zkušeností. Vytvářený model byl zasazen do komplexního balíků modelu ANSA-INET. Spolu s dalšími modely poskytují velmi užitečný nástroj pro simulaci sítí. Vývoj jednotlivých modelů probíhal v simulačním nástroji OMNeT++. To mi umožnilo poznat tento nástroj velmi podrobně a osvojit si programovací techniky, jež se při vytváření modelu v tomto nástroji používají. Simulátor OMNeT++ je vhodný například i pro agentní systémy. Jeho znalost je tedy velkým přínosem.

Model byl ověřen jak vůči teoretickému předpokladu, tak proti reálnému zapojení. Ověření bylo úspěšné. Vytvořil jsem si představu o možnostech a přesnosti simulace, která může být při správném přístupu překvapivě velká. Tato práce úzce navazuje na projekt Analýzy toků v síti VUT, která je hlavním podkladem pro nastavení a topologii VUT. Model poskytne vhodný prostředek dalším simulacím sítí, kde je nutno zkoumat chování linkové vrstvy. Implementace je také základem pro rozšíření funkcí přepínače. Cílem je dokončit přechod na novější protokol Rapid Spanning Tree.

Analýzou konfigurací sítě VUT bylo zjištěno nastavení priorit na sjednocených linkách. Ty jsou tímto zvýhodněny v aktivní topologii. Není však optimalizováno nastavení časovačů. Pokud tedy z nějakého důvodu na síti nebude fungovat mechanizmus Proposal/Agreement, konvergence bude velmi zdlouhavá. Bylo by vhodné zvážit optimalizaci časovačů tak, aby nedocházelo ke generování zbytečného provozu a zároveň aby při náhodné ztrátě BPDU nedocházelo k rekonfiguraci. Pro rychlost konvergence navrhuji snížit hodnoty časovačů.

Vynucením funkce RSTP místo MSTP na některých prvcích, se vhodně šetří výpočetní prostředky. Z hlediska bezpečnosti je také na určitých zařízeních nastaven *bpdu-filter*, který zabraňuje přijímání BPDU zpráv z okrajových portů. Díky tomu je síť chráněna proti útokům na protokol STP.

Všechna řešení jsou dostupná na přiloženém disku, kde jsou také uloženy volně dostupné dokumenty formátu pdf obsažené v bibliografii v podobě, v jaké jsem je použil. Práce jako celek mi podala velice cenné znalosti a zkušenosti k tématu simulací a přepínaných sítí. Práce se mi velmi líbila.

Literatura

[1] Cisco Systems, I.: Inter-Switch Link and IEEE 802.1Q Frame Format. [online], 2006-08-25 [vid. 2011-05-05].
URL http://www.cisco.com/en/US/tech/tk389/tk689/technologies_tech_note09186a0080094665.shtml

[2] Cisco Systems, I.: Understanding Rapid Spanning Tree Protocol (802.1w). 2006-11-24 [online], [vid. 2011-04-18].
URL http://www.cisco.com/en/US/tech/tk389/tk621/technologies_white_paper09186a0080094cfa.shtml

[3] Cisco Systems, I.: Configuring Spanning Tree. [online], [vid. 2011-05-05].
URL http://www.cisco.com/en/US/docs/switches/lan/catalyst4000/7.4/configuration/guide/spantree.html

[4] Cisco Systems, I.: Configuring Spanning Tree PortFast, BPDU Guard, BPDU Filter, UplinkFast, BackboneFast, and Loop Guard. [online], [vid. 2011-05-05].
URL http://www.cisco.com/en/US/docs/switches/lan/catalyst4000/7.4/configuration/guide/stp_enha.html

[5] Cisco Systems, I.: Multiple Instance STP (MISTP) / 802.1s: Introduction. [online], [vid. 2011-05-05].
URL http://www.cisco.com/en/US/tech/tk389/tk621/tk844/tsd_technology_support_sub-protocol_home.html

[6] Cisco Systems, I.: Per VLAN Spanning Tree Plus (PVST+). [online], [vid. 2011-05-05].
URL http://www.cisco.com/en/US/tech/tk389/tk621/tk847/tsd_technology_support_sub-protocol_home.html

[7] Cisco Systems, I.: Per VLAN Spanning Tree (PVST). [online], [vid. 2011-05-05].
URL http://www.cisco.com/en/US/tech/tk389/tk621/tk846/tsd_technology_support_sub-protocol_home.html

[8] Cisco Systems, I.: Understanding Multiple Spanning Tree Protocol (802.1s). 2007-04-17 [online], [vid. 2011-05-06].
URL http://www.cisco.com/en/US/tech/tk389/tk621/technologies_white_paper09186a0080094cfc.shtml

[9] Cisco Systems, I.: Virtual Router Redundancy Protocol. [online], [vid. 2011-05-21].
URL https://nes.fit.vutbr.cz/ansa/uploads/Nesim/Hazmuk_paterni_sit.pdf

[10] Danko, M.: *Model směrovacího protokolu OSPF v simulátoru OMNeT++*. 2009, bakalářská práce.

[11] H. Krawczyk, R. C., M. Bellare: RFC 2104: HMAC: Keyed-Hashing for Message Authentication. [online], 1998-01 [vid. 2011-05-06].
URL http://datatracker.ietf.org/doc/rfc2104/

[12] Harrington, D.: RFC 2271: An Architecture for Describing SNMP Management Frameworks. [online], 1998-01 [vid. 2011-05-06].
URL http://datatracker.ietf.org/doc/rfc2271/

[13] Hažmuk, I.: Páteřní síť VUT. [online], 2009-12-02 [vid. 2011-05-21].
URL https://nes.fit.vutbr.cz/ansa/uploads/Nesim/Hazmuk_paterni_sit.pdf

[14] IEEE: *IEEE Standard for Local and metropolitan area networks: Media Access Control (MAC) Bridges*. IEEE, 1998.

[15] IEEE: *IEEE Standard for Local and metropolitan area networks: Media Access Control (MAC) Bridges*. IEEE, 2004.

[16] IEEE: *IEEE Standard for Local and metropolitan area networks: Virtual Bridged Local Area Networks*. 2005.

[17] ITU-T: E.164: The International public telecomunication nubering plan. [online], [vid. 2011-04-11].
URL http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-E.164-201011-I!!PDF-E&type=items

[18] Kozierok, C.: TCP/IP Guide: Circuit Switching and Packet Switching Networks. 2005-09-20 [online], [vid. 2011-04-11].
URL http://www.tcpipguide.com/free/t_CircuitSwitchingandPacketSwitchingNetworks.htm

[19] Masopust, T.; Křivka, Z.: *Grafové Algoritmy*. 2010.

[20] Perlman, R.: *Interconnections: Bridges, Routers, Switches and Internetworking Protocols*. Addison-Wesley, druhé vydání, 2001, ISBN 0-201-63448-1.

[21] Rybová, V.: *Modelování a simulace návrhových vzorů směrování v počítačových sítích*. 2009, bakalářská práce.

[22] Scherfel, P.: *Simulace chování sítě na základě analýzy konfiguračních souborů aktovních síťových zařízení*. 2009, bakalářská práce.

[23] Suchomel, T.: *Rozšíření simulátoru OMNeT++ o filtrovací pravidla ACL*. 2009, bakalářská práce.

[24] Varga, A.: OMNeT++ 4.1 Manual [online].
http://omnetpp.org/doc/omnetpp41/manual/usman.html, [vid. 2011-01-08].

Seznam obrázků

Seznam tabulek

Seznam zkratek

ANSA – Automated Network-wide Security Analysis

ACL – Access Control List

IEEE – Institute of Electrical and Electronics Engineers

VLAN – Virtual Local Area Network

STP – Spanning Tree Protocol

UPS – Uninterruptable Power Supply

VRRP – Virtual Router Redundancy Protocol

RIP – Routing Information Protocol

OSPF – Open Shortest Path First

IP – Internet Protocol

BGP – Border Gateway Protocol

MAC – Media Access Control

OUI – Organizationally Unique Identifier

PSTN – Public Switched Telephone Network

ITU – Internation Telecomunication Union

MTU – Maximum Transfer Unit

VID – VLAN Identifier

BPDU – Bridge Protocol Data Unit

TC – Topology Change

TCN – TC Notification

TCA – TC Acknowledge

RSTP – Rapid STP

PVST – Per-VLAN STP

MISTP – Multiple Instance STP

MSTP – Multiple STP

CIST – Common and Internal Spanning Tree

IST – Internal Spanning Tree

CST – Common Spanning Tree

MSTI – MST Instance

XML – eXtensible Markup Language

ARP – Address Resolution Protocol

Seznam příloh

A – Obsah CD

B – Konfigurační soubor simulace VLAN topologie

C – Konfigurační soubor simulace STP konvergence

D – Konfigurační soubor simulace STP v oddělených VLAN

A Obsah CD

/ansainet/* – revize zdrojových kódu projektu ANSAINET z 23.5. 2011

/ansainet/src/ansa/switch – zdrojové soubory zařízení ANSASwitch, implementovaného v rámci této práce

/ansainet/examples/ansaExamples/switchtest/* – simulační scénáře a konfigurační soubory potřebné pro dané simulace

/text/ – revize zdrojových souborů textu této práce z 23.5. 2011

/projekt.pdf – vysázený text práce

/readme.txt – popis překladu a spuštění projektu ANSAINET

B Konfigrační soubor simulace VLAN topologie

```
<Devices>
 <Switch id="Switch1">
  <Hostname>Foo</Hostname>
  <Interfaces>
   <Interface id="0">
    <Name>SW2</Name>
    <VLAN>1</VLAN>
   </Interface>
   <Interface id="1">
    <Name>host 1</Name>
    <VLAN>1</VLAN>
   </Interface>
   <Interface id="2">
    <Name>host 2</Name>
    <VLAN>1</VLAN>
   </Interface>
   <Interface id="3">
    <Name>host 3</Name>
    <VLAN>2</VLAN>
   </Interface>
  </Interfaces>
  <VLANs>
   <VLAN id="1">
    <Name>Study</Name>
    <Tagged>0</Tagged>
    <Nountagged>3</Nountagged>
   </VLAN>
   <VLAN id="2">
    <Name>Employee</Name>
    <Tagged>0</Tagged>
    <Untagged>3</Untagged>
   </VLAN>
  </VLANs>
 </Switch>
 <Switch id="Switch2">
  <Hostname>Bar</Hostname>
  <Interfaces>
   <Interface id="0">
    <Name>SW1</Name>
    <VLAN>1</VLAN>
   </Interface>
   <Interface id="1">
    <Name>SW3</Name>
    <VLAN>1</VLAN>
   </Interface>
   <Interface id="2">
    <Name>host 4</Name>
    <VLAN>1</VLAN>
   </Interface>
   <Interface id="3">
    <Name>host 5</Name>
    <VLAN>2</VLAN>
   </Interface>
   <Interface id="4">
    <Name>host 6</Name>
    <VLAN>2</VLAN>
   </Interface>
  </Interfaces>
  <VLANs>
   <VLAN id="1">
    <Name>Study</Name>
    <Tagged>0</Tagged>
    <Tagged>1</Tagged>
    <Nountagged>3</Nountagged>
    <Nountagged>4</Nountagged>
   </VLAN>
   <VLAN id="2">
    <Name>Employee</Name>
    <Tagged>0</Tagged>
    <Tagged>1</Tagged>
```

```
    <Untagged>3</Untagged>
    <Untagged>4</Untagged>
   </VLAN>
  </VLANs>
  <STP>
   <Instance id="1">
    <BridgePriority>
     16000
    </BridgePriority>
   </Instance>
   <Instance id="2">
    <BridgePriority>
     16000
    </BridgePriority>
   </Instance>
  </STP>
 </Switch>
 <Switch id="Switch3">
  <Hostname>redrum</Hostname>
  <ClearConfig />
  <Interfaces>
   <Interface id="0">
    <Name>SW2</Name>
    <VLAN>1</VLAN>
   </Interface>
   <Interface id="1">
    <Name>host 7</Name>
    <VLAN>2</VLAN>
   </Interface>
   <Interface id="2">
    <Name>host 8</Name>
    <VLAN>2</VLAN>
   </Interface>
   <Interface id="3">
    <Name>host 9</Name>
    <VLAN>2</VLAN>
   </Interface>
  </Interfaces>
  <VLANs>
   <VLAN id="1">
    <Name>Study</Name>
    <Tagged>0</Tagged>
   </VLAN>
   <VLAN id="2">
    <Name>Employee</Name>
    <Tagged>0</Tagged>
    <Untagged>1</Untagged>
    <Untagged>2</Untagged>
    <Untagged>3</Untagged>
   </VLAN>
  </VLANs>
 </Switch>
</Devices>
```

C Konfigrační soubor simulace STP konvergence

```
<Devices>
 <Switch id="Switch1">
  <STP>
   <Instance id="1">
    <BridgePriority>
     16000
    </BridgePriority>
    <ForwardDelay>5</ForwardDelay>
    <MaxAge>6</MaxAge>
    <HelloTimer>1</HelloTimer>
   </Instance>
  </STP>
 </Switch>
 <Switch id="Switch2">
  <STP>
   <Instance id="1">
    <BridgePriority>
     32000
    </BridgePriority>
   </Instance>
  </STP>
 </Switch>
 <Switch id="Switch3">
  <STP>
   <Instance id="1">
    <BridgePriority>
     64000
    </BridgePriority>
   </Instance>
  </STP>
 </Switch>
</Devices>
```

D Konfigrační soubor simulace STP v oddělených VLAN

```
<Devices>
 <Switch id="hq1">
  <Interfaces>
   <Interface id="0">
    <Name>hq1-hq2</Name>
    <VLAN>1</VLAN>
   </Interface>
   <Interface id="1">
    <Name>hq1-hq3</Name>
    <VLAN>1</VLAN>
   </Interface>
   <Interface id="2">
    <Name>hq1-branch1</Name>
    <VLAN>1</VLAN>
   </Interface>
   <Interface id="3">
    <Name>hq1-branch2</Name>
    <VLAN>2</VLAN>
   </Interface>
  </Interfaces>
  <VLANs>
   <VLAN id="1">
    <Name>Accounting</Name>
    <Tagged>0</Tagged>
    <Tagged>1</Tagged>
    <Tagged>2</Tagged>
    <Nountagged>0</Nountagged>
    <Nountagged>1</Nountagged>
    <Nountagged>2</Nountagged>
    <Nountagged>3</Nountagged>
   </VLAN>
   <VLAN id="2">
    <Name>Development</Name>
    <Tagged>0</Tagged>
    <Tagged>1</Tagged>
    <Tagged>3</Tagged>
    <Nountagged>2</Nountagged>
   </VLAN>
  </VLANs>
  <STP>
   <Instance id="1">
    <BridgePriority>
     16000
    </BridgePriority>
   </Instance>
   <Instance id="2">
    <BridgePriority>
     16000
    </BridgePriority>
   </Instance>
  </STP>
 </Switch>

 <Switch id="hq2">
  <Interfaces>
   <Interface id="0">
    <Name>hq2-hq1</Name>
    <VLAN>1</VLAN>
   </Interface>
   <Interface id="1">
    <Name>hq2-hq3</Name>
    <VLAN>1</VLAN>
   </Interface>
   <Interface id="2">
    <Name>hq2-branch1</Name>
    <VLAN>1</VLAN>
   </Interface>
   <Interface id="3">
    <Name>hq2-server1</Name>
    <VLAN>1</VLAN>
```

```
  </Interface>
  <Interface id="4">
   <Name>hq2-server3</Name>
   <VLAN>2</VLAN>
  </Interface>
 </Interfaces>
 <VLANs>
  <VLAN id="1">
   <Name>Accounting</Name>
   <Tagged>0</Tagged>
   <Tagged>1</Tagged>
   <Tagged>2</Tagged>
   <Nountagged>0</Nountagged>
   <Nountagged>1</Nountagged>
   <Nountagged>2</Nountagged>
   <Untagged>3</Untagged>
   <Nountagged>4</Nountagged>
  </VLAN>
  <VLAN id="2">
   <Name>Development</Name>
   <Tagged>0</Tagged>
   <Tagged>1</Tagged>
   <Untagged>4</Untagged>
  </VLAN>
 </VLANs>
 <STP>
  <Instance id="1">
   <BridgePriority>
    32000
   </BridgePriority>
  </Instance>
  <Instance id="2">
   <BridgePriority>
    32000
   </BridgePriority>
  </Instance>
 </STP>
</Switch>

<Switch id="hq3">
 <Interfaces>
  <Interface id="0">
   <Name>hq3-hq1</Name>
   <VLAN>1</VLAN>
  </Interface>
  <Interface id="1">
   <Name>hq3-hq2</Name>
   <VLAN>1</VLAN>
  </Interface>
  <Interface id="2">
   <Name>hq3-branch2</Name>
   <VLAN>1</VLAN>
  </Interface>
  <Interface id="3">
   <Name>hq3-server3</Name>
   <VLAN>1</VLAN>
  </Interface>
  <Interface id="4">
   <Name>hq3-server4</Name>
   <VLAN>2</VLAN>
  </Interface>
 </Interfaces>
 <VLANs>
  <VLAN id="1">
   <Name>Accounting</Name>
   <Tagged>0</Tagged>
   <Tagged>1</Tagged>
   <Nountagged>0</Nountagged>
   <Nountagged>1</Nountagged>
   <Nountagged>2</Nountagged>
   <Nountagged>4</Nountagged>
   <Untagged>3</Untagged>
  </VLAN>
  <VLAN id="2">
   <Name>Development</Name>
   <Tagged>0</Tagged>
   <Tagged>1</Tagged>
   <Tagged>2</Tagged>
   <Untagged>4</Untagged>
  </VLAN>
 </VLANs>
 <STP>
  <Instance id="1">
   <BridgePriority>
    32000
   </BridgePriority>
  </Instance>
  <Instance id="2">
   <BridgePriority>
    32000
   </BridgePriority>
  </Instance>
 </STP>
</Switch>

<Switch id="branch1">
```

```
 <Interfaces>
  <Interface id="0">
   <Name>branch1-hq1</Name>
   <VLAN>1</VLAN>
  </Interface>
  <Interface id="1">
   <Name>branch1-hq2</Name>
   <VLAN>1</VLAN>
  </Interface>
  <Interface id="2">
   <Name>branch1-office1</Name>
   <VLAN>1</VLAN>
  </Interface>
 </Interfaces>
 <VLANs>
  <VLAN id="1">
   <Name>Accounting</Name>
   <Tagged>0</Tagged>
   <Tagged>1</Tagged>
   <Untagged>2</Untagged>
   <Nountagged>0</Nountagged>
   <Nountagged>1</Nountagged>
  </VLAN>
 </VLANs>
 <STP>
  <Instance id="1">
   <BridgePriority>
    64000
   </BridgePriority>
  </Instance>
  <Instance id="2">
   <BridgePriority>
    64000
   </BridgePriority>
  </Instance>
 </STP>
</Switch>

<Switch id="branch2">
 <ClearConfig />
 <Interfaces>
  <Interface id="0">
   <Name>branch2-hq1</Name>
   <VLAN>2</VLAN>
  </Interface>
  <Interface id="1">
   <Name>branch2-hq3</Name>
   <VLAN>2</VLAN>
  </Interface>
  <Interface id="2">
   <Name>branch2-office2</Name>
   <VLAN>2</VLAN>
  </Interface>
 </Interfaces>
 <VLANs>
  <VLAN id="2">
   <Name>Development</Name>
   <Tagged>0</Tagged>
   <Tagged>1</Tagged>
   <Untagged>2</Untagged>
   <Nountagged>0</Nountagged>
   <Nountagged>1</Nountagged>
  </VLAN>
 </VLANs>
 <STP>
  <Instance id="1">
   <BridgePriority>
    64000
   </BridgePriority>
  </Instance>
  <Instance id="2">
   <BridgePriority>
    64000
   </BridgePriority>
  </Instance>
 </STP>
</Switch>

</Devices>
```