

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

MODELOVÁNÍ PROTOKOLU IS-IS

DIPLOMOVÁ PRÁCE

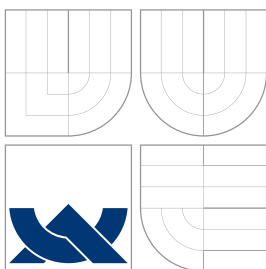
MASTER'S THESIS

AUTOR PRÁCE

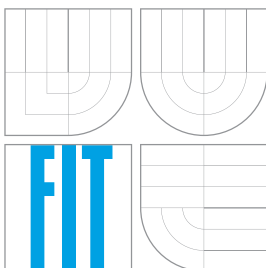
AUTHOR

MARCEL MAREK

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

MODELOVÁNÍ PROTOKOLU IS-IS

MODELLING OF IS-IS PROTOCOL

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE

AUTHOR

MARCEL MAREK

VEDOUCÍ PRÁCE

SUPERVISOR

Ing. VLADIMÍR VESELÝ

BRNO 2012

Abstrakt

Výtah (abstrakt) práce v českém jazyce.

Abstract

Výtah (abstrakt) práce v anglickém jazyce.

Klíčová slova

IS-IS, ANSA, OMNeT++, směrovací protokol

Keywords

IS-IS, ANSA, OMNeT++, routing protocol

Citace

Marcel Marek: Modelování protokolu IS-IS, diplomová práce, Brno, FIT VUT v Brně, 2012

Modelování protokolu IS-IS

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Vladimíra Veselého

.....

Marcel Marek

31. srpna 2012

Poděkování

Zde je možné uvést poděkování vedoucímu práce a těm, kteří poskytli odbornou pomoc.

© Marcel Marek, 2012.

Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.

Obsah

1	Úvod	3
2	OMNeT++	4
3	IS-IS	5
3.1	OSI adresování	5
3.1.1	NSEL	6
3.1.2	System-ID	6
3.1.3	Area ID	6
3.2	Typy zpráv	6
3.2.1	Obecná hlavička	7
3.2.2	Hello zprávy	8
3.2.3	LAN Hello zprávy	8
3.2.4	Point-to-point Hello zprávy	9
3.2.5	LSP	9
3.2.6	Sequence Numbers PDU	10
3.3	Link-state databáze	12
3.3.1	Místní výpočet	12
3.4	Číslování LSP	12
3.5	Životnost LSP	13
3.6	Periodické aktualizace	13
3.7	Expirace LSP	13
3.8	Zaplavování	13
3.9	Mesh Groups	14
3.10	Odstranění LSP	14
3.11	Volba DIS	14
3.12	Pseudonode a DIS	15
3.12.1	Vyvážení počtu sousedství na rozsáhlých LAN	15
3.12.2	Problém synchronizace	15
3.12.3	Pseudonode	15
3.12.4	Pseudonode-ID	15
3.12.5	Modelování link-state databáze	16
3.12.6	Volba DIS	16
3.13	Synchronizace databází	16
3.13.1	Synchronizace databází na point-to-point	17
3.14	Fragmentace	17
3.14.1	Hello zprávy	18
3.14.2	Sequence Number Packets	18

3.14.3	LSP	18
3.15	Výpočet nejkratších cest	19
3.15.1	SPF	19
4	Implementace protokolu	21
4.1	Konfigurace	21
4.2	Plánování zpráv	22
4.3	Hello zprávy	22
4.4	LSP	24
4.4.1	Generování	24
4.4.2	Periodické zasílání	24
4.4.3	Refresh LSP	24
4.4.4	Handle LSP	25
4.4.5	LSP Purge	25
4.5	Sequence Numbers PDU	25
4.5.1	CSNP	25
4.6	SPF	25
5	Závěr	27

Kapitola 1

Úvod

Tato technická zpráva stručně popisuje simulační prostředí OMNeT++, specifikaci protokolu IS-IS dle ISO 10589, případně jeho rozšíření dle RFC 1195. Následně mapuje průběh implementace tohoto protokolu ve zmíněném simulačním prostředí OMNeT++ s rozšířením INET.

Kapitola 2

OMNeT++

OMNeT++ je simulační prostředí pro modelování nejrůznějších jevů. S rozšířením INET se z něj stává simulační nástroj pro modelování síťové komunikace. ++ v jeho názvu napovídá, že je vytvořen v jazyce C++. Základní prvek simulace tvoří modul. Jednotlivé moduly jsou pak spojovány pomocí NED souborů, čímž vytváří entity odpovídající reálným síťovým zařízením. Takto vytvořené entity i samotné moduly je možné mezi sebou propojovat pomocí bran.

Kapitola 3

IS-IS

V této kapitole se seznámíme s vývojem protokolu IS-IS. Představíme si používané formáty zpráv, způsob jejich výměny na různých topologiích.

IS-IS celým názvem *Intermediate System to Intermediate System* je dvou úroňový směrovací protokol běžící na síťové vrstvě modelu ISO/OSI. Každý prvek v síti označuje IS-IS jako *System*. *End System* označuje *koncový* prvek, za který můžeme považovat například PC. Naopak, zařízení podílející se na doručování dat, jsou označovány jako *Intermediate System*. Triviálně více používaný termín pro *Intermediate System* je *router*.

IS se sdružují do oblastí (*area*). Směrování v rámci jedné oblasti označujeme jako *L1 směrování*. Naopak směrování mezi oblastmi jako *L2 směrování*.

V IS-IS rozlišujeme pouze dva typy rozhraní, broadcastové a point-to-point.

Na broadcastových rozhraních jako je Ethernet se nepoužívají unicastové adresy, ale pouze multicastové. IS-IS se snaží, aby každý systém připojený do dané LAN slyšel každou zprávu. Ke komunikaci se všemi *L1 IS* využívá multicastovou MAC adresu 0180:c200:0014 a pro *L2 IS* adresu 0180:c200:0015. Jako zdroj se většinou používá adresa daného Ethernet portu, přes který byla zpráva odeslána. Dále je v ethernetové hlavičce uveden DSAP a SSAP kód 0xFE určující, že se jedná o OSI protokol na obou stranách spojení. Uvnitř takového rámce se nachází nativní IS-IS zpráva, která může být od 27 B do MTU dané linky minus 21 B. 21 B je součet ostatních částí Ethernetového rámce. IS-IS proces musí sám zaručit, že nepřekročí tuto velikost, i kdyby to mělo znamenat rozdělení zprávy do několika rámců, protože Ethernet samotný nepodporuje fragmentaci.

3.1 OSI adresování

IS-IS používá adresní model síťových protokolů OSI. Největší změnou oproti IP adresování je, že IS používá pouze jedinou OSI adresu pro IS, tedy všechna rozhraní. Taková adresa je většinou přiřazena směrovacímu procesu (Cisco IOS), případně virtuálnímu loopback rozhraní (Juniper JUNOS). Můžeme ovšem přiřadit i několik adres – využívá se při migraci oblastí. Nutno podotknout, že všechny adresy platí pro všechna rozhraní daného IS. V OSI adresování se používá mnoho specifických termínů, některé jsme zmínili již na začátku, jiné nikoli. Shrňme si proto ty nejdůležitější. Musí být nakonfigurován alespoň jeden **NET** (*Network Entity Title*) na IS. NET je velký od 3 do 22 B a skládá se z několika částí:

3.1.1 NSEL

NSEL, neboli *Network Selector Label*, je poslední byte v **NET** ???. Jeho hodnota, pro použití v IS-IS, musí být vždy 0. V jiném případě by nedošlo k navázání sousedství mezi dvěma IS. NSEL je obdobou identifikátoru protokolu v IP hlavičce a umožňuje provozování několika služeb na daném NET.

3.1.2 System-ID

Každý link-state protokol musí zajistit unikátní identifikaci všech uzlů v síti pro svoje fungování. Dle specifikace v ISO 10589, System ID může být různé délky od jednoho do osmi bytů. Nicméně současné implementace protokolu IS-IS používají ustálenou velikost 6 B. Konkrétní používaná délka je specifikována v **IIH** (IS-IS Hello) hlavičce v položce *ID Length* velké 1 B. Běžným způsobem jak zajistit unikátnost *System-ID* v síti je použití dekadického zápisu 32 b IP adresy, kde každý byte je zapsán jako trojice cifer. Pokud daná hodnota zabírá v dekadickém zápisu méně než tři cifry, doplní se zleva nulami. Výsledná adresa se pak zapisuje hexadecimálně vždy po dvou bytech oddělených tečkou. Pro IP adresu 192.168.1.1 by *System-ID* vypadalo následovně 1921.6800.1001. Způsob zajištění unikátnosti není nijak definován a je čistě na rozhodnutí administrátora.

3.1.3 Area ID

Area ID může mít od 1 do 13 B. Formát je specifikován v prvním bytu tzv. **AFI** – *Address Family Identifier*, někdy také označován jako *Authority and Format Identifier*.

Mezi dobře známé **AFI** patří například

- 39 DCC (data country code),
- 45 E.164,
- 46 ICD (Internation Code Designator) a
- 49 private addressing.

AFI 49 označující *private-addressing* (*soukromé adresování*) umožňuje používání libovolného formátu a je jakousi obdobou RFC 1918 [?] pro používání privátních adres v IP sítích.

V návaznosti na příklad zápisu v sekci 3.1.2 by celá OSI adresa mohla vypadat takto 49.0001.1921.6800.1001.00. První byte specifikující *private addressing*, následovaný dvěma byty pro *Area-ID*, 6 B *System-ID* a poslední byte *NSEL*, který musí být vždy nulový.

IS-IS používá tři základní typy zpráv. *Hello* zprávy pro objevování sousedů a navazování sousedství. **LSP** - *Link State PDU* pro výměnu informací i síťové topologii a *Sequence Number PDU* pro synchronizaci LSP databází mezi jednotlivými IS. V další části popíšeme dané zprávy a jejich varianty.

3.2 Typy zpráv

Všechny typy zpráv tvoří tři části *obecná hlavička*, hlavička specifická pro daný typ zprávy a TLV část.

3.2.1 Obecná hlavička

Obecná hlavička, neboli IS-IS hlavička předchází všem IS-IS zprávám. Její délka je 8 B a říká příjemci o verzi používaného protokolu, délce hlavičky, maximálním počtu provozovaných oblastí, společně s dalšími IS-IS parametry jako délka *System ID*. Tvoří ji:

NPLID

V prostředí OSI má každý protokol přidělený jednobyteový kód. Pro IS-IS je to hodnota 0x83.

Délka hlavičky

Určuje délku hlavičky v bytech včetně hlavičky pro daný typ PDU.

Verze protokolu

1

Délka *System-ID*

Určuje délku *System-ID* viz 3.1.2.

Rezervované bity

Jedná se o 3 b, při odesílání se nastavují na hodnotu 0 a při přijetí jsou ignorovány.

Typ

Typ (*PDU Type*) je 5-ti bitová položka a specifikuje jeden z následujících typů zprávy:

- Level 1 LAN IS to IS Hello PDU
- Level 2 LAN IS to IS Hello PDU
- Point-to-Point IS to IS Hello PDU
- Level 1 Link State PDU
- Level 2 Link State PDU
- Level 1 Complete Sequence Numbers PDU
- Level 2 Complete Sequence Numbers PDU
- Level 1 Partial Sequence Numbers PDU
- Level 2 Partial Sequence Numbers PDU

PDU Version

1

Reserved

Rezervované pole délky 1 B.

Maximální počet oblastí

Maximální počet povolených oblastí, jichž daný IS může být součástí.

3.2.2 Hello zprávy

Směrovací protokoly používají *Hello* zprávy pro objevování sousedů a pro vyjednání parametrů spojení. V případě IS-IS se jedná o *IS-IS Hello (IIH)* zprávy. IS-IS podporuje několik typů IIH, podle toho o jakou úroveň (L1, L2) a o jakou topologii (point-to-point, broadcast) se jedná. Pro snížení šířky používaného pásma pro point-to-point linku v případě provozování L1 i L2 IIH je definováno, že na takové lince používá jeden typ pro obě úrovně.

Jako všechny IS-IS zprávy i IIH začínají IS-IS hlavičkou. Jaký typ zprávy následuje za touto hlavičkou určuje *PDU type*. *PDU type* pro IIH může být jeden z trojice:

- 15 – L1 LAN IIH,
- 16 – L2 LAN IIH a
- 17 – point-to-point IIH.

PDU type 17 určuje, že se jedná o IIH pro L1, L2, nebo L1L2 úroveň na point-to-point lince.

3.2.3 LAN Hello zprávy

Tento typ zpráv se používá na broadcastových sítích (LAN). Délka *LAN Hello* zprávy je vždy 27 B. Skládá se z běžné hlavičky (8 B) a LAN Hello hlavičky (19 B). *PDU type* je vždy 15, nebo 16, podle toho, zda se jedná o *Hello* zprávu pro L1, nebo L2 IS.

Reserved

Jedná se o prvních 6 b, které jsou rezervované a nepoužívají se. Jejich hodnota by měla být 0.

Circuit type

Poslední dva bity prvního bytu určují jaké úrovně byly nastaveny pro tuto linku. Povolené hodnoty jsou:

- 0x1 pro L1,
- 0x2 pro L2,
- 0x3 pro L1 a L2 současně.

V případě, že se v tomto poli vyskytne jiná hodnota, předpokládá se, že se někde stala chyba a daná Hello zpráva se zahodí.

Source ID

Délka tohoto pole se odvíjí od hodnoty *ID length* z *common header*, nejčastěji tedy 6 B a obsahuje *System ID* odesílatěho IS.

Holding Time

Holding Time říká po jaké době chce být zdrojový IS prohlášen za mrtvého. Jinými slovy, pokud se daný IS neozve do daného počtu vteřin, budeme navázané sousedství považovat za zrušené a provedeme potřebné kroky. Takové kroky obvykle obnášejí rozeslání zprávy o zrušení sousedství okolním IS. Každá přijatá *Hello* zpráva vynuluje čítač odpočtu prohlášení za mrtvého na 0. Narozdíl např. od protokolu OSPF, *Hello time* a *Hold time* intervaly se nemusí shodovat aby došlo k navázání sousedství. Každý IS si může určit svůj vlastní interval a ten případně měnit i v době provozování IS-IS.

PDU Length

PDU Length obsahuje délku celého packetu v B včetně *obecné hlavičky* a *LAN Hello hlavičky*.

Priority

Priority společně s *Designated IS LAN-ID* hrají roli ve volbě designated IS

Designated IS LAN-ID

3.2.4 Point-to-point Hello zprávy

Narozdíl od LAN Hello zprávy, na point-to-point lince nepotřebujeme *Priority* ani *DIS ID*, protože nedochází k volbě DIS. Ostatní pole mají stejný význam jako v LAN Hello.

Local Circuit-ID je "nepodstatná" část protokolu a nemusí být nastavena. Některé implementace ho nastavují na předem danou konstantu. Informační hodnota takového pole je nulová.

3.2.5 LSP

LSP hlavička obsahuje (mimo obecnou hlavičku):

Délka PDU

Určuje délku v bytech celého PDU včetně hlavičky.

Životnost

Doba v sekundách, po kterou je daný LSP platný v rozsahu 2^{16} .

LSP-ID

LSP-ID je složené z položky *System-ID* popsané v části 3.1.2 a dvou bytů reprezentujících *Pseudonode-ID* a *Fragment-ID*. Obě položky si podrobně popíšeme později. Pro jednoduchost si uveďme alespoň, že *Pseudonode-ID* se používá při komunikaci na LAN pro označení všech IS na segmentu. *Fragment-ID* se používá při fragmentaci LSP větších než MTU.

Sekvenční číslo

32-bitová hodnota určující sekvenční číslo podrobněji zpracováno v části 3.4.

Kontrolní součet

Kontrolní součet obsahu LSP od položky *LSP-ID* do konce.

P/ATT/LSPDBOL/IS Type

Jedná se o blok (1 B) sdružených atributů.

První bit P označuje, zda daný IS podporuje funkci Partition Repair. Jedná se o volitelnou funkci protokolu IS-IS, která nemusí být implementována.

Pokud jsou následující 4 b nastaveny, vyjadřují, že daný IS je připojen do okolních sítí, a používá některou z následujících metrik:

- 7. bit Error Metric,
- 6. bit Expense Metric,
- 5. bit Delay Metric,
- 4. bit Default Metric.

LSPDBOL se nastavuje pokud dojde k vyčerpání paměti daného IS. Ostatní IS při výpočtu nejkratší cesty vyřadí takové cesty, které by používaly daný IS jako tranzitní. Výhodou je, že přímo připojené sítě daného IS zůstávají stále dostupné. Poslední dva bity - IS Type - určují topologii, které je daný IS součástí. Každý router je součástí L1 topologie. Pokud je nastaven i druhý bit, je součástí i L2 topologie. Hodnota *IS-Type* pro takový IS by byla 3.

3.2.6 Sequence Numbers PDU

Sequence Numbers PDU, respektive *Complete Sequence Number PDU* a *Partial Sequence Number PDU* slouží pro synchronizaci LSP databáze. CSNP zprávy na Ethernetu jsou zasílány na adresy 0180:c200:0014 (všechny L1 IS) a 0180:c200:0015 (všechny L2 IS) podle odpovídající úrovně. Na point-to-point linkách se CSNP zprávy používají pouze na prvotní synchronizaci při vytvoření sousedství. Následnou synchronizaci zajišťují PSNP zprávy. Oproti broadcastovým rozhraním, kde se PSNP používá pro zaslání chybějícího LSP, na point-to-point lince se využívá jako potvrzení. CSNP hlavička obsahuje:

PDU Length

PDU Length udává délku celé CSNP zprávy včetně obecné hlavičky, CSNP hlavičky a samotných dat v TLV části.

Source-ID

Délka *Source-ID* se odvíjí od délky *System-ID* uvedené v obecné hlavičce, rozšířené o nulový byte reprezentující *Circuit-ID*¹.

¹Je Circuit-ID nekde popsany?

Start LSP-ID a End LSP-ID

Stejně jako velikost *Source-ID*, tak velikost *Start LSP-ID* i *End LSP-ID* je závislá na délce *System-ID* z obecné hlavičky. V tomto případě rozšířené o 2 B – *Circuit-ID* a *Fragment-ID*. Pokud se podaří vměstnat všechny záznamy z link-state databáze (posílají se pouze hlavičky LSP), nastaví se *Start LSP-ID* na hodnotu 0000.0000.0000.00-00 (binárně samé nuly) a *End LSP-ID* na FFFF.FFFF.FFFF.FF-FF (binárně samé jedničky). Tímto způsobem dá odesílatel příjemci najevo, že daný CSNP obsahuje celý rozsah LSP z odesílatelovy databáze. V případě, že se všechny záznamy nevejdou do jednoho CSNP nastavíme položku *Start LSP-ID* prvního CSNP na 0000.0000.0000.00-00 a *End LSP-ID* na hodnotu posledního LSP z TLV části. U dalších CSNP nastavíme *Start LSP-ID* na hodnotu prvního LSP z TLV části a *End LSP-ID* na hodnotu posledního LSP. U posledního CSNP nastavíme *End LSP-ID* opět na hodnotu FFFF.FFFF.FFFF.FF-FF, čímž příjemce identifikuje poslední CSNP. Příjemce také předpokládá, že odesílané LSP záznamy jsou seřazeny podle LSP-ID.

V TLV části CSNP zpráv se mohou objevit pouze dva typy – #9 (LSP Entries) a #10 (Authentication). Autentizaci si popíšeme později s dalšími rozšířeními IS-IS. V CSNP neposíláme celé LSP, ale pouze hlavičku původního LSP. Struktura TLV #9 *LSP Entry* je následující:

TLV Type

Identifikuje typ daného TLV. Pro LSP Entry je to hodnota 9.

TLV Length

Určuje délku obsahu daného TLV. Jedná se o násobky N . Velikost N je závislá na délce používaného *System-ID*. V případě používání 6 B dlouhého *System-ID*, pracujeme s násobky 16.

Remaining lifetime

Zbývající životnost daného LSP záznamu.

LSP-ID

Identifikátor daného LSP záznamu.

Sequence number

Sekvenční číslo daného LSP záznamu.

Checksum

Kontrolní součet původního LSP.

Pomocí těchto čtyř parametrů můžeme jednoznačně identifikovat LSP v link-state databázi. ISO 10589 definuje pro všechny LSP dva příznaky na každou linku pro lepší kontrolu zasílání LSP aktualizací. Jedná se o **SRM** (Send Routine Message) a **SSN** (Send Sequence Numbers). Oba příznaky jsou čistě lokální informací a neposílají se v žádné zprávě ostatním IS. Pokud nastavíme SRM příznak, znamená to, že odpovídající LSP musí být zasláno na

danou linku. Pokud je nastaven SSN příznak, měl by být opodvídající LSP zahrnut v další zasílané PSNP zprávě.

3.3 Link-state databáze

Link-state databáze reprezentuje kompletní topologii dané oblasti. Každý IS v dané oblasti postupně rozesílá své a přeposílá LSP od ostatních IS. Po ustálení tak všechny IS v dané oblasti mají stejnou databázi. K vytvoření kompletní mapy oblasti IS-IS používá několik metod zvaných zaplávání (flooding) a synchronizace (synchronizing). IS-IS si vytváří dvě databáze pro ukládání topologických informací. První pro reprezentaci IS v nejbližším okolí, nazývanou point-of-presence (POP). Jedná se o databázi L1 IS. V druhé databázi je uložena celková topologie reprezentující IS na L2. Díky tomu, že každý IS má kompletní/stejnou databázi může vypočítat topologii sítě nezávisle na ostatních IS. Tomuto principu říkáme *místní výpočet*.

3.3.1 Místní výpočet

V případě distance-vector směrovacího protokolu výpočet nejlepší cesty probíhá distribuovaným způsobem. Pokaždé když RIP router přeposílá informace o dostupné síti, zhorší její dostupnost (metriku) o počet hopů, většinou tedy zvýší hodnotu o 1. Nejenom díky tomu, že tato hodnota nemusí být pouze 1, ostatní routery v síti netuší s jakou metrikou jsou dané sítě dostupné pro určitý router. V link-state protokolu informace o dostupných sítích pouze přeposíláme, ale neupravujeme. Po úvodním rozeslání zpráv (zaplavení) všem IS v oblasti, IS-IS spustí výpočet cest pomocí algoritmu shortest path first zkráceně SPF. Každý IS provádí tento výpočet samostatně.

Víme tedy, že IS-IS používá distribuovanou link-state databázi pro ukládání topologie a místní/lokální výpočet pro určení nejlepší cesty. Jak ale můžeme zajistit, abychom správně reagovali na zpožděné zprávy o stavu linky? , které se někde posílají všechny informace o dostupných sítích bez nutnosti posílání celé lokální link-state databáze a zároveň zaručit aktuálnost těchto informací?

Jak ale zaručíme aktuálnost jednotlivých databází a jakým způsobem vyhodnotíme, která zpráva reprezentuje nejaktuálnější stav určité linky? Abychom zjistili, v jakém pořadí byly jednotlivé zprávy generovány, a tím určili, která je poslední a tedy nejaktuálnější, používáme sekvenční čísla.

Potřebujeme, aby LSP obsahoval informaci, která by vyjadřovala co je aktuální a co zastaralé.

3.4 Číslování LSP

Podobně jako je tomu u TCP i v IS-IS se využívá sekvenčních čísel. Pomocí nich dokážeme vyjádřit o jakou verzi link-state databáze se jedná. IS-IS používá 32 b pole Sequence number hodnotou 1. První odeslaný LSP bude mít tedy hodnotu sekvenčního čísla 0x1.

Pokaždé, když IS oznamuje změnu v topologii svým sousedům, inkrementuje hodnotu sekvenčního čísla v LSP o jedničku. IS, který danou zprávu přijal, zkontroluje, zda od tohoto zdroje nějaký LSP již přijal, pokud ne, nainstaluje daný LSP do místní link-state databáze. V případě, že už od daného zdroje nějaké sekvenční číslo má, musí zkontrolovat, jestliže přijaté sekvenční číslo je vyšší než aktuální hodnota. Pokud je přijatá hodnota vyšší, nahradí, případně vymaže, existující LSP nově přijatým. Pokud je hodnota nižší, přijatý

LSP se jednoduše zahodí. Jelikož je IS-IS *spolehlivý* protokol, potvrzují se i LSP, které zahazujeme.

Maximální hodnota sekvenčního čísla je tedy 2^{32} . V případě zasílání LSP každých 5 s by nám to stále stačilo na zhruba 681 let. Rozsah je tedy poměrně velký a dalo by se předpokládat, že nebude nikdy vyčerpán. Ovšem pro případ, že by k takové situaci přece jen došlo, obsahuje každý LSP položku *Lifetime*, životnost.

3.5 Životnost LSP

Všechny LSP pro určení platnosti obsahují mimo sekvenčního čísla ještě pole životnost. *Životnost* zajišťuje platnost daného LSP pouze po určitou dobu. Pomáhá tak s odstraňováním zastaralých a potencionálně chybných LSP z link-state databáze. V LSP je pro životnost vyhrazeno 16 b celkem 65535 s.

Využíváním životnosti LSP dochází ke stárnutí již nainstalovaných LSP v link-state databázi. Je proto nutné položky pravidelně obnovovat pomocí periodických aktualizací.

3.6 Periodické aktualizace

Periodická aktualizace znamená, že IS musí pravidelně odesílat již jednou odeslané LSP. Interval mezi znovuodesláním LSP musí být samozřejmě menší než je doba životnosti daného LSP. Při každé aktualizaci se inkrementuje hodnota sekvenčního čísla. Doporučený interval periodické aktualizace dle ISO 10589 je 1200 s.

Životnost i interval periodické aktualizace můžeme nastavovat nezávisle na sobě, ale musíme zajistit, aby interval periodické aktualizace byl vždy menší než životnost. Nejlépe o desítky sekund, abychom vykompenzovali případně zpoždění, nebo výpadek.

3.7 Expirace LSP

Pokud není LSP včas obnoven periodickou aktualizací, nebo není vynuceno jeho odstranění pomocí přijetí prázdného LSP, dojde k vypršení doby životnost - expiraci daného LSP. Pokud se hodnota některého LSP dostane na 0 dojde k jeho odstranění z link-state databáze. Aby byla zajištěna konzistence mezi link-state databázemi všech IS, iniciuje IS odeslání LSP, který způsobí odstranění i z ostatních IS viz sekce 3.10. Ačkoliv by měla životnost daného LSP vypršet ve stejnou dobu na všech IS, je kvůli možnosti rozsynchronizování hodin tato vlastnost vynucena explicitně. V normálním případě by k vypršení LSP nemělo nikdy dojít, protože buď jej IS obnoví pomocí pravidelné aktualizace, nebo si původce daného LSP vynutí jeho odstranění. Ovšem ani po vyžádání odstranění, ani po expiraci životnosti není LSP okamžitě odstraněn. Po vypršení doby životnosti LSP je daný LSP udržován v link-state databázi po dobu značenou jako *Zero Age Lifetime*. Po tuto dobu se LSP nachází v databázi, ale není zahrnováno do výpočtu nejkratších cest. Výchozí *Zero Age Lifetime* je 60 s.

3.8 Zaplavování

Zaplavování (z anglického výrazu *flooding*) je mechanismus rozesílání LSP svým sousedům. Pracuje ve dvou verzích podle toho, zda je původcem LSP, nebo jej pouze preposílá. V

případě, že je původcem LSP, odešle jej na všechna rozhraní, na kterých má úspěšně navázaná sousedství. Pokud LSP pouze přeposílá, zkontroluje nejdříve sekvenční číslo pro ověření aktuálnosti. Pokud je hodnota vyšší než aktuálně nainstalovaná v link-state databázi, nainstaluje nový LSP a přepośle jej na všechna rozhraní kromě toho, na kterém daný LSP přijal. Díky kontrole sekvenčního čísla při přeposílání zabráníme jevu *LSP bouře* (*LSP storm*), kdy dochází k nekontrolovanému zvyšování přeposílaných zpráv až dojde k zahlcení sítě. Sekvenční číslo je v tomto případě využíváno obdobně jako TTL v IP.

Ačkoliv kontrolováním sekvenčního čísla při předávání LSP zabráníme nekonečnému putování LSP po síti, na mesh sítích dochází ke zbytečnému přeposílání. Pokud bychom měli full-meshed síť, tak již při *záplavě* LSP od zdroje dojde k informování všech IS. Ti ale nemají tušení o tom, jakou zprávu dostali ostatní sousedé, a proto přepošlou daný LSP všem sousedům, kromě původního zdroje. První IS tedy rozeslal $N - 1$ zpráv, na které $N - 1$ IS odpovědělo $N - 2$ zprávami. Při fully-meshed síti se dostáváme až ke složitosti $O(n^2)$ při rozesílání aktualizací.

Jakým způsobem můžeme zabránit několikanásobnému přeposílání LSP na jeden IS z více sousedů?

3.9 Mesh Groups

Odpovědí na tento problém je RFC 2973 IS-IS Mesh Groups. Jde o způsob prořezávání topologie jako je tomu např. u Spanning Tree. To, které části topologie mají být prořezány, musíme určit ručně. Je nutné si dávat pozor, abychom topologii neprořezali příliš těsně, protože se neumí sama adaptovat při výpadku některé linky. Ačkoliv Mesh Groups je zajímavý způsob jak snížit množství LSP, je to spíše záležitost minulosti při využívání ATM a Frame Relay sítí.

V dnešní době, především kvůli statické konfiguraci a neschopnosti automatické opravy, se příliš nevyužívá. Místo Mesh Groups se na broadcastových sítích využívá *Pseudonode*.

3.10 Odstranění LSP

Pro odstranění LSP bychom se mohli spolehnout na vypršení *životnosti* daného LSP, ale to by při maximální hodnotě mohlo trvat až 18 h. V případě, že chceme bezpečně odebrat IS z dané topologie, tedy ze všech link-state databází, iniciujeme *Network Wide Purge*.

Jedná se o LSP, které má vynulované položky *Remaining lifetime* a kontrolní součet. Sekvenční číslo takového LSP je stejné, nebo větší než poslední odeslané LSP. IS při přijetí takového LSP vymaže všechny LSP od daného IS ze své link-state databáze.

Network Wide Purge se využívá při volbě nového *Designated Intermediate System* (DIS).

3.11 Volba DIS

Na broadcastových LAN sítích má jeden IS speciální funkci. Slouží jako *Designated Intermediate System* (DIS). DIS používá LAN-ID, které je jedinečné v rámci dané LAN, a šíří jej všem ostatním IS na segmentu. LAN-ID je System-ID daného IS rozšířené o 1 B. V případě volby nového DIS potřebujeme odstranit LSP záznamy od původního LSP. Původní DIS tedy vygeneruje a rozešle LSP s nulovými položkami pro životnost a kontrolní součet

a hodnotu sekvenčního čísla zvýší o jedničku. Každý IS, který tento LSP přijme, odstraní uvedené LSP-ID ze svojí link-state databáze.

3.12 Pseudonode a DIS

Na broadcastových sítích, kde každý "vidí" každého, si IS-IS musí vytvářet velké množství záznamů v link-state databázi a dochází k výměně velkého počtu Hello PDU. Abychom snížili tyto "problémy"² zavádí protokol IS-IS pojmy *Pseudonode* a *Designated Intermediate System* (DIS). V následující sekci si popíšeme, v kterých situacích je tento koncept vhodný, jakým způsobem probíhá volba DIS, preempece a další.

3.12.1 Vyvážení počtu sousedství na rozsáhlých LAN

Kdykoli máme velký počet IS na LAN je potřeba vzít v úvahu množství "mluvčích" na segmentu a s tím související počet Hello zpráv. Pokud do stávající rozsáhlé sítě přidáme dalšího mluvčího (IS), dojde ke generování velkého množství zpráv - nový mluvčí (po 3-way handshaku) zašle všem Hello zprávu, ostatní mu na ní odpoví, a vygenerují LSP se změnou v síti. Pokud by všechny IS odpověděly okamžitě, docházelo by nárazově k obrovskému nárůstu zpráv. A v případě, že všichni používají stejný (např. výchozí) Hold time, docházelo by k těmto "bouřím" pravidelně. Tomuto jevu se říká *self-synchronization problem*.

3.12.2 Problém synchronizace

Abychom zabránili generování odpovědí, Hello zpráv a periodických aktualizací ve stejný čas a přitom zajistili dodržení časových limitů specifikuje ISO 10589 povinně *jitter* (odchylku) 25 %. Tato odchylka se generuje náhodně a měla by odpovídat uniformnímu rozložení během celé doby generování. Vygenerovaná hodnota se odečítá od původní hodnoty daného časovače.

3.12.3 Pseudonode

Ve chvíli kdy máme skupinu IS připojených na jednom segmentu LAN dochází k vytváření sousedství každý s každým, což má složitost $O(N^2)$. Roste tedy exponenciálně s počtem IS. Způsob jakým tento problém vyřešili tvůrci IS-IS spočívá v pojmu *Pseudonode*. Jedná se o reprezentaci dané LAN jako dalšího uzlu. Protože *pseudonode* jsou pouze dráty propojující IS bez jakékoliv logiky nutné pro provádění nezbytných úkonů v IS-IS, musí jej zastoupit některý skutečný IS. Takový IS označujeme jako *Designated Intermediate System* (DIS). DIS je jedním z IS na dané LAN a je označen speciálním *System-ID*, díky kterému poznáme, že se jedná právě o *Pseudonode*. Strukturu a způsob vytváření *System-ID* pro DIS si popíšeme později. Použitím *Pseudonode* výrazně redukuje původní složitost každý s každým $O(N^2)$ na hvězdicovou topologii s $O(N)$. Nyní si popíšeme formát a způsob generování *System-ID* pro *Pseudonode*.

3.12.4 Pseudonode-ID

Každý IS má nakonfigurované *System-ID* např. 6 B. V LSP se ale posílá *LSP-ID* s dvěma byty navíc oproti *System-ID*. Poslední byte řeší problém fragmentace. Předposlední byte

²Opravit výraz

označujeme jako *Pseudonode-ID*. Prvních 7 bytů v *LSP-ID* bývá označováno jako *Node-ID* - identita uzlu.

Pokud je hodnota *Pseudonode-ID* nulová, znamená to, že se jedná o skutečný IS, naopak nenulová pak indikuje, že se jedná o *pseudonode*. *Node-ID* je tvořeno *System-ID* DIS pro danou LAN a jedním bytem (*Pseudonode-ID*), který zaručí unikátnost mezi *pseudonode* a skutečným IS, který jedná jeho jménem. Díky osmi bitovému poli *Pseudonode-ID* může daný IS jednat "ve jménu" až 255 pseudonodů. V případě, že se daný IS nechce účastnit volby DIS, nastaví *Pseudonode-ID* na 0.

3.12.5 Modelování link-state databáze

Každé sousedství na LAN je ohodnoceno určitou cenou. Ve chvíli, kdy *DIS* vytvoří pseudonode, musí zajistit, že celková cena skrz LAN nebude zkreslena. Řešením je použití asymetrické ceny pro cestu *k* a *od* pseudonode. Původní cena je nastavena ve směru od reálných IS do pseudonode a pro cestu z pseudonode do IS je nastavena nulová cena. Pro reálné IS je nulová cena neplatná hodnota. Při výpočtu SPF proto musíme pseudonodu věnovat speciální pozornost.

Pseudonody byly navrženy pro snížení zátěže na IS, ale v některých situacích generování a údržba pseudonodu naopak přidává další zatížení.

V návrhu *draft-ietf-isis-igp-p2p-over-lan-03* je popsán způsob jak se vyhnout generování pseudonodu. Návrh spočívá v poslání point-to-point Hello zpráv i na broadcastovém spojení. Pokud se oba (všechny) IS shodnou, nedochází k volbě DIS ani generování pseudonodu.

Ještě předtím než dojde ke generování pseudonode musí být na LAN přítomen DIS.

3.12.6 Volba DIS

Volba DIS je bezestavová a jsou pro ni vyhrazeny dvě pole v LAN IIH hlavičce – *Priority* (priorita) a *Source SNPA* (MAC adresa na Ethernetu). Priorita může být v rozsahu 1 - 127. Hodnota 0 znamená, že daný IS nechce být *DIS*. Pokud je na segmentu více IS se stejnou *Priority*, rozhoduje vyšší *Source SNPA*. Jednotlivé IS provádí výpočet lokálně při přijetí Hello zprávy porovnáním s aktuální *DIS* prioritou. K volbě dochází *pre-emptivně*. Pokud se připojí IS s vyšší prioritou, původní *DIS* rezignuje. Pro zdokumentování³ své akce uvede *Node-ID* nového *DIS* v poli *LAN-ID*. Následně musí odstranit původní pseudonode z link-state databáze. Zašle tedy *Purge LSP*, který obsahuje pouze hlavičku a nulové hodnoty pro *Lifetime* a *Checksum*.

Narozdíl od protokolu OSPF, v IS-IS není záložní *DIS*. Pokud dojde k výpadku spojení s *DIS*, musí proběhnout nová volba. Oproti OSPF má IS-IS výhodu, že může nastavit *Hold time* pro *DIS* na menší hodnotu, čímž zajistí, že výpadek *DIS*, bude detekován v kratším čase oproti ostatním IS. U OSPF bychom si to dovolit nemohli, protože *Holdtime* musí být stejný pro všechny routery na LAN, což by výrazně zvýšilo zátěž sítě.

3.13 Synchronizace databází

Link-state protokoly jsou závislé na faktu, že všechny IS (směrovače) v dané oblasti mají přehled o stejné topologii. Pokud by neměly přehled o stejné topologii mohlo by při směrování docházet ke zbytečně dlouhým cestám, případně vytvoření smyčky. V měnící se síti je

³změnit

potřeba synchronizovat topologii co nejrychleji. Jakým způsobem je toho docíleno v IS-IS si popíšeme v následující sekci.

Synchronizované link-state databáze a výsledné směrovací tabulky jsou zásadní pro směrování paketů do jejich cíle. Pro zajištění synchronizace link-state databází používá IS-IS dva speciální typy zpráv – **CSNP** (**C**omplete **S**equences **N**umber **P**acket) a **PSNP** (**P**artial **S**equences **N**umber **P**acket). Způsob jejich použití je závislý od typu linky, zda se jedná o point-to-point, nebo broadcastová LAN.

DIS získá ze své link-state databáze všechny záznamy a naplní jimi odpovídající počet CSNP zpráv. Následně každý IS na LAN porovná záznamy ze své databáze s těmi přijatými od DIS. Pokud se u přijatých záznamu shoduje sekvenční číslo, všechno je v pořádku. Pokud ne, přijatý LSP záznam může být *starší*, *novější*, nebo *neznámý*.

V případě přijetí staršího LSP je řešení jednoduché. Protože se zdá, že DIS nemá aktuální informace, pošleme poslední verzi daného LSP znovu na LAN.

Pokud je přijatý LSP záznam novější, musí příjemce nastavit SRM příznak pro tento LSP, což způsobí zaslání PSNP zprávy pro DIS. PSNP zprávy mají *PDU Type* 24 pro L1 a 25 pro L2. Požadované LSP jsou do PSNP přidány pomocí TLV #9. DIS po přijetí takové PSNP zprávy jednoduše zašle nejnovější verzi požadovaných LSP na LAN.

Nakonec v případě, že DIS zasílá nový, nebo neznámý LSP záznam, příjemce opět odpovídá PSNP zprávou. Jelikož nemá žádné informace o daném LSP, signalizuje tuto situaci vynulováním položek *Sequence number*, *Lifetime* a *Checksum*. Reakce na takovou PSNP zprávu je stejná jako v případě žádosti o zaslání pouze novější verze. DIS jednoduše zašle nejnovější verzi požadovaných LSP.

Zmiňované PSNP má jednodušší formát než CSNP. V hlavičce PSNP zprávy, kromě obecné hlavičky, je pouze *PDU Length* a *Source-ID*. V TLV části používá stejně jako CSNP TLV #9.

S využitím *DIS* je synchronizace databází na LAN velice jednoduchá a čistá. Kromě dvou příznaků, SRM a SSN nepotřebuje v podstatě žádné jiné stavové informace. Celý proces je tak velice dobře debugovatelný.

3.13.1 Synchronizace databází na point-to-point

Pro synchronizaci databází na point-to-point linkách používáme také CSNP a PSNP, ale mají zde jiný význam než na broadcastových sítích.

V okamžiku navázání sousedství mezi dvěma IS si oba vzájemně zašlou obsah svojí databáze prostřednictvím CSNP. Dojde tak k prvotní synchronizaci databází obou IS. Pokud příjemce (IS-B) zjistí, že odesílatel (IS-A) má novější verzi určitého LSP, nebo dané LSP vůbec nezná, negeneruje žádnou akci. Ve chvíli, kdy se situace obrátí, a IS-B bude odesílatelem CSNP zprávy, IS-A detekuje starší verzi, případně absenci daného LSP, a sám zašle dané LSP. PSNP se pak použije jako potvrzení přijetí daného LSP a zajištění tak *spolehlivé* komunikace. Při odeslání určitého LSP, si odesílatel nastaví SRM příznak, jež signalizuje nutnost zaslání daného LSP, a nezruší ho, dokud mu nepřijde potvrzení ve formě PSNP. IS v pravidelných intervalech kontroluje SRM příznak a rozesílá dosud nepotvrzené LSP.

3.14 Fragmentace

Jak již bylo několikrát zmíněno, IS-IS se od ostatních směrovacích protokolů liší v mnoha ohledech. Narozdíl od protokolu OSPF, který běží na vrstvě IP modelu TCP/IP stacku, případně BGP, který využívá dokonce služeb TCP, se protokol IS-IS nemůže spolehnout

na služby nižších vrstev zajišťujících posílání zpráv větších než MTU dané linky. Protože běží přímo na druhé vrstvě modelu OSI, musí být možnost zasílání zpráv delších než MTU dané linky zahrnuta přímo do protokolu IS-IS.

IS-IS využívá dva ze tří způsobů používaných v *IP – fragmentace na síťové vrstvě a předpokládané minimální MTU*.

Předpokládané minimální MTU vychází z faktu, že ISO 10589 stanovuje minimální MTU, které musí linka splňovat, aby se na ní mohl provozovat IS-IS. V případě, že daná linka nepodporuje MTU minimálně 1492 B, nedojde k navázání sousedství. Kontrola MTU probíhá v *handshake* fázi posíláním uměle nafouknutých IIH zpráv. Nevýhoda tohoto řešení je, že na linkách s větším MTU dochází k plýtvání přenosovým pásmem zvýšenou režii.

Další způsob řešení je použití fragmentace. Protože protokoly nižších vrstev nejsou schopny tuto činnost zajistit, musí ji implementovat přímo IS-IS. IS-IS používá tři základní typy zpráv – Hello (IIH) pro objevování sousedů a výše zmíněnou kontrolu MTU, Sequence number packet (SNP) pro synchronizaci link-state databází a Link-state (LSP).

3.14.1 Hello zprávy

Co se týče IIH zpráv, IS-IS nepodporuje zpracování IIH zpráv rozprostřených přes několik paketů. V současnosti to neznámá žádný problém, protože průměrná velikost IIH zpráv je v rozmezí od 40–70 B a všechny linky používané v IS-IS musí podporovat přenášení minimálně 1492 B. Při odhadovaném nárůstu velikosti hlavičky o 5 B za rok, díky přidávání nových funkcí, máme pořád spoustu místa. IIH obecně *trpí* spíše opačným problémem, kdy se při testování schopnosti linky IIH zprávy uměle zvětšují pomocí TLV částí.

3.14.2 Sequence Number Packets

Sequence Number Packets zahrnují jak CSNP, tak i PSNP. PSNP jsou používány pro potvrzování přijatých LSP, nebo naopak pro vyžádání LSP uvedeného v TLV #9 LSP Entry. V obou případech může PSNP zpráva obsahovat více položek v LSP Entry. Je jedno, jestli odesílatel čeká určitou dobu a sdružuje více položek do jednoho záznamu, nebo je posílá jednotlivě. Ačkoliv záznamy v PSNP bývají seřazené, jejich pořadí nemá žádný vliv na jejich význam. Každý záznam v LSP Entry je zpracováván nezávisle na ostatních, proto není fragmentace PSNP potřeba.

U CSNP zpráv nastává úplně jiná situace ve srovnání s PSNP. Při prvotní synchronizaci na point-to-point linkách, nebo pravidelných aktualizacích na LAN, zasíláme hlavičky všech záznamů z link-state databáze a příjemce musí vědět, které zprávy patří do jedné verze link-state databáze. Při špatné interpretaci by docházelo k opakovanému znovu posílání všech LSP záznamů, kromě těch přijatých v konkrétním CSNP. Proto CSNP hlavička obsahuje *Start LSP-ID* a *End LSP-ID*. Pomocí těchto položek dokáže příjemce identifikovat začátek a konec jednoho setu CSNP zpráv. V menších prostředích, kde se obsah link-state databáze vleze do jedné CSNP zprávy se *Start LSP-ID* nastaví na samé nuly a *End LSP-ID* na samé jedničky (binárně). Příjemce tak pozná, že se jedná zároveň o první i poslední CSNP zprávu. Pro zamezení fragmentování již fragmentovaných dat jsou hlavičky z link-state databáze rozděleny po maximálně 1492 B na jednu CSNP zprávu.

3.14.3 LSP

Pokud zmiňujeme potřebu fragmentace u zasílání pouze hlaviček, v podobě CSNP zpráv, musíme ji uvažovat i při zasílání celých LSP.

V LSP je fragmentace řešena pomocí 1 B pole *Fragment-ID*, které společně s *LAN-ID* (*System-ID* + *Pseudonode-ID*) tvoří dohromady *LSP-ID*. *Fragment-ID* udává o kterou část původního LSP se jedná.

Pokud potřebujeme přenést LSP větší než MTU, rozdělíme jej vždy po celých TLV částech, a očíslovíme od 0. Následující části daného LSP mají položku *Fragment-ID* vždy o jedničku větší.

Přijaté fragmenty původního LSP jsou instalovány do link-state databáze příjemce a rozeslány sousedícím IS. IS-IS není závislý na přijetí všech fragmentů původního LSP v jedné iteraci distribuce LSP záznamů. Pokud některý ze záznamů nedorazí, dojde k jeho vyžádání pomocí synchronizačních mechanismů v podobě CSNP a PSNP zpráv. Předchozí tvrzení má jednu výjimku. Pokud nedorazí část s *Fragment-ID* 0, ostatní fragmenty jsou zahozeny. Fragment 0 je důležitý, protože obsahuje některé informace, které mohou být pouze v nultém fragmentu, a jejich přítomnost určuje výskyt dalších parametrů ve všech následujících fragmentech. Bez fragmentu 0 nemůže být například zahájen výpočet SPF (Shortest Path First) algoritmu.

Stejně jako v dalších částech, tak i zde ISO 10589 striktně nespecifikuje všechny niance rozdělování LSP do jednotlivých fragmentů, ale nechává řešení na konkrétní implementaci. V nepromyšleném návrhu implementace může docházet ke zbytečnému generování a následnému fragmentování LSP. Jde o snahu negenerovat všechny fragmenty při výpadku některého IS a následného posunu všech následujících TLV, čímž dojde ke generování i fragmentů, které by původním výpadkem nebyly ovlivněny. Ačkoliv i změna jediného fragmentu způsobí přepočítání SPF, šetříme prostředky IS nutné pro znovu generování LSP a následné rozesílání všem sousedícím IS.

Celkově můžeme díky osmi bitovému poli *Fragment-ID* rozdělit původní LSP na 256 částí. Každý fragment může pojmout až 1470 B užitečných dat (bez LSP hlavičky), což nám dává celkem 376320 B. Při nasazení pro IPv4 může jediný router rozesílat kolem 42000 prefixů. Pro případ, že se jednoho dne narazí i na toto omezení bylo zavedeno TLV #14 *LSP Buffer Size* pomocí něhož si IS může požádat o zvýšení minimálního MTU z 1492 B.

3.15 Výpočet nejkratších cest

Už víme jakým způsobem navázat sousedství, ověřit linku, udržování spojení/sousedství pomocí IIH, budování link-state databáze pomocí LSP a její udržování pomocí CSNP a PSNP zpráv, ale jakým způsobem dostaneme z link-state databáze data relevantní pro směrování. Pro správné směrování a zajištění nejlepších cest, potřebujeme zajistit bezsmýškovost do všech cílů v síti.

V této části si popíšeme Dijkstrův Shortest Path First algoritmus, určený pro grafy s nezápornými hranami, a jeho nasazení v rámci IS-IS. Zaměříme se na samotný výpočet SPF, route resolution a vkládání prefixů.

3.15.1 SPF

Shortest Path First, nebo také Dijkstrův algoritmus, je algoritmus vymyšlený Holandským vědcem Edsgarem Dijkstrou v roce 1956. Jedná se o algoritmus z oboru teorie grafů na výpočet nejkratších cest z daného uzlu do všech ostatních. Pro zajištění bezsmýškovosti musí být ohodnocení všech hran v grafu nezáporné. Jednotlivé IS reprezentují uzly, linky mezi IS reprezentují hrany a metrika linky odpovídá ohodnocení hrany grafu.

SPF při výpočtu používá tři základní seznamy: UNKNOWN, TENT a PATH. Všechny uzly z link-state databáze jsou nejprve nakopírovány do seznamu UNKNOWN. Všechny dostupné IS aktuálně zpracovávaného uzlu se umístí do seznamu TENT počínaje záznamy uzlu z kterého počítáme cesty (kořen). Poté co SPF nalezne nejlepší cestu do daného uzlu, je tento uzel přemístěn do seznamu PATH. Seznam PATH je na začátku prázdný.

Základní iterace algoritmu spočívá v těchto krocích:

1. Najdi uzel s nejmenší cenou a přemísti jej do seznamu PATH.
2. Najdi všechny dosažitelné uzly z daného uzlu a přesuň uzly ze seznamu UNKNOWN do TENT.
3. Pro každý uzel, který je přesunut do seznamu TENT, udržuj cenu do daného uzlu a *first-hop*.

Při přesunu záznamu uzlu ze seznamu UNKNOWN v kroku 2, musíme vždy provést nejprve kontroly obousměrnosti daného spoje. Pro záznam IS-A -> IS-B, se snažíme najít záznam IS-B -> IS-A. Pokud jej nanejdeme je takový záznam ignorován.

Zvláštní péči při výpočtu SPF musíme věnovat pseudonodu. Pokud máme v TENT seznamu více uzlů se stejnou cenou (nejmenší), upřednostníme při výběru pseudonode.

Zjednodušeně řečeno, do seznamu PATH přidáváme nejkratší cestu z TENT. Do seznamu TENT si dáváme kandidáty na nejkratší cestu, vždy když přidáme záznam do TENT, přidáme z něho dostupné uzly (následníky) do seznamu TENT. Výpočet končí, pokud je seznam TENT prázdný.

Pro každou úroveň (L1 a L2) a každou metriku (default, expense, delay, error) je prováděn výpočet samostatně. Ačkoliv v návrhu IS-IS je počítáno s až čtyřmi metrikami podle nichž je možno směřovat, v praxi se tento model neuplatňuje a používá se pouze jedna. Pro L1L2 IS se všemi metrikami bz bylo nutné provést výpočet SPF celkem osmkrát.

Nesmíme zapomenout na kontrolu overload bitu v nultém LSP daného IS. Pokud je tento příznak nastaven, dané cesty neuvažujeme, protože takový IS může mít nekonzistentní link-state databázi a mohlo by dojít k vytváření smyček při směřování.

Při spuštění výpočtu SPF musíme *zmrazit* aktuální stav link-state databáze.

Výpočet SPF probíhá ve dvou bězích. V prvním běhu se nejprve vytvoří topologická struktura oblasti čistě na základě informací z TLV *IS Reachability*. V druhém průchodu jsou zpracovávány ostatní informace z daného LSP.

Kapitola 4

Implementace protokolu

V této části si uvedeme hlavní principy protokolu IS-IS, tak jak byly implementovány, případně zmíněny odchylky od specifikace dle ISO 10589. Pro podrobný popis zdrojových kódů, prosím, prostudujte příloženou programovou dokumentaci.

Původní návrh implementace počítal pouze s dokončením stávajícího protokolu IS-IS, v kterém bylo implementováno navazování sousedů a výměna LSP na pro L1. Zbývalo už tedy jenom zkopírovat jednotlivé metody pro zprovoznění L2 a implementovat SPF pro nalezení nejkratších cest a následné propagování těchto informací do směrovací tabulky. Bohužel se ukázalo, se bude muset přepracovat většina stávajících metod, protože v částech jako volba DIS a navazování sousedství byly chyby a nakládání s LSP bylo v rozporu s ISO 10589 [?].

Celý protokol je tvořen jednou hlavní třídou `ISIS`. Kromě zpráv, které si vyměňují jednotlivé IS, je využívána třída `ISISTimer` pro hlídání vypršení časových limitů.

Protože v OMNeTu nemůžeme využít multicastových MAC adres `0180:c200:0014` a `0180:c200:0015` jsou všechny zprávy posílány na broadcast adresu `FFFF:FFFF:FFFF`. Proto musíme přijaté zprávy kontrolovat, zda jsou skutečně určené pro daný IS, respektive, že na rozhraní je nastavený odpovídající *Circuit Type*.

4.1 Konfigurace

Chování protokolu je možné ovlivnit pomocí konfigurovatelných proměnných. Jejich hodnota se odvozuje podle toho na jaké úrovni byly zadány. Každá úroveň má jinou prioritu. Nejnížší prioritu mají hodnoty definované v hlavičkovém souboru `ISISTypes.h` odkud jsou hodnoty přebírány, pokud nejsou zadány globálně pro celý IS-IS proces, ani pro jednotlivé rozhraní. Druhou úroveň tvoří "globální" proměnné pro celý proces. Ty, pokud nejsou zadány, jsou přebírány právě z hlavičkového souboru `ISISTypes.h`.

Nejvyšší úroveň tvoří hodnoty zadané pro konkrétní rozhraní. Takové parametry přepíše výše zmíněné. Tento způsob je zvolen pro jednodušší tvorbu konfiguračních souborů a využívá obodný přístup jako je tomu při dědění tříd v objektovém programování. Pokud chceme používat výchozí konfiguraci IS stačí zadat pouze několik povinných parametrů jako *System-ID*, *Area-ID* apod. Pro ostatní parametry se použijí výchozí hodnoty. Případně stačí změnu zadat na úrovni procesu a automaticky se aplikuje do nastavení všech rozhraní. Příklad dvou možných konfigurací IS-IS směrovače, nebo přesněji IS-IS systému můžeme vidět na 4.1 a 4.2. Parametry, které se konfiguruji na rozhraních jsou uvozeny "ISIS", abychom se vyhnuli možné kolizi s ostatními protokoly. Většina parametrů má svůj přímý ekvivalent v Cisco IOS i Juniper JUNOS, s ohledem na možnost jednoduchého importu reálné

konfigurace.

```
<Router id="192.168.3.8">
  <Interfaces>
    <Interface name="eth0">
      <IPAddress>192.168.3.8</IPAddress>
      <Mask>255.255.255.0</Mask>
    </Interface>
  </Interfaces>
  <Routing>
    <ISIS>
      <NET>49.0003.0100.0000.0008.00</NET>
    </ISIS>
  </Routing>
</Router>
```

Obrázek 4.1: Příklad jednoduché konfigurace protokolu IS-IS

4.2 Plánování zpráv

Protože řízení simulace je v OMNeTu prováděno pomocí zasílání zpráv sám sobě, byla nutnost kontrolovat a měnit jednotlivé časy na všech místech implementace jednoduchým způsobem. Proto místo volání funkce `scheduleAt()` z různých míst, došlo k vytvoření centrálního plánování pomocí metody `ISIS::schedule`, která má dva parametry. První je povinný - `ISISTimer *timer` a druhý nepovinný - `double timee` čas. Pro případy, kdy plánujeme zprávu na základě vlastní konfigurace nám stačí pouze objekt `ISISTimer`, pomocí nějž vyhledáme požadované informace například pro dané rozhraní a úroveň. Pokud, ale plánujeme například vypršení platnosti LSP, potřebujeme znát hodnotu z přijatého LSP. V takové situaci využijeme právě druhého volitelného parametru.

Další inovací je zavedení *jitteru* 25 % při plánování většiny zpráv, aby došlo k rozložení nárazového množství zpráv. Tato vlastnost patří mezi vyžadované specifikací a je užitečná pro závěrečné modelování, ale při vývoji poněkud stěžuje odlaďování.

4.3 Hello zprávy

Základ výměny Hello zpráv byl převzat z původní implementace. Navíc byl rozšířen o podporu point-to-point rozhraní. Nejedná se ovšem o opravdové point-to-point spojení, ale pouze simulaci na Ethernetu při přímém propojení dvou IS. Toto zjednodušení nenarušuje principy IS-IS, pouze ulehčuje implementační detaily. Pro oba typy Hello zpráv (LAN a PtP) byl zaveden 3-way handshake pomocí TLV #240. Pro správné přiřazení Hello zprávy k vytvořenému sousedství (dále jako adjacency) je kromě *Source-ID*, z hlavičky konkrétní Hello zprávy, porovnána také odesílatelova MAC adresa a rozhraní, přes které byla zpráva přijata, s odpovídajícími informacemi pro dané adjacency. Díky tomu můžeme úspěšně odlišit několik redundantních spojení mezi dvěma IS. Došlo také na změny pro volbu a rezignaci DISu na LAN. V původní verzi docházelo k nesprávnému vyhodnocení priority

```

<Router id="192.168.3.8">
  <Interfaces>
    <Interface name="eth0">
      <IPAddress>192.168.3.8</IPAddress>
      <Mask>255.255.255.0</Mask>
      <ISIS-Priority>10</ISIS-Priority>
      <ISIS-Network>broadcast</ISIS-Network>
      <ISIS-Priority>10</ISIS-Priority>
      <ISIS-Circuit-type>L1</ISIS-Circuit-type>
      <ISIS-L1-Hello-Interval>10</ISIS-L1-Hello-Interval>
      <ISIS-LSP-Interval>50</ISIS-LSP-Interval>
      <ISIS-L1-CSNP-Interval>10</ISIS-L1-CSNP-Interval>
    </Interface>
  </Interfaces>
  <Routing>
    <ISIS>
      <NET>49.0003.0100.0000.0008.00</NET>
      <IS-Type>level-1-2</IS-Type>
      <LSP-Max-Lifetime>1200</LSP-Max-Lifetime>
      <L1-LSP-Gen-Interval>5</L1-LSP-Gen-Interval>
      <L1-Hello-Multiplier>1</L1-Hello-Multiplier>
    </ISIS>
  </Routing>
</Router>

```

Obrázek 4.2: Příklad rozsáhlejší konfigurace protokolu IS-IS

v případě, že dva (a více) IS měly nastavenou vyšší prioritu než zbývající IS na LAN. K ustanovení DIS nedochází při přijetí každé Hello zprávy, ale pouze, pokud je zjištěna změna při volbě DIS. Samotná volba/kontrola DIS samozřejmě i nadále probíhá při přijetí každé LAN Hello zprávy. Pokud nastane změna DIS, spustí se metoda `ISIS::purgeRemainLSP`, která zajistí *system wide purge* všech LSP předchozího DIS. Další situací, kdy může dojít ke změně DIS, je při vypršení *Hold time* intervalu. Při jeho vypršení již nedochází k resetování DIS na všech rozhraních, ale pouze na rozhraní, které odpovídá dané adjacenci, a to navíc pouze pokud byl daný IS na uvedeném rozhraní veden jako DIS. Pro plánování Hello zpráv přibýly v konfiguraci parametry `ISIS-L1-Hello-Interval` a `ISIS-L1-Hello-Multiplier`. Oba parametry jsou *per-interface* a mají svůj ekvivalent pro *L2*. Můžeme je rovněž konfigurovat i v rámci celého procesu. V tom případě vypouštíme ISIS z názvu parametru. Slouží pro implementaci principu, který můžeme nalézt v Cisco IOS, a dává nám možnost zasílat Hello zprávy v intervalu kratším než 1 s. Výsledný *Hold time* interval, vypočítaný z výše zmíněných parametrů, platí pouze pro non-DIS rozhraní. Na rozhraních, kde daný IS pracuje jako DIS, se používá pouze třetina vypočítané hodnoty. Zajistíme tak rychlejší detekci výpadku DIS.

4.4 LSP

Zpracování LSP bylo předěláno kompletně tak, aby využívalo periodického rozesílání LSP na základě nastavených SRM a SSN příznaků, a aby se rozesílaly pouze změněné LSP. Z toho důvodu musel být pozměněn i formát LSP databáze.

4.4.1 Generování

Pokud při vypršení intervalu pro generování LSP dojde k vygenerování stejného setu LSP, jejich *Sequence number* se nezmění. Ovšem oproti doporučení z ISO 10589 nejsou jednotlivé adjacencies vázány na dané LSP. Stačí tak změna v jednom LSP (přidání/ubrání/změna parametru) a dojde k přepsání všech následujících LSP daného IS. Pojmem *následujících LSP* jsou myšleny všechny LSP s *Fragment-ID* stejným, nebo vyšším než první rozdílné LSP. Takové chování sice není v rozporu se specifikací, nicméně je navrženo vylepšení. Do struktury pro adjacencies se navíc přidá odkaz na LSP a při znovugenerování LSP se bude kontrolovat, zda tyto informace byly naposledy v daném LSP. Dojde tak k zredukování množství zasílaných zpráv při drobné změně konfigurace daného IS. Generování LSP navíc podporuje vytváření fragmentů. Pro možnost simulovat tuto vlastnost i na méně rozsáhlé konfiguraci slouží parametr `ISIS_LSP_MAX_SIZE`, kterým můžeme donutit IS-IS využívat fragmentaci i na menší zprávy. Nově jsou do databáze ukládány celé LSP zprávy tak, jak byly přijaty. To usnadňuje jejich pozdější přeposílání a respektuje filozofii, že pokud IS některému TLV nerozumí, tak jej nezpracovává, ale posílá dál.

4.4.2 Periodické zasílání

Aby nemohlo dojít k "vyhladovění" LSP při čekání na přeposlání je mechanismus SRM a SSN příznaků pro každé LSP navíc doplněn speciální frontou pro každou kombinaci úrovně (L1,L2)/příznaky(SRM, SSN)/typ rozhraní (broadcast, PtP). Zároveň tím respektujeme doporučení ve specifikaci, které umožní efektivnější způsob kontroly nastavení příznaků jednotlivých u jednotlivých LSP. Namísto kontroly všech příznaků pro všechny LSP v databázi stačí zkontrolovat jestli je daná fronta neprázdná. Fronty pro broadcast a PtP jsou odděleny proto, že na rozdíl od PtP, kde se posílají všechny LSP čekající ve frontě, na broadcast rozhraní se náhodně vybírá jedno z aktuálně čekajících LSP. Je to z důvodu ochrany zahlcení DIS, pokud by mu všechny ostatní IS poslali kompletní frontu. Náhodný navíc přispívá k tomu, aby DIS nedostával stejné LSP od všech ostatních IS na segmentu. Aby nedocházelo k zasílání stejného LSP několikrát v průběhu jedné iterace kontroly SRM příznaků, je nastavení SRM příznaku podmíněno jeho aktuálním stavem. Pokud je příznak již nastaven, nemůže být znovu přidán do fronty.

4.4.3 Refresh LSP

Refresh LSP probíhá každých `ISIS_LSP_SEND_INTERVAL` sekund, což je ve výchozím stavu 5 s. Opět je možné tento interval nastavit v konfiguraci pomocí `L1_LSP_Send_Interval`. Refresh interval se nastavuje odděleně pro *L1* a *L2*. Aktualizovány jsou pouze LSP patřící danému LSP a všem jeho DIS inkarnacím. Aby bylo LSP obnoveno, musí být jeho *Remaining lifetime* větší než nula a přidružený časovač - `ISISTimer` - nesmí mít nastaven typ `LSP_DELETE`. V takovém případě se jedná o LSP, pro které již bylo iniciováno odstranění a je uchovávána pouze jeho hlavička po dobu $2 \times \text{ISIS_LSP_MAX_LIFETIME}$. Po uplynutí i tohoto intervalu dojde k úplnému vymazání daného LSP z databáze.

4.4.4 Handle LSP

Při přijetí LSP je kontrolováno, zda existuje adjacency pro odesílající IS. Musí souhlasit nejenom *Source-ID*, ale i rozhraní, přes které byla zpráva přijata. V opačném případě je LSP zahozeno. Celé zpracování LSP při přijetí je v podstatě zjednodušeno pouze na nastavování příslušných SSN a SRM příznaků.

4.4.5 LSP Purge

LSP Purge pracuje ve dvou režimech podle toho, zda se jedná o volání smazání LSP z databáze při vypršení časovače a tedy položky *Remaining lifetime*, nebo IS obrdžel LSP s *Remaining lifetime* rovný nule od svého souseda. Při vypršení časovače LSP, které je uloženo v databázi, nedochází k jeho úplnému vymazání. V databázi se i nadále uchovává hlavička, ale celá TLV část je odstraněna. Typ přidruženého časovače se změní na *LSP_DELETE* a signalizuje, že pokud vyprší, má se dané LSP úplně vymazat. Varianty *purgeRemainLSP* a *purgeMyLSPs* slouží k iniciaci smazání série LSP.

4.5 Sequence Numbers PDU

4.5.1 CSNP

CSNP zprávy jsou plánovány pro každé rozhraní zvlášť dle nastaveného intervalu. Můžeme tedy nastavit zasílání CSNP v jiném intervalu na každém rozhraní.

4.6 SPF

Algoritmus *Shortest Path First* je oproti ISO specifikaci rozšířen o seznam *init*, do které vyextrahujeme všechna spojení z LSP databáze. Po vyextrahování celé LSP databáze dojde k ověření, že jsou všechna spojení obousměrná tak, že v seznamu najdeme jak spojení A->B, tak i B->A. Všechny záznamy, které touto kontrolou neprojdou, jsou ze seznamu vyřazeny. Ve specifikaci je stanoveno, že by tato kontrola měla probíhat při přesunu záznamu položek do seznamu *TENT*. Ale protože po přesunu několika záznamů z *init* bude tento seznam neúplný a ověření by neprobíhalo správně. Další kroky už respektují postup dle specifikace. Aktuálně je implementována podpora pouze pro výpočet nejkratších cest na základě TLV #2 - IS Neighbours. Výpočet je ovšem uzpůsoben pro jednoduché začlenění RFC 1195 a použití TLV #128. Velikost zdroje a cíle je stanovena na 8 B, ačkoliv pro *LAN-ID* stačilo 7 B. Je to z důvodu, že IP adresa s maskou zabírají právě 8 B. Bude tedy stačit pouze naplnit seznam *init* na základě informací v TLV #128 a ostatní části výpočtu mohou zůstat stejné.

Příklad "směrovací tabulky" na základě TLV #2 můžeme vidět na 4.3.

```
Best paths of IS: 49.0002.0100.0000.0004.00 No. of paths: 3
0100.0000.0004.00 metric: 0 via: 0100.0000.0004.00
0100.0000.0006.00 metric: 10 via: 0100.0000.0006.00
0100.0000.0007.00 metric: 20 via: 0100.0000.0006.00
```

Obrázek 4.3: Ukázka nejkratších cest

Protože simulace v OMNeTu je diskrétní, nemusíme se zabývat *zmrazením* LSP databáze. V průběhu výpočtu nejkratších cest nemůže dojít ke změně LSP databáze, protože v danou chvíli běží pouze právě výpočet SPF.

Kapitola 5

Závěr

V této části si shrneme co vše je již implementované,

Je kompletně zprovozněná funkcionality pro vytváření sousedství pro L1 i L2 na obou typech rozhraní. Výměna LSP je zprovozněna a otestována pouze na L1. Při implementaci jednotlivých metod pro práci s LSP bylo počítáno se změnou pouze parametru `circuitType` a tím zprovoznění i L2. Daná funkcionality, ale nebyla zatím dostatečně otestována. Obdobná situace je i v případě SPF algoritmu. Plně funkční je podpora L1 s využitím TLV #2.

Jelikož se mi možnost přidat, nebo odebrat rozhraní v průběhu simulace nepodařilo najít, spoléhá implementace, že se jejich počet ani jejich pořadí nezmění. Stejně jako je tomu u nemodulárních routerů. Pokud bychom chtěli napodobit fungování modulárních routerů, museli bychom zajistit, aby informace o přidání rozhraní byla propagována až do modulu ISIS a muselo by dojít především k ověření délky front SRM a SSN příznaků, protože ty odpovídají počtu rozhraní.

Stávající implementace neumožňuje aktivaci protokolu IS-IS na rozhraní po zahájení simulace. Toto chování je ovšem velice snadno rozšiřitelné a je zvoleno na základě názoru řešitele, nikoliv pro svoji obtížnost, případně jednoduchost implementace. Jde pouze o to, zda při inicializaci naplánovat časovače i na zakázaných rozhraních a při každém zpracování testovat stav rozhraní. V případě, že je rozhraní i nadále neaktivní, naplánovat časovač znovu, nebo v opačném případě vykonat danou funkci. Druhý přístup kontroluje nastavení rozhraní již při úvodní inicializaci a pokud zjistí, že je dané rozhraní neaktivní, časovač není naplánován. Díky menšímu počtu časovačů je možné pouštět simulaci rychleji.

Literatura