ŽILINSKÁ UNIVERZITA V ŽILINE

AUTOREFERÁT
DIZERTAČNEJ PRÁCE

Žilina, máj 2020

Ing. Martin Kontšek

Žilinská univerzita v Žiline
Fakulta riadenia a informatiky

Ing. Martin Kontšek

Autoreferát dizertačnej práce

Riešenie susedských vzťahov integrovaných smerovacích protokolov

na získanie akademického titulu **„philosophiae doctor“ (v skratke PhD.)**
v študijnom programe doktorandského štúdia
aplikovaná informatika

v študijnom odbore:
informatika

Žilina, máj 2020

Dizertačná práca bola vypracovaná v dennej forme doktorandského štúdia **na Katedre informačných sietí, Fakulte riadenia a informatiky Žilinskej univerzity v Žiline**

Predkladateľ: **Ing. Martin Kontšek**
Katedra informačných sietí
Fakulta riadenia a informatiky
Žilinská univerzita v Žiline

Školiteľ: **doc. Ing. Pavel Segeč, PhD.**
Katedra informačných sietí
Fakulta riadenia a informatiky
Žilinská univerzita v Žiline

Oponent: **Ing. Peter Feciľak, PhD.**
Katedra počítačov a informatiky
Fakulta elektrotechniky a informatiky
Technická univerzita v Košiciach

Oponent: **prof. Ing. Pavel Čičák, PhD.**
Ústav počítačového inžinierstva a aplikovanej informatiky
Fakulta informatiky a informačných technológií
Slovenská technická univerzita v Bratislave

Autoreferát bol rozoslaný dňa: .....

Obhajoba dizertačnej práce sa koná dňa **19.8.2020** o **9:00** h. pred komisiou pre obhajobu dizertačnej práce schválenou odborovou komisiou v študijnom odbore **informatika**, v študijnom programe **aplikovaná informatika**, vymenovanou dekanom Fakulty riadenia a informatiky Žilinskej univerzity v Žiline dňa .....

prof. Ing. Karol Matiaško, PhD.
predseda pracovnej skupiny odborovej komisie
študijného programu **aplikovaná informatika**
v študijnom odbore **informatika**

Fakulta riadenia a informatiky
Žilinská univerzita
Univerzitná 8215/1
010 26 Žilina

Obsah

Úvod

Neodmysliteľnou súčasťou nášho každodenného života sú počítače, smartfóny a aj iné zariadenia, pomocou ktorých vykonávame svoju prácu, nakupujeme a komunikujeme s priateľmi, ktorí sa nachádzajú na rôznych miestach planéty. Tieto zariadenia nám taktiež sprostredkúvajú rôzne iné služby, ktoré boli v minulosti nemysliteľné, ako napríklad narábanie s účtom v banke, nákup cestovného lístka na autobus alebo vlak online, alebo aj prístup k množstvu informácií odkiaľkoľvek. Napriek tomu si množstvo ľudí neuvedomuje, že tieto služby by neboli možné bez počítačových sietí, ako sú Internet, ale aj lokálne počítačové siete vo firmách, školách aj našich domácnostiach.

Hlavným prvkom počítačovej siete je smerovač, ktorý prepája viaceré siete a je zodpovedný za prenos paketov medzi nimi na základe cieľovej IP adresy, ktorá môže prislúchať zariadeniu, ako je napríklad webový server v korporátnej sieti, alebo emailový server v zahraničí. Keďže takmer v každej lokalite, okrem domácich sietí alebo malých firiem, sa nachádza viacero smerovačov, je potrebné, aby každý z nich mal informácie o trasách do konkrétnych cieľových sietí. Tieto informácie je možné nakonfigurovať manuálne, čo je vhodné iba pri malom počte smerovačov, alebo je možné použiť jeden zo smerovacích protokolov, pomocou ktorých si smerovače vymenia smerovacie informácie automaticky.
Aj napriek tomu, že smerovacie protokoly sa vyvíjajú už dlhú dobu, ich vývoj nie je ani zďaleka ukončený, pretože prichádzajú nové trendy, ako napríklad SDN (softvérovo definované siete), používajú sa viaceré adresové rodiny súčasne (IPv4 a IPv6) a smerovacie protokoly sa začínajú objavovať aj v iných oblastiach, ako napríklad MANET (mobilná ad-hoc sieť) a VANET (automobilová ad-hoc sieť) siete.

Práve prostredie IP sietí, v ktorých sa používa „*starý*“ IPv4 protokol a zvažuje sa, alebo dokonca už je tiež prevádzkovaný aj „*nový*“ IPv6 protokol, stále čelí rôznym problémom alebo prekážkam. Jednou z problematických oblastí je oblasť nasadenia a prevádzkovania smerovacích protokolov v multiprotokolovom prostredí (IPv4 a IPv6 protokol súčasne). Štúdiom dostupnej literatúry a experimentmi bolo zistené, že každý v súčasnosti dostupný smerovací protokol rieši problém svojej činnosti a budovania IPv4/IPv6 smerovacích informácii iným, svojím spôsobom.

Keďže problematika integrovaných smerovacích protokolov, teda protokolov podporujúcich IPv4 aj IPv6 súčasne, je veľmi široká, počas riešenia dizertačnej práce padlo rozhodnutie na zúženie oblasti skúmania na problematiku vytvárania susedských vzťahov a protokolov využitých na prenos správ smerovacieho protokolu.
Následne sa nám počas analýzy existujúcich integrovaných protokolov podarilo identifikovať štyri kombinácie počtu vytváraných susedstiev a protokolu slúžiaceho na prenos správ samotného smerovacieho protokolu. Tieto kombinácie, ktoré sú v práci nazývané ako triedy riešenia protokolovej integrovanosti, sú spolu so službami nevyhnutnými na vytvorenie susedstiev a prenos informácii popísané v kapitole 2.

Cieľom práce bolo objasniť, ktorá z tried riešenia protokolovej integrovanosti je efektívnejšia a následne navrhnúť všeobecné odporúčania využiteľné pre budúci dizajn a implementáciu smerovacích protokolov, ktorý je vhodný práve pre nasadenia v prostredí sietí využívajúcich viaceré sieťové protokoly a adresové systémy. Aby bolo možné dosiahnuť tento cieľ, vykonali sme analýzu formálnych metód, popísanú v kapitole 3, z ktorej vyplynulo, že jednou z najvhodnejších metód je použitie farbených Petriho sietí.

Naša katedra má so smerovacími protokolmi skúsenosti aj vďaka vývoju otvorenej implementácie smerovacieho protokolu EIGRP. Preto padlo rozhodnutie na začatie modelovania treťou triedou riešenia protokolovej integrovanosti, keďže smerovací protokol EIGRP patrí práve do tejto triedy. Nadobudnuté znalosti z hlbšej analýzy dizajnu protokolu,

sme pretavili práve do konceptuálneho modelu tretej triedy riešenia protokolovej integrovanosti, od ktorej sme odvádzali modely ostatných tried.

Vytvorené modely jednotlivých tried boli do detailov popísané, ich dizajn verifikovaný a následne porovnaný medzi sebou na základe vybraných kritérií. Na základe výsledkov porovnania bolo neskôr možné stanoviť odporúčania pre dizajn budúcich, ale aj úpravu súčasných integrovaných smerovacích protokolov.

1 Smerovanie v sieťach

Ako už bolo spomenuté v úvode, aby mohli zariadenia v našej sieti komunikovať so zariadeniami nachádzajúcimi sa vo vzdialenej sieti, je potrebné, aby bol v našej sieti smerovač. Tento prepája našu sieť s inou sieťou (či sieťami) a má informácie, ako dosiahnuť túto vzdialenú cieľovú sieť (tzv. smerovacie informácie). Smerovač má tieto informácie uložené vo forme položiek v smerovacej tabuľke a aktuálnosť týchto informácií zabezpečuje efektívne smerovanie paketov prechádzajúcich cez smerovač [1], [2].

Keď smerovač prijme IP paket, na základe cieľovej adresy zapísanej v hlavičke paketu prehľadá svoju smerovaciu tabuľku, kde sa pokúsi nájsť najlepšiu cestu do siete, ktorá prislúcha cieľovej adrese. Ak sa položka v smerovacej tabuľke nájde, smerovač zabalí paket do rámca a pošle ho susednému smerovaču cez výstupné rozhranie prislúchajúce nájdenému záznamu. Ak sa prislúchajúci záznam nenájde a k dispozícii je predvolená cesta (angl. *default route*), smerovač pošle paket cez rozhranie prislúchajúce tejto default route. Ak sa ale nenájde ani default route, smerovač paket zahodí [3], [4].

Do smerovacej tabuľky sa automaticky pridávajú záznamy o sieťach priamo pripojených k smerovaču. Ostatné smerovacie záznamy o vzdialených sieťach je potrebné pridať buď manuálnou konfiguráciou, alebo použiť dynamický smerovací protokol. V prípade podpory viacerých sieťových protokolov má smerovač pre každý z nich vytvorenú samostatnú smerovaciu tabuľku [3]–[5].

Účelom dynamických smerovacích protokolov je automatické zisťovanie informácií o dostupnosti a stave vzdialených sietí, výmena informácií so susednými smerovačmi a napĺňanie smerovacích tabuliek. Dôležitou vlastnosťou smerovacích protokolov je čo najefektívnejšia výmena smerovacích informácií medzi susednými smerovačmi, aby sa neobjavovali smerovacie slučky, následkom ktorých sa narúša prenos dát a stabilita siete [1], [2].

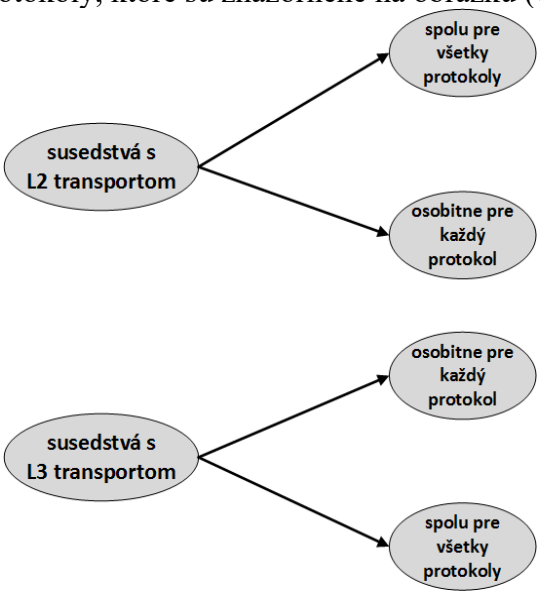
Dynamické smerovacie protokoly sa rozdeľujú podľa rôznych kritérií, ako napríklad podľa oblasti použitia alebo typu vymieňaných smerovacích informácií posielaných medzi susedmi. Ako hlavné delenie sa v literatúre uvádza delenie podľa toho, či sa smerovací protokol používa v rámci autonómneho systému (AS) alebo medzi rôznymi autonómnymi systémami. Pod pojmom autonómny systém je typicky chápaná sieť alebo jej časť, ktorá je pod kontrolou jednej entity. Veľké spoločnosti sú spravidla reprezentované jedným AS a ak máme napríklad pripojenie ku dvom rôznym poskytovateľom sieťovej konektivity (ISP), každý z nich má vlastný autonómny systém. Každý autonómny systém je rozlíšený unikátnym číslom, pričom tieto čísla sú prideľované organizáciou IANA [4].

Sieťový protokol IPv6 prišiel neskôr ako IPv4 a v prípade väčšiny vtedajších smerovacích protokolov ich dizajnéri s podporou viacerých sieťových protokolov nerátali. Preto niektoré protokoly, ako napríklad RIPv2 alebo OSPF sieťový protokol nepodporujú vôbec a bola vytvorená nová verzia, s podporou výlučne IPv6 protokolu. Iné protokoly boli o podporu IPv6 rozšírené, čo je prípad napríklad protokolu IS-IS [4], [5].

2 Aspekty dôležité pre podporu viacerých sieťových protokolov

Na základe dlhoročných skúseností členov Katedry informačných sietí v oblasti sieťových protokolov, ako aj súčasných trendov v oblasti integrovaných smerovacích protokolov sme pristúpili k zúženiu oblasti skúmania. Na základe analýzy dostupnej odbornej literatúry sme sa zamerali na podporu viacerých sieťových protokolov z pohľadu protokolu použitého na transport (prenos) smerovacích informácií a na otázku počtu susedských relácií špecificky pre oblasť, kedy sú v sieti nasadené viaceré sieťové protokoly, keďže táto problematika doteraz nebola riešená. V otázke použitého transportného protokolu nás zaujíma riešenie transportu jednak vzhľadom na vrstvu komunikačného modelu (L2 vs. L3), ako aj na problematiku prenosu samotnej smerovacej informácie. V oblasti susedstva nás podobne zaujímajú otázky riešenia susedských vzťahov opäť jednak na vrstvu komunikačného modelu, ako aj použité L3 protokoly.

Na základe analýzy sa nám podarilo identifikovať štyri možné triedy kombinácií spôsobu riešenia transportu smerovacích informácii a počtu susedských relácií vzhľadom na viaceré použité sieťové protokoly, ktoré sú znázornené na obrázku (Obrázok 1).



Obrázok 1: Triedy kombinácií transportu a počtu susedstiev, zdroj autor

Ako vyplýva z obrázka, možné kombinácie riešenia, nazvime ich ďalej triedy riešenia protokolovej integrovanosti, sú nasledovné:

1. Riešenie transportu smerovacích informácií na linkovej vrstve, riadenie susedstva osobitné pre každý sieťový protokol
2. transport na linkovej vrstve, jedno susedstvo pre všetky použité sieťové protokoly
3. transport na sieťovej vrstve, osobitné susedstvo pre každý sieťový protokol
4. transport na sieťovej vrstve, jedno susedstvo pre všetky použité sieťové protokoly.

Súčasné smerovacie protokoly podporujúce oba sieťové protokoly (IPv4 aj IPv6) môžeme rozdeliť nasledovne:

EIGRP, keďže využíva vlastný transportný protokol RTP a jeho pakety balí priamo do IP protokolu, pričom vytvára samostatné susedstvá pre každý sieťový protokol, zaraďujeme do triedy 3.

Smerovací protokol OSPFv3 vytvára samostatné susedstvá pre každý sieťový protokol a ich správy posiela v oboch prípadoch pomocou IPv6 sieťového protokolu. Preto je rovnako ako EIGRP zaradený do triedy 3.

IS-IS vďaka jeho dizajnu ešte pre OSI siete a jeho protokolovú nezávislosť, zaraďujeme do triedy 2, keďže na prenos správ používa priamo rámce linkovej vrstvy a v prípade prenosu smerovacích informácií IPv4 aj IPv6 protokolu vytvára iba jedno susedstvo.

Pri riešení problematiky „integrovanosti“ a jej vplyvu na dizajn protokolu je potrebné sa zamyslieť nad oblasťami implementácie častí smerovacieho protokolu súvisiacimi s triedami riešenia integrovanosti identifikovanými vyššie. Preto sme následne pre každú z tried vykonali analýzu súvisiacich implementačných oblastí či funkcií (nazývaných ďalej v práci ako služba), na ktoré má priamy vplyv konkrétne riešenie. Ako inšpirácia slúžili funkcie konkrétnych, v súčasnosti dostupných smerovacích protokolov.

3 Formálne popisy smerovacích protokolov

Pri návrhu smerovacích, ale aj iných sieťových protokolov zohráva dôležitú úlohu korektný návrh komplexného systému, akým protokoly bezpochyby sú, a následná verifikácia vytvoreného návrhu, ktorá pomáha odstrániť prípadné zraniteľnosti, nepresnosti a zlepšuje efektivitu navrhovaného protokolu. Pre čitateľa je zaiste zrejmé, že chyby navrhovaného protokolu je ťažšie odstrániť počas implementačnej fázy a neskôr počas testovania výslednej implementácie, ako v počiatočných fázach zameraných na dizajn protokolu. Preto rôzne zdroje odporúčajú už počas návrhovej a verifikačnej časti vývoja protokolu využiť niektoré z nástrojov a metód uľahčujúcich kontrolu dizajnu navrhovaného protokolu.

Jedným zo spôsobov, ktorý napomáha pri korektnom návrhu systémov je použitie formálnych metód (angl. *Formal Description Techniques - FDT*). Základom týchto metód je notácia, ktorá reprezentuje jednoznačnú špecifikáciu navrhovaného protokolu a jeho podporných nástrojov, a za pomoci matematiky umožňuje skúmať vlastnosti návrhu. Tam, kde prirodzený jazyk (angličtina, slovenčina...) môže spôsobiť nejednoznačnosť, formálne metódy ponúkajú prostriedky na predchádzanie takýchto nechcených javov. Samotné použitie formálnych metód nezaručuje korektnosť návrhu, avšak ich správna aplikácia zvyšuje porozumenie požiadavkám na návrh systému [10]–[12].

Farbené Petriho siete (CPN) je modelovací jazyk, ktorý kombinuje užitočné funkcie vysokoúrovňových programovacích jazykov a matematického základu poskytovaného Petriho sieťami. Grafická notácia a prvky potrebné pre modelovanie súbežných procesov, synchronizáciu a komunikáciu sú zabezpečované Petriho sieťami. Prvky vysokoúrovňových programovacích jazykov poskytuje CPN ML čo je funkcionálny jazyk odvodený od všeobecne používaného programovacieho jazyka Standard ML (SML). Medzi prvky poskytované CPN ML jazykom môžeme zaradiť dátové typy, nástroje na vytváranie základných, ako aj parametrických modelov a taktiež prvky nevyhnutné pre popis manipulácie s údajmi. Vďaka funkciám popísaným vyššie, sa farbené Petriho siete zaraďujú medzi siete vyššej úrovne [12], [24], [26]–[30].

4 Ciele práce

Ako sme spomenuli v úvode, v súčasnosti čelia smerovacie protokoly výzvam v podobe objavovania sa nových technológií ako aj ich použitia v nových oblastiach, ktoré sa odlišujú od prostredia klasických počítačových sietí. Nás zaujal hlavne problém využitia smerovacích protokolov v prostredí využívajúcom viaceré sieťové protokoly s vlastným adresovým systémom. Problematika činnosti smerovacích protokolov spolu s riešeniami ich implementácií je veľmi obsiahla. Preto sme sa rozhodli vykonať dekompozíciu funkcionalít smerovacieho

protokolu s bližším zameraním len na parciálne časti priamo súvisiace s oblasťou činnosti v multiprotokolovom nasadení (a nie napr. aj problematiky behu smerovacích algoritmov, ktorá je nezávislá od prevádzkovaného protokolu). Za týmto účelom sa zameriame na analýzu častí/služieb smerovacieho protokolu, ktoré sú potrebné pre transport smerovacích informácií a správu susedstiev v komunikačnom prostredí používajúcom viaceré protokolové adresové rodiny.

Aby bolo možné rigorózne porovnať spôsoby riešenia transportu a susedstiev medzi súčasnými smerovacími protokolmi a navrhnúť zlepšenia, vykonali sme analýzu dostupnej literatúry. Na základe analýzy sa ako vhodný nástroj na vytvorenie modelu činnosti a porovnania jednotlivých služieb javia formálne metódy. Špecificky pre smerovacie protokoly sú vhodným typom formálnych metód farbené Petriho siete.

Na základe našich skúseností s otvorenou implementáciou smerovacieho protokolu EIGRP padla voľba práve na tento protokol ako vzor pre model tretej triedy protokolovej integrovanosti vytvorenej pomocou farbených Petriho sietí.

Pri modelovaní jednotlivých tried protokolovej integrovanosti je potrebné zakomponovať jednotlivé služby smerovacích protokolov nevyhnutné na prenos smerovacích informácií a správu susedstiev v prostredí viacerých adresových rodín. Namodelované triedy za pomoci farbených Petriho sietí je potrebné porovnať za účelom návrhu zlepšenia týchto procesov pre súčasné a budúce integrované smerovacie protokoly. Ako finálny výstup je potrebné formulovať odporúčania na základe výsledkov porovnania pre dizajnérov budúcich smerovacích protokolov.

Jednotlivé čiastkové ciele sú teda nasledovné:

- vytvorenie modelov jednotlivých tried protokolovej integrovanosti
- popísanie jednotlivých modelov
- analýza a porovnanie vytvorených modelov
- vytvorenie odporúčaní pre dizajn integrovaných smerovacích protokolov
- zhodnotenie.

5 Modelovanie tried riešenia protokolovej integrovanosti

Keďže cieľom práce bolo navrhnúť metodiku pre dizajn integrovaných smerovacích protokolov so zameraním na susedstvá a transport údajov smerovacieho protokolu, v kapitole 0 boli popísané štyri identifikované triedy kombinácií počtu susedstiev a spôsobu transportu informácií smerovacieho protokolu.

Aby bolo možné tieto štyri triedy navzájom porovnať a určiť, ktorá z tried je najvhodnejšia pre dizajn budúcich smerovacích protokolov, na základe našej analýzy literatúry, sme sa rozhodli vytvoriť modely každej zo štyroch tried pomocou farbených Petriho sietí za pomoci nástroja CPN Tools.

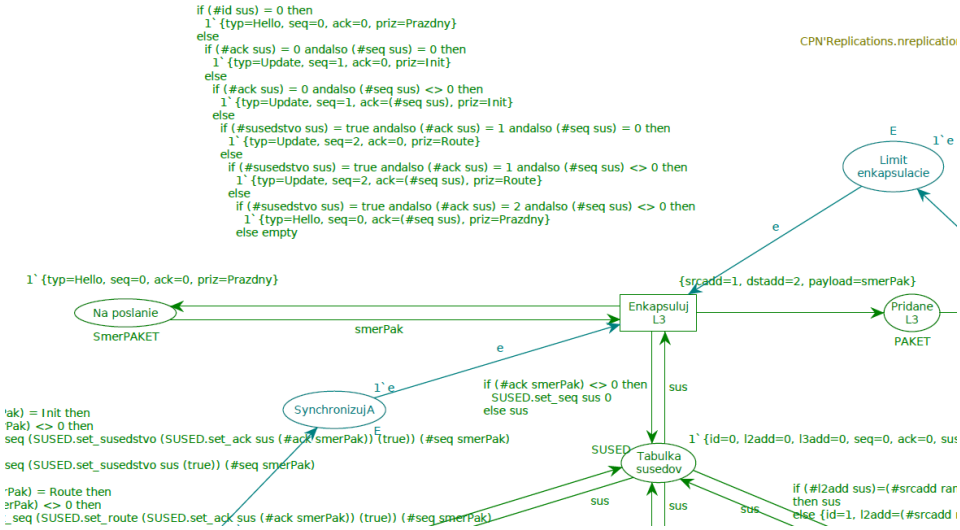
Keďže najviac znalostí z vnútornej štruktúry smerovacieho protokolu sme získali analýzou protokolu EIGRP, rozhodli sme sa ako prvú modelovať tretiu triedu, do ktorej tento protokol zaraďujeme a ako vzor modelu sme použili práve protokol EIGRP. Každý z modelov modeluje komunikáciu a stav práve dvoch smerovačov.

5.1 Tretia trieda – transport L3, viaceré susedstvá

V nasledujúcich podkapitolách vysvetlíme funkcionalitu a dizajn jednotlivých prvkov modelu, pričom časti sú popísané v poradí, v akom bol model vytváraný.

5.1.1 Odosielacia časť

Vytváranie modelu tretej triedy protokolovej integrovanosti sme začali dizajnom častí protokolu nevyhnutných pre vytvorenie susedského vzťahu z pohľadu jedného smerovača (smerovač A). Ako prvé sme na základe procesu vytvorenia susedského vzťahu vytvorili prvok generujúci správy smerovacieho protokolu.



Obrázok 2: Trieda 3 - Generovanie správ, tabuľka susedov

Prvok, ktorý generuje správy smerovacieho protokolu, je na obrázku (Obrázok 2) reprezentovaný miestom s názvom *Na poslanie*. Do tohto miesta môžu byť uložené značky s farbou *SmerPAKET*, čo je údajová štruktúra, definovaná v jazyku CPN ML ako:

```
colset SmerPAKET = record typ:TYP * seq:SEQ * ack:SEQ * priz:PRIZNAK;
```

Táto údajová štruktúra reprezentuje hlavičku smerovacieho protokolu EIGRP, ktorá ale bola zjednodušená a obsahuje iba nasledovné polia:

- typ – identifikuje Hello alebo Update paket
- seq – sekvenčné číslo potvrdzovaného paketu
- ack – potvrdzovacie číslo (sused potvrdzuje prijatie paketu)
- priz – pole príznakov (Prazdne, Init – vytvorenie susedstva, Route – prenos smerovacej informácie)

Miesto *Na poslanie* v úvode obsahuje jedinú správu, Hello paket, ktorý slúži na objavenie suseda a je definovaný nasledovne:

```
1`{typ=Hello, seq=0, ack=0, priz=Prazdny}
```

Ďalšie pakety potrebné na vytvorenie susedstva a prenos smerovacích informácií sú generované na základe informácií z tabuľky susedov, ktorá zachytáva stav, v akom sa susedstvo nachádza. Je reprezentovaná miestom *Tabulka susedov* pričom jednotlivé záznamy sú reprezentované značkou farby SUSED definovanej ako:

```
colset SUSED = record id:INT * l2add:INT * l3add:INT * seq:INT * ack:INT *
susedstvo:BOOL * route:BOOL;
```

Toto miesto, aj napriek názvu nezodpovedá tabuľke susedov v smerovacom protokole EIGRP, ale zachytáva stav, v akom sa susedstvo nachádza, ako aj informáciu o tom, či boli smerovacie záznamy prenesené. Môžeme ho považovať za veľmi zjednodušenú formu kombinácie niektorých funkcionalít v EIGRP zabezpečovaných algoritmom DUAL, tabuľkou susedov a topologickou tabuľkou. Položky, ktoré obsahuje každý záznam tabuľky susedov sú nasledovné:

- id – interný jednoznačný identifikátor suseda
- l2add – linková adresa suseda
- l3add – sieťová adresa suseda
- seq – správu s akým sekvenčným číslom treba susedovi potvrdiť
- ack – správu s akým sekvenčným číslom nám sused naposledy potvrdil
- susedstvo – informácia, či sme prijali správu s príznakom Init, čo značí, že sa začína vytvárať susedstvo
- route – informácia, či sme prijali správu so smerovacími záznamami.

Generovanie ďalších paketov do miesta Na poslanie, prebieha volaním nasledujúcej funkcie vždy, keď sa aktivuje prechod Enkapsuluj L3:

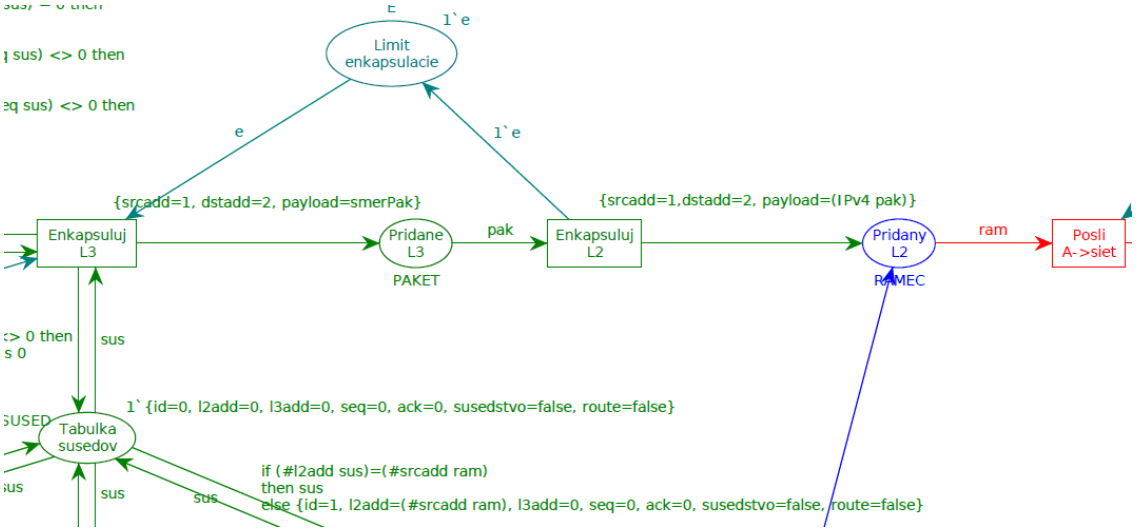
Na začiatku, keď smerovač ešte neprijal žiadnu Hello správu od suseda (id v tabuľke susedov je nastavené na hodnotu 0), sú generované len úvodné Hello správy.

Akonáhle sme suseda objavili (prijali sme od neho Hello správu), id suseda je nastavené na nenulovú hodnotu, funkcia generuje Update správu so sekvenčným číslom 1 a Init príznakom. Potvrdzovacie číslo v tejto správe je nastavené na 0 ak netreba susedovi potvrdiť prijatie predchádzajúcej správy, ak treba, potvrdzovacie číslo je nastavené na sekvenčné číslo z tabuľky susedov.

Potom, ako je susedstvo vytvorené (prijali sme Update správu s Init príznakom a sused potvrdil príjem ekvivalentnej správy od nás), nasleduje posielanie Update správy so sekvenčným číslom 2 a so smerovacou informáciou (vyjadrené príznakom Route). Potvrdzovacie číslo je nastavené rovnako, ako v prípade predchádzajúcej Update správy s Init príznakom (0 ak netreba potvrdiť prijatie správy, hodnota sekvenčného čísla z tabuľky susedov ak treba potvrdiť).

Posledným krokom, ktorý sa vykonáva iba v prípade, že je potrebné ešte potvrdiť prijatie nejakej potvrdzovanej správy od suseda, je poslanie Hello správy s nastaveným potvrdzovacím číslom.

Prechod *Enkapsuluj L3* slúži na enkapsuláciu smerovacej správy do L3 paketu. Ako vstup do prechodu slúži aktuálna správa na poslanie z miesta *Na poslanie* a záznam o susedovi z miesta *Tabulka susedov*.



Obrázok 3: Trieda 3 – Enkapsulácia IPv4 paketu a rámca

IPv4 paket, ktorý je reprezentovaný farbou *PAKET* (miesto *Pridane L3*, znázornené na obrázku Obrázok 3) je definovaný v CPN ML nasledovne:

```
colset PAKET = record srcadd:INT * dstadd:INT * payload:SmerPAKET;
```

kde *srcadd* je zdrojová IPv4 adresa, *dstadd* je cieľová IPv4 adresa a *payload* predstavuje telo paketu, ktoré obsahuje posielanú smerovaciu správu.

Následná enkapsulácia IPv4 paketu do rámca je reprezentovaná prechodom *Enkapsuluj L2*, ktorý má ako vstup IPv4 paket z miesta *Pridane L3* a výstupom je rámec, ktorý je uložený do miesta *Pridany L2*. Samotný rámec je pomocou CPN ML jazyka definovaný nasledovne:

```
colset RAMEC = record srcadd:INT * dstadd:INT * payload:L3PAKET;
```

kde srcaddr je reprezentácia zdrojovej MAC adresy v rámci, dstadd je cieľová MAC adresa a payload reprezentuje telo rámca, zastúpené farbou *L3PAKET*. *L3PAKET* je definovaný nasledovne:

```
colset L3PAKET = union IPv4:PAKET + IPv6:PAKET6;
```

pričom jeho definícia nám umožní uložiť do tela rámca IPv4 ako aj IPv6 paket. Napríklad na hrane vedúcej z prechodu *Enkapsuluj L2* do miesta *Pridany L2* je rámec vytváraný výrazom `{srcadd=1,dstadd=2, payload=(IPv4 pak)}`, čo indikuje prítomnosť IPv4 paketu v rámci. Z definície paketu ako aj rámca je zrejmé, že adresy (IP aj MAC) sú reprezentované farbou INT, čo je celočíselná hodnota. Pre sprehľadnenie simulácie a modelu boli adresy IP aj MAC nastavené nasledovne: ľavý smerovač v modeli hodnota 1, pravý smerovač hodnota 2. Následne je rámec poslaný cez sieť ku druhému smerovaču.

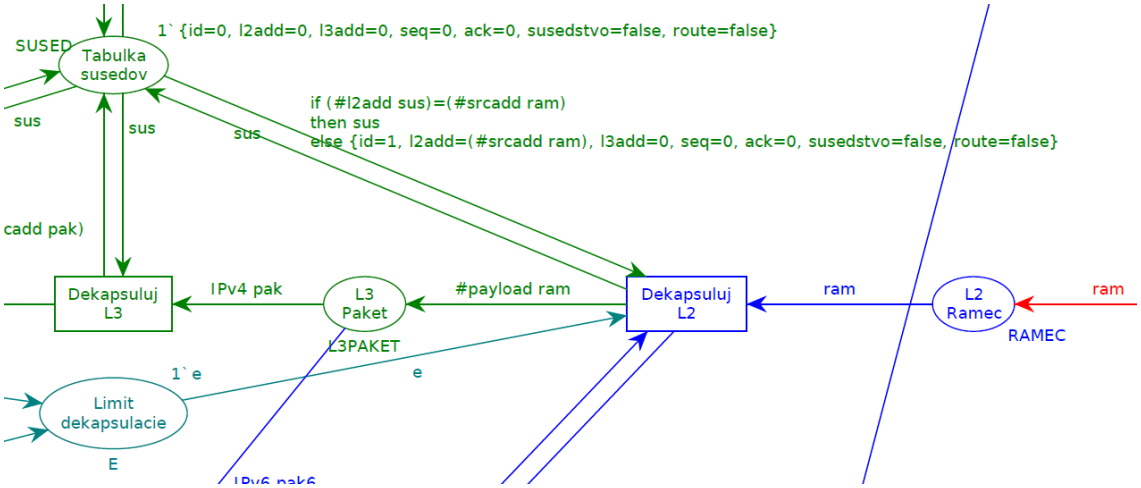
5.1.2 Sieť medzi smerovačmi

Časť modelu, ktorá reprezentuje sieť prenášajúcu rámce medzi smerovačmi pozostáva z nasledovných prvkov. Každý smer je reprezentovaný dvomi prechodmi a jedným miestom. Smer od smerovača A ku smerovaču B je namodelovaný nasledovne: Prechod *Posli A->siet* má ako vstup rámec od smerovača A z miesta *Pridany L2* a výstupom je rámec, ktorý je následne uložený do miesta *Siet A->B*. Potom je rámec zo siete preposlaný smerovaču B do miesta *L2 Ramec B* pomocou prechodu *Posli siet1->B*.

Ekvivalentne je namodelovaný smer od smerovača B ku smerovaču A, ktorý pozostáva z prechodov *Posli siet->B* a *Posli B->siet* a miesta *Siet B->A*.

5.1.3 Prijímacia časť

Prijímacia časť vytvoreného modelu zodpovedá za prijatie rámca, jeho dekapsuláciu a spracovanie.

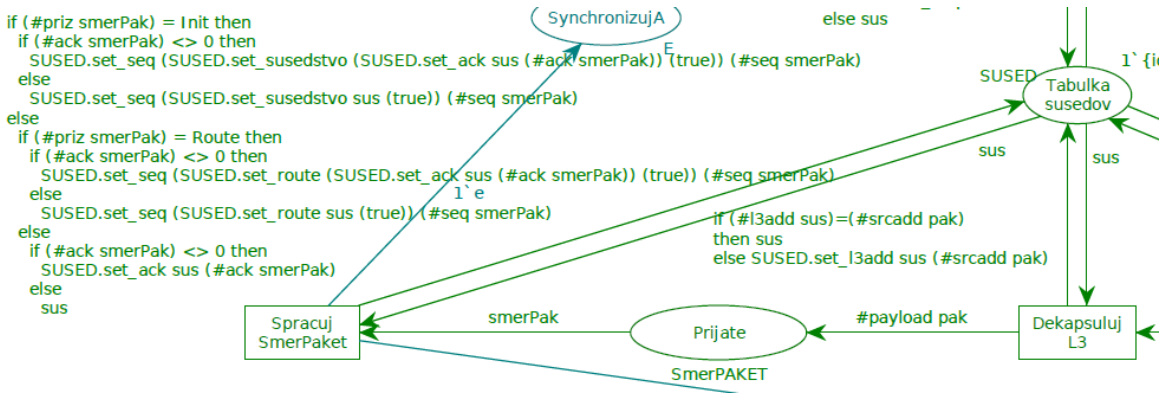


Obrázok 4: Trieda 3 – Prijatie a dekapsulácia rámca

Časť modelu zodpovedná za dekapsuláciu rámca je znázornená na obrázku (Obrázok 4). Najskôr sa prijatý rámec zo siete uloží do miesta *L2 Ramec*. Následne je spracovaný prechodom *Dekapsuluj L2*, ktorý z rámca vyberie IP paket a uloží ho do miesta *L3 Paket*. Ak sa jedná o IPv4 paket, tento je následne spracovávaný prechodom *Dekapsuluj L3*. Pri aktivácii prechodu *Dekapsuluj L2* je upravená aj *Tabulka susedov* nasledujúcou funkciou:

```
if (#l2add sus)=(#srcadd ram)
then sus
else {id=1, l2add=(#srcadd ram), l3add=0, seq=0, ack=0, susedstvo=false, route=false}
```

Úlohou tejto funkcie je, v prípade, že sa v mieste *Tabulka susedov* ešte nenachádza záznam o susedovi, tento záznam vytvoriť s identifikátorom suseda a s MAC adresou prijatou od druhého smerovača.



Obrázok 5: Trieda 3 – Dekapsulácia IPv4 paketu a spracovanie smerovacej správy

Prechod *Dekapsuluj L3* spracúva IPv4 paket tak, že vyberie z neho smerovaciu správu a uloží ju do miesta *Prijate*. Ak záznam o susedovi ešte nemá nastavenú IPv4 adresu, bude po aktivovaní tohto prechodu nastavený na hodnotu zdrojovej adresy z paketu výrazom:

```
if (#l3add sus)=(#srcadd pak)
then sus
```

```
else SUSED.set_l3add sus (#srcadd pak)
```

Posledným prechodom spracúvajúcim prijatú správu smerovacieho protokolu je prechod *Spracuj SmerPaket*. Na základe informácií v prijatej správe nastaví požadované položky záznamu v mieste *Tabulka susedov* nasledujúcou funkciou:

V prvej vetve funkcie rozlišujeme, či bola prijatá správa s Init príznakom, čomu vyhovuje len Update správa posielaná pri vytváraní susedstva. V tomto prípade vykonáme nasledujúce zmeny v zázname tabuľky susedov:

- sekvenčné číslo v tabuľke nastavíme podľa sekvenčného čísla prijatej správy
- položku susedstvo nastavíme na hodnotu true
- potvrdzovacie číslo v tabuľke nastavíme v prípade, že potvrdzovacie číslo v prijatej správe bolo nenulové

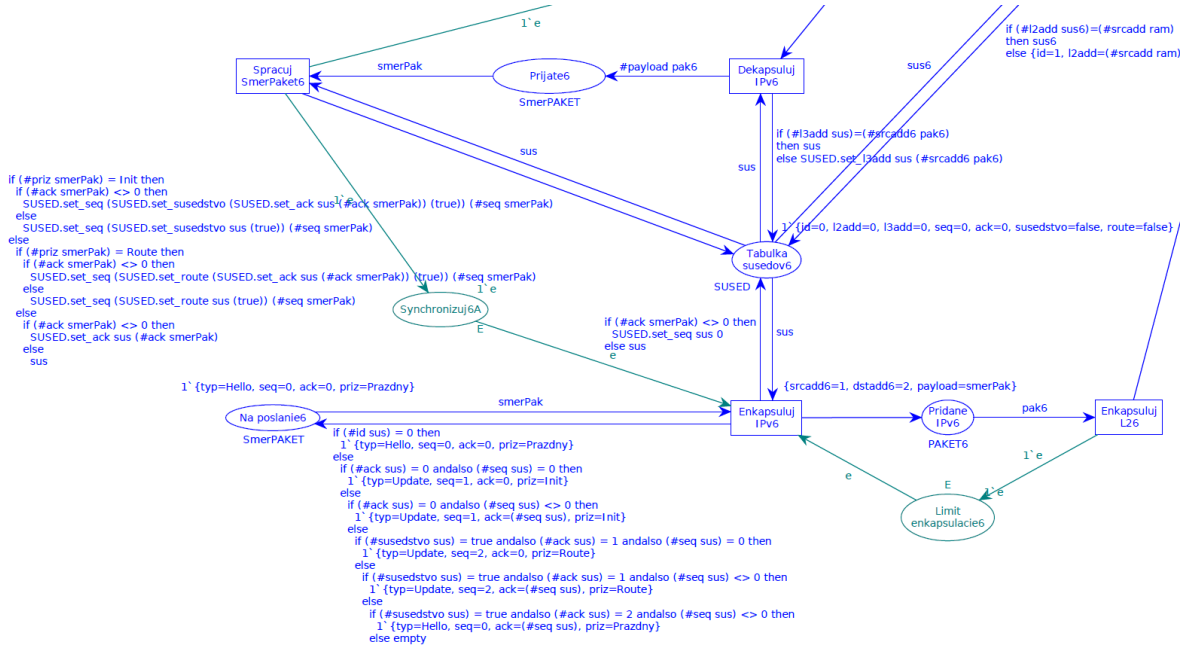
V druhej vetve funkcie zisťujeme, či sa jedná o správu s nastaveným príznakom Route. Tejto podmienke vyhovuje len Update paket, ktorý reprezentuje prijaté smerovacie záznamy od suseda. V tomto prípade vykonáme nasledujúce zmeny v tabuľke susedov:

- sekvenčné číslo v tabuľke nastavíme podľa sekvenčného čísla prijatej správy
- položku route nastavíme na hodnotu true
- potvrdzovacie číslo v tabuľke nastavíme v prípade, že potvrdzovacie číslo v prijatej správe bolo nenulové

Ako posledné rozlišujeme prijatie Hello správy. V prípade, že potvrdzovacie číslo v prijatej správe bolo nenulové, nastavíme potvrdzovacie číslo v tabuľke susedov na hodnotu z prijatej správy. Inak tabuľku susedov nemeníme.
Keďže prijatú smerovaciu správu už ďalej spracovávať nepotrebujeme (všetky potrebné informácie z nej boli uložené v tabuľke susedov), správa v modeli zaniká.

5.1.4 Spracovanie IPv6

V predchádzajúcich podkapitolách boli popísané jednotlivé časti modelu, ktoré reprezentujú funkcionalitu spoločnú pre oba sieťové protokoly (spracovanie rámca a jeho prenos po sieti k ďalšiemu smerovaču) ako aj funkcionalitu špecifickú pre sieťový protokol IPv4. Časť modelu, ktorá je špecifická pre podporu IPv6 sieťového protokolu v smerovacom protokole tretej triedy je znázornená na obrázku (Obrázok 6).



Obrázok 6: Trieda 3 – Podpora IPv6

Ako je z obrázka zrejmé (Obrázok 6), časť siete reprezentujúca podporu IPv6 je veľmi podobná IPv4 časti. Zameriame sa preto hlavne na popis odlišných prvkov a spôsobu napojenia na všeobecnú časť modelu (spracovanie rámcov a poslanie cez sieť). Miesta, ktoré majú rovnakú funkciu ako v prípade IPv4 podmodelu, majú v IPv6 časti za názvom číslicu 6, pretože v jednom modeli nie je možné mať viaceré prvky s rovnakým názvom (model potom nie je možné verifikovať).

Odosielacia časť opäť pozostáva z miesta *Na poslanie6*, prechodu *Enkapsuluj IPv6*, ktorý má funkciu rovnakú ako prechod *Enkapsuluj L3* v prípade IPv4, akurát funkcia na výstupnej hrane je definovaná nasledovne:

```
{srcadd6=1, dstadd6=2, payload=smerPak}
```

Ako je z definície zrejmé, jednotlivé premenné reprezentujúce adresy majú za názvom 6, pretože v tomto prípade funkcia vytvára farbu s názvom *PAKET6* a nasledujúcou definíciou:

```
colset PAKET6 = record srcadd6:INT * dstadd6:INT * payload:SmerPAKET;
```

Na enkapsuláciu paketu sa opäť využívajú informácie z miesta *Tabulka susedov6*, rovnakým spôsobom ako v prípade IPv4 časti. Následne sa IPv6 paket spracúva v prechode *Enkapsuluj L26*, kde je vytvorený rámec, v tomto prípade ale nasledujúcim výrazom:

```
{srcadd=1,dstadd=0, payload=(IPv6 pak6)}
```

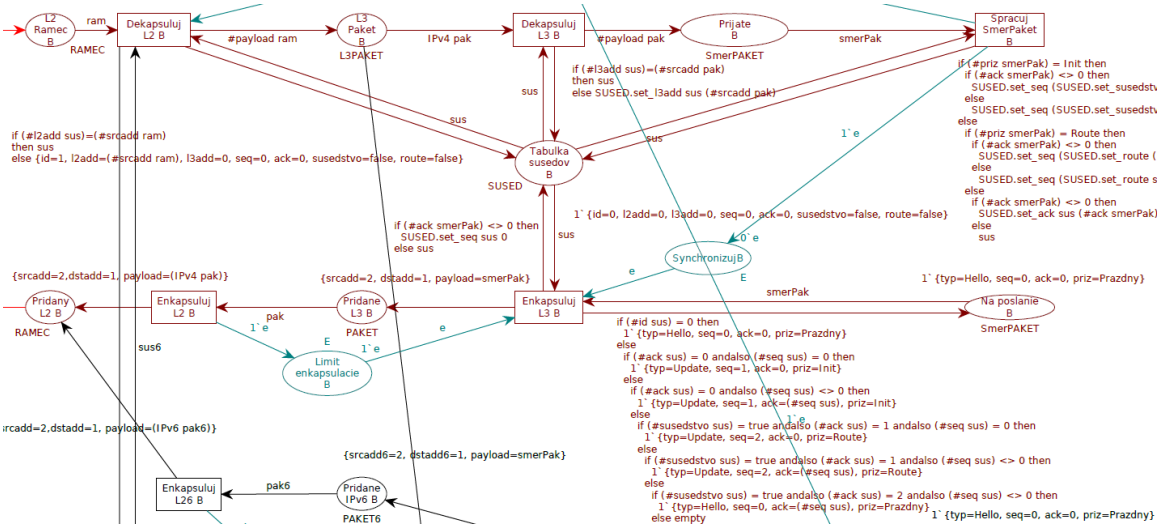
Premenná *payload* v tomto prípade identifikuje v rámci prítomnosť IPv6 paketu. Paket je uložený v mieste *Pridany L2*, rovnako ako v prípade IPv4.

Prijímacia časť IPv6 opäť pozostáva z prechodu *Dekapsuluj IPv6*, kde sa IPv6 paket dostane z miesta *L3 Paket*. Na základe údajov v pakete je podobne ako v prípade IPv4 upravená IPv6 adresa suseda v mieste *Tabulka susedov6*. Z paketu sa vyberie smerovacia správa a uloží sa do miesta *Prijate6*. Výsledná smerovacia správa sa spracúva prechodom Spracuj *SmerPaket6*, úplne rovnako, ako v prípade IPv4 časti modelu.

Vďaka vytvoreniu separátnej časti modelu pre spracovanie IPv6, popísanej v tejto podkapitole, je možná reprezentácia vytvárania dvoch samostatných susedských relácii, jedna pre IPv4 protokol a druhá pre IPv6 protokol.

5.1.5 Reprezentácia smerovača B

Grafické znázornenie spoločných a IPv4 častí smerovača B je zobrazené na obrázku (Obrázok 7).



Obrázok 7: Trieda 3 – Smerovač B spoločná časť a IPv4

Ako si môžeme na obrázku všimnúť, smerovač B je kópiou časti modelu smerovača A, so zmenenými názvami miest a prechodov (pridané písmeno B na konci). Taktiež boli zmenené L2 a L3 adresy v paketoch a rámcoch a tabuľkách susedov, keďže smerovač B má adresu nastavenú na hodnotu 2.

5.2 Štvrtá trieda – transport L3, jedno susedstvo

Model štvrtej triedy riešenia protokolovej integrovanosti sme odvodili od úplného modelu tretej triedy. V prvom kroku sme z modelu odstránili časti špecifické pre IPv6, keďže model štvrtej triedy vytvára iba jedno spoločné susedstvo. Následne sme sa zaoberali modifikáciou odosielacej časti.

5.2.1 Odosielacia časť

Odosielacia časť opäť pozostáva z miest Na poslanie, Tabulka susedov, Pridane L3, Pridany L2 a prechodov Enkapsuluj L3, Enkapsuluj L2, ktoré ale boli modifikované.

Prvým problémom, bol fakt, že v modeli triedy štyri vytvárame iba jedno susedstvo a prenos správ prebieha na sieťovej vrstve, takže smerovacie správy môžu byť enkapsulované do IPv4 aj IPv6 paketu. Smerovače ale musia komunikovať rovnakou verziou IP protokolu, keďže navzájom nie sú „kompatibilné“. Ako najvšeobecnejšie riešenie nám po rôznych pokusoch vyšiel nasledujúci algoritmus:

Prvá smerovacia správa sa pošle dvakrát, pričom jedna bude zabalená do IPv4 paketu a druhá do IPv6 paketu a poslaná cez sieť (predpokladáme, že smerovač má zapnutú podporu oboch sieťových protokolov). Druhý smerovač, na základe toho, ktorý paket prijme skôr, bude s týmto susedom komunikovať iba pomocou sieťového protokolu, ktorého paket prijal ako prvý. Druhý smerovač bude postupovať ekvivalentne. Optimalizáciou tohto algoritmu môže byť poslanie prvého z týchto dvoch paketov IPv6, aby sa minimalizovali situácie, kde každý zo smerovačov zašle ako prvý paket s rôznym sieťovým protokolom. Týmto algoritmom zaistíme, že smerovače vedia spolu komunikovať pomocou spoločného sieťového protokolu a nič

nebráni nadviazaniu susedstva a následnej výmene smerovacích informácii (oboch podporovaných sieťových protokolov).

Pre namodelovanie navrhnutého algoritmu boli potrebné nasledujúce zmeny v odosielacej časti. Najskôr sme do tabuľky susedov pridali položku signalizujúcu, susedstvo s ktorým sieťovým protokolom sme nadviazali (hodnota 4 pre IPv4 a 6 pre IPv6 protokol). Výsledná upravená definícia položky tabuľky susedov je nasledovná:

```
colset SUSED = record id:INT * l2add:INT * l3add:INT * seq:INT * ack:INT *
susedstvo:BOOL * route:BOOL * typ:INT;
```

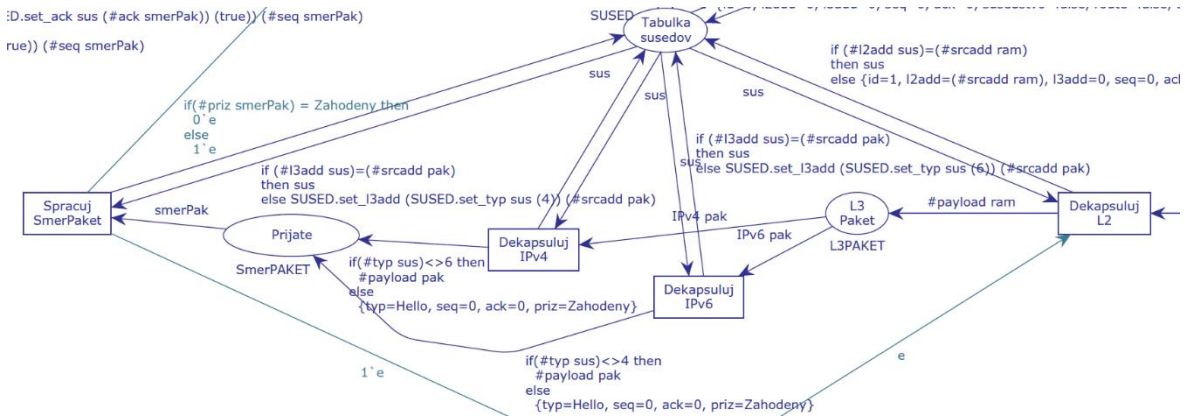
Následne sme upravili funkciu na hrane z prechodu *Enkapsuluj L2* do miesta *Pridany L2* nasledovne:

```
if (#typ sus) = 4 then
  1`{srcadd=1,dstadd=2, payload=IPv4 pak}
else
  if (#typ sus) = 6 then
    1`{srcadd=1,dstadd=2, payload=IPv6 pak}
  else
    1`{srcadd=1,dstadd=2, payload=IPv4 pak}++
    1`{srcadd=1,dstadd=2, payload=IPv6 pak}
```

V prípade, že v tabuľke susedov je typ susedstva nastavený na hodnotu 4 (prijali sme ako prvý paket s IPv4 protokolom) funkcia bude do rámcov baliť len IPv4 pakety. Ak je ale typ nastavený na hodnotu 6, do rámcov pôjdu len IPv6 pakety. Ak je typ nastavený na inú hodnotu ako 4 alebo 6 (ešte sme neprijali paket od susedného smerovača), vytvoria sa dva rámce, jeden s IPv4 a druhý s IPv6 paketom.

5.2.2 Prijímacia časť

Prijímaciu časť modelu štvrtej triedy riešenia protokolovej integrovanosti bolo potrebné modifikovať pre uplatnenie algoritmu popísaného v podkapitole 5.1.1. Modifikovaná časť je zobrazená na obrázku (Obrázok 8).



Obrázok 8: Trieda 4 – Zmeny v prijímacej časti

Prijatý rámec je dekapsulovaný prechodom *Dekapsuluj L2* a následne uložený do miesta *L3 Paket*. Následne sú IPv4 pakety spracované prechodom *Dekapsuluj IPv4*. Ak sa jedná o prvý paket od suseda, do tabuľky susedov sa nastaví typ susedstva na hodnotu 4 a uloží sa IP adresa suseda. Ak sa nejedná o prvý paket od suseda a už bol prijatý IPv6 paket, tento paket sa zahodí a vygeneruje sa prázdny smerovací paket s príznakom *Zahodeny*. Paket sa nakoniec uloží do miesta *Prijate* a neskôr je spracovávaný prechodom *Spracuj SmerPaket*.

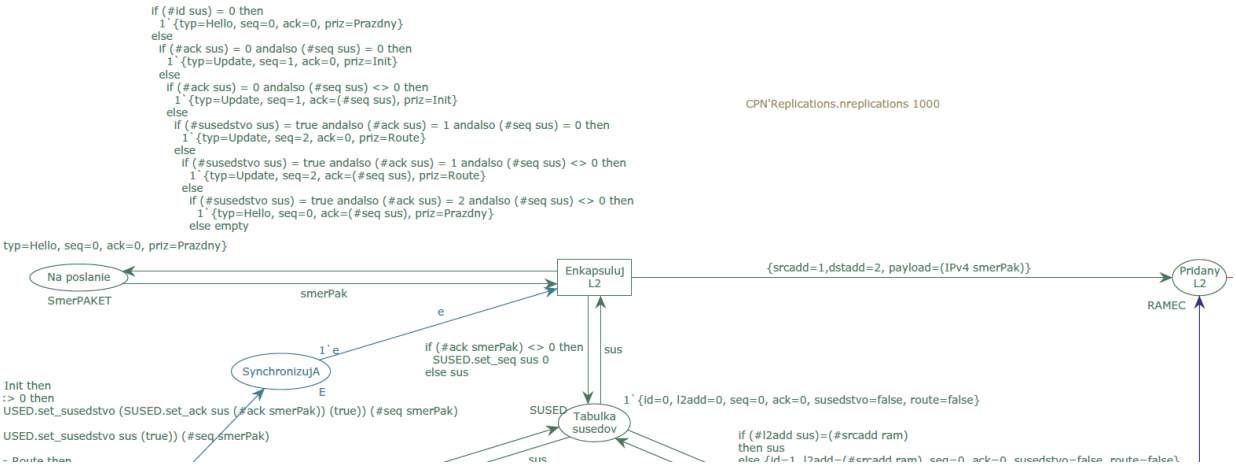
V prípade prijatia IPv6 paketu ako prvého je situácia rovnaká, akurát je využitý prechod *Dekapsuluj IPv6*, ktorý nastavuje hodnoty špecifické pre IPv6 (hodnota typu v tabuľke susedov s číslom 6). Časť modelu reprezentujúca smerovač B bola vytvorená rovnakým spôsobom, ako bolo predstavené v podkapitole 5.1.5. Bolo potrebné vytvoriť kópiu časti smerovača A spolu so zmenami adries.

5.3 Prvá trieda – transport L2, viaceré susedstvá

Model prvej triedy riešenia protokolovej integrovanosti sme rovnako, ako v prípade štvrtej triedy (kapitola 5.2) odvodili od úplného modelu tretej triedy (kapitola 5.1). Keďže prvá trieda prenáša smerovacie správy priamo zabalené v rámcoch, museli sme odstrániť všetky prvky modelu, ktoré vkladali smerovacie správy do paketov a tie následne do rámcov. Najskôr sme sa zaoberali modifikáciou odosielacej časti.

5.3.1 Odosielacia časť

Grafické znázornenie modifikovanej odosielacej časti je zobrazené na obrázku (Obrázok 9).



Obrázok 9: Trieda 1 – Zmeny v odosielacej časti

Odosielacia časť pozostáva z miest *Na poslanie*, *Tabulka susedov*, *Pridany L2* a prechodu *Enkapsuluj L2*. Z položky tabuľky susedov bola odobratá premenná ukladajúca IP adresu suseda. Generovanie smerovacích správ zostalo nezmenené. Najdôležitejšia zmena bola vykonaná prepojením miesta *Na poslanie* priamo s prechodom *Enkapsuluj L2*, pričom výstup z prechodu je rámec, kde *payload* je smerovacia správa pre IPv4 susedstvo vytvorená nasledovnou funkciou:

```
{srcadd=1,dstadd=2, payload=(IPv4 smerPak)}
```

kde výraz *(IPV4 smerPak)* indikuje, že v rámci je nesená práve smerovacia správa IPv4 susedstva (nejedná sa ale o IPv4 paket na sieťovej vrstve).

5.3.2 Prijímacia časť

Prijímacia časť pozostáva z miest *L2 Ramec*, *Prijate*, *Tabulka susedov* a prechodov *Dekapsuluj L2*, *Spracuj SmerPaket*. Z rámca je prechodom *Dekapsuluj L2* vybratá smerovacia správa a uložená do miesta *Prijate*. Súčasne sa aktualizuje tabuľka susedov o informácie z rámca, ak je to potrebné. V mieste Prijate sa môžu uložiť správy pre IPv4 aj IPv6 susedstvo, keďže jeho farba je *SmerPAKET46* s nasledovnou definíciou:

```
colset SmerPAKET46 = union IPv4:SmerPAKET + IPv6:SmerPAKET6;
```

Ak sa jednalo o smerovaciu správu IPv4 susedstva, správa sa následne spracuje prechodom *Spracuj SmerPaket*. Po spracovaní je správa zahodená.

5.3.3 Spracovanie IPv6 susedstva

Keďže prvá trieda riešenia protokolovej integrovanosti vytvára dve osobitné susedstvá pre oba sieťové protokoly, časť modelu popísanú v predchádzajúcich podkapitolách sme zduplikovali a upravili pre spracovanie IPv6 susedstva. Názvy miest a prechodov sme museli upraviť pridaním číslice 6 na koniec názvu. Odosielacia časť po enkapsulovaní rámca prechodom *Enkapsuluj L26* uloží vytvorený rámec do miesta *Pridany L2* rovnako, ako IPv4 časť. IPv6 časť má vlastnú databázu susedov reprezentovanú miestom *Tabulka susedov6*.

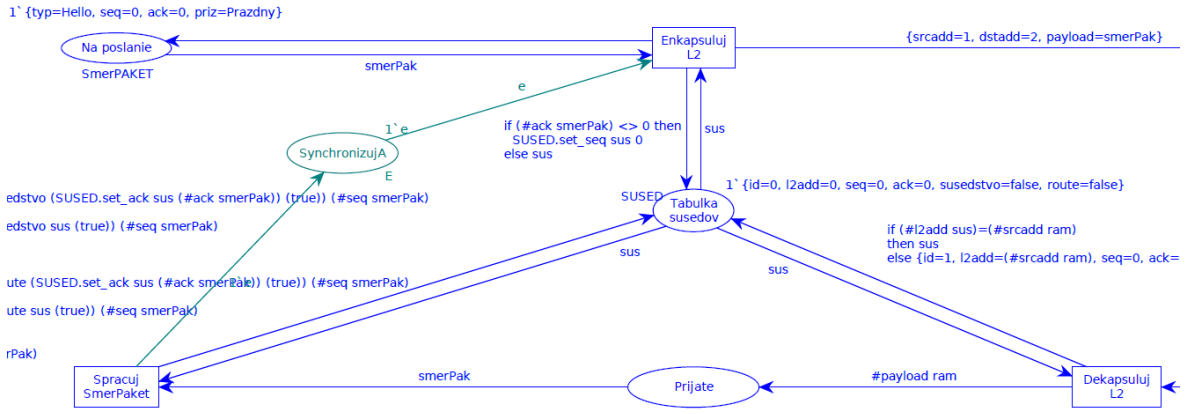
Prijímacia časť zdieľa miesta *L2 Ramec*, *Prijate* a prechod *Dekapsuluj L2* s IPv4 časťou. Samostatné sú len miesto *Tabulka susedov6* a prechod *Spracuj SmerPaket6*, do ktorého sa smerovacia správa IPv6 susedstva dostane z miesta *Prijate*. Definícia smerovacej správy pre IPv6 susedstvo je reprezentovaná farbou *SmerPAKET6* a vyzerá nasledovne:

```
colset SmerPAKET6 = record typ6:TYP * seq6:SEQ * ack6:SEQ * priz6:PRIZNAK;
```

Časť modelu reprezentujúca smerovač B bola vytvorená rovnakým spôsobom, ako bolo predstavené v podkapitole 5.1.5. Bolo potrebné vytvoriť kópiu časti smerovača A, a následne zmeniť adresy. K názvom prechodov a miest bolo pridané písmeno B, keďže v jednom modeli sa nesmú nachádzať prvky s rovnakým názvom.

5.4 Druhá trieda – transport L2, jedno susedstvo

Model druhej triedy riešenia protokolovej integrovanosti sme odvodili od úplného modelu prvej triedy (kapitola 5.3), keďže s ním zdieľa najviac podobností. Celá grafická reprezentácia vytvoreného modelu znázorňujúceho druhú triedu je zobrazená v prílohách D (časť reprezentujúca sieť a smerovač A) a E (časť reprezentujúca sieť a smerovač B). Keďže druhá trieda vytvára iba jedno spoločné susedstvo, odstránili sme všetky prvky zabezpečujúce IPv6 susedstvo z kópie modelu prvej triedy. Časť výsledného modelu reprezentujúca smerovač A je znázornená na obrázku (Obrázok 10).



Obrázok 10: Trieda 2 – Reprezentácia smerovača A

Ako vidíme na obrázku (Obrázok 10), výsledný model smerovača A je najjednoduchší spomedzi modelov všetkých tried riešenia protokolovej integrovanosti. Odosielacia časť pozostáva len z miest *Na poslanie*, *Tabulka susedov*, *Pridany L2* a prechodu *Enkapsuluj L2*. Prijímacia časť obsahuje miesta *L2 Ramec*, *Tabulka susedov*, Prijate a prechody *Dekapsuluj L2* a *Spracuj SmerPaket*.

Reprezentácia smerovača B bola opäť vytvorená ako kópia časti modelu smerovača A, pričom na koniec názvu všetkých prvkov bolo pridané písmeno B a boli zmenené zdrojové a cieľové L2 adresy.

6 Verifikácia vytvorených modelov

Po vytvorení prvého modelu (tretej triedy riešenia protokolovej integrovanosti, popísaného v kapitole 5.1) bolo nutné overiť, či vytvorený model spĺňa funkčné požiadavky, medzi ktoré patrí vytvorenie susedstva medzi smerovačmi, prítomnosť očakávaných hodnôt v tabuľke susedov, generovanie správnych správ na základe hodnôt v tabuľke susedov a iných.

6.1.1 Problém výpočtu stavového priestoru a automatickej simulácie

Ako prvý krok overenia návrhu, na základe vykonanej analýzy farbených Petriho sietí a nástroja CPN Tools, sme sa pokúsili vykonať verifikáciu modelu pomocou vypočítania stavového priestoru. Bohužiaľ, výpočet bol ukončený po 5 minútach (štandardná hodnota ako dlho sa má počítať stavový priestor), pričom nás nástroj informoval, že bol vypočítaný iba čiastočný stavový priestor. Preto sme zvýšili dĺžku výpočtu na 30 minút, ale opäť sme dostali informáciu o iba čiastočnom výpočte stavového priestoru, pričom ale počet uzlov a hrán stavového priestoru násobne narástol. Ani po zvýšení výpočtového času na 1 deň sme nedostali informáciu o plnom stavovom priestore. Keďže sme ani po ďalšom štúdiu literatúry a predlžovaní času výpočtu nedokázali zistiť problém iba čiastočného výpočtu stavového priestoru, pokúsili sme sa odskúšať model iným spôsobom.

Podobným spôsobom, ako sme overovali funkčnosť častí modelu už počas vytvárania, sme sa rozhodli overiť celý vytvorený model. Vykonali sme interaktívnu simuláciu vytvoreného modelu tak, že sme manuálne vyberali z aktívnych prechodov tie, ktoré sa majú vykonať v nasledovnom poradí: od IPv4 časti smerovača A -> enkapsulácia smerovacej správy do paketu -> enkapsulácia paketu do rámca -> poslanie cez sieť -> prijatie rámca susedom B -> dekapsulácia paketu z rámca -> dekapsulácia smerovacej správy -> spracovanie správy -> poslanie úvodnej správy susedom B a tak ďalej až pokým si smerovače nevymenili všetky požadované správy, a teda nenadviazali susedstvo a vymenili si smerovacie informácie. Priebeh bol teda podobný ako bolo znázornené na obrázku reprezentujúcom vytvorenie susedstva smerovacím protokolom EIGRP. Proces dopadol podľa očakávania, preto sme ho opakovali pre IPv6 susedstvo s opätovným úspechom. Interaktívna simulácia IPv4 aj IPv6 si vyžiadala okolo 250 simulačných krokov, čo bolo časovo náročné (bolo treba 250 krát vybrať požadovaný prechod). Keďže interaktívna simulácia dopadla úspešne, rozhodli sme sa pokračovať v preskúmavaní možností simulácie.

Ako nasledujúci krok sme sa pokúsili vykonať automatickú simuláciu nastavením počtu vykonaných krokov na 300, keďže sme očakávali ukončenie simulácie po 250 krokoch ako v prípade interaktívnej simulácie. Avšak aj napriek sľubným výsledkom interaktívnej simulácie, automatická simulácia neskončila ani po nastavení krokov na počet v 10000, podobne ako v prípade výpočtu stavového priestoru.

Tento fakt nás viedol k hlbšiemu preskúmaniu vzorových modelov v literatúre [12], [26] a dokumentácie na oficiálnej stránke nástroja CPN Tools [32], ktorými sme sa zaoberali už počas predchádzajúcej analýzy, keďže vedecké články, ktoré sme využívali počas analýzy nepopisovali do detailov spôsob modelovania, verifikovania a simulácie farbených Petriho sietí. Dodatočné štúdium týchto zdrojov a výpočet stavového priestoru a simulácia vzorových modelov farbených Petriho sietí pre nástroj CPN Tools nás priviedlo k riešeniu popisovaného problému.

6.1.2 Identifikácia problému výpočtu plného stavového priestoru a automatickej simulácie

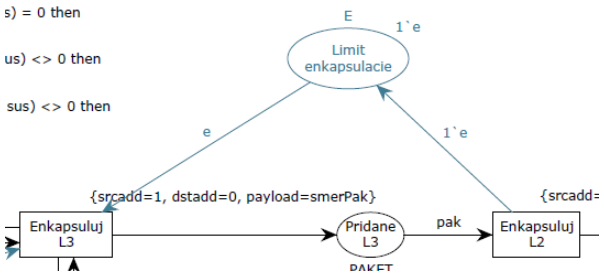
Problém bol spôsobený situáciou, že veľký počet prechodov v modeli bol pri určitých krokoch vykonávania modelu aktívny. Ako príklad môžeme uviesť situáciu, že len v prvom kroku vykonávania navrhnutého modelu tretej triedy riešenia protokolovej integrovanosti sa môže vykonať jeden z nasledujúcich štyroch prechodov: *Enkapsuluj L3, Enkapsuluj IPv6, Enkapsuluj L3 B, Enkapsuluj IPv6 B.* Preto je zrejmé, že už len kombináciou aktívnych prechodov narastá stavový priestor o veľmi veľký počet uzlov a hrán.

Druhý z identifikovaných problémov spôsobených neriadeným počtom aktivovaných prechodov je situácia, keď by sa napríklad vykonávali stále iba prechody posielajúce úvodnú správu IPv4 susedstva od suseda A k smerovaču B. Keďže smerovač B by nikdy neodpovedal smerovaču A, nezačalo by sa vytvárať susedstvo a teda v modeli by nastalo uviaznutie.

Presne tieto popísané problémy spôsobili výpočet iba čiastočného stavového priestoru a „nekončiacu“ automatickú simuláciu. Vytvorený model teda nebol deterministický a bolo potrebné obmedziť počet aktívnych prechodov a synchronizovať oba susedné smerovače.

6.1.3 Riešenie problému výpočtu plného stavového priestoru a automatickej simulácie

Na základe vzorových modelov v literatúre [12], [26] a dokumentácie na oficiálnej stránke nástroja CPN Tools [32], sme vytvorili viaceré limitujúce a synchronizačné miesta, ktoré obmedzujú aktívne prechody v modeli.



Obrázok 11: Limitujúce miesto v modeli

Na obrázku (Obrázok 11) je znázornené limitujúce miesto s názvom *Limit enkapsulacie* (zvýraznené tyrkysovou farbou), ktoré zabezpečuje, že v jednom kroku môže byť prechodmi *Enkapsuluj L3* a *Enkapsuluj L2* spracovávaná iba jedna správa. To znamená, že prechod *Enkapsuluj L3* bude aktívny iba v prípade, že nie je aktívny prechod *Enkapsuluj L2*. Podobným spôsobom môžu byť aplikované limity aj na viaceré prechody (nie len dva). My sme aplikovali limitujúce miesto aj napríklad medzi prechodmi *Dekapsuluj L2* a *Spracuj SmerPaket*, čo znamená, že prechod *Dekapsuluj L2* bude môcť byť opäť aktívny až potom, ako sa vykoná prechod *Dekapsuluj L3* a *Spracuj SmerPaket*.

Synchronizačné miesto s názvom *SynchronizujA*, zvýraznené tyrkysovou farbou je rovnako, ako limitujúce miesta. Toto miesto slúži na odstránenie problému uviaznutia modelu spôsobeného neustálym posielaním správ iba jedným smerovačom v modeli, ktorý ale nikdy nedostane odpoveď. Synchronizačné miesta boli vytvorené jedno pre každé susedstvo v reprezentácii každého zo smerovačov. Podobnú funkciu ako synchronizačné miesta v modeli plnia v reálnych implementáciách smerovacích protokolov časovače.

Po aplikovaní popísaných opráv do všetkých štyroch modelov tried riešenia protokolovej integrovanosti sa nám úspešne podarilo verifikovať všetky vytvorené modely.

6.1.4 Výsledky verifikácie modelov

Výsledkom výpočtu stavového priestoru každého modelu pomocou nástroja CPN Tools je textový súbor s informáciami a štatistikami. Na základe analýzy týchto výstupov sme boli schopní zistiť, či nenastalo uviaznutie v modeli a štatistické informácie ako napríklad, koľko značiek bolo maximálne prítomných v každom mieste.
Najdôležitejšia časť výstupu výpočtu stavového priestoru prvej triedy je nasledovná:

```
Statistics
------------------------------------------------------------------------

  State Space
     Nodes:  21787
     Arcs:   42190
     Secs:   6
     Status: Full
------------------------------------------------------------------------

  Dead Markings [21765,21772,21787]
```

Odstavec State space obsahuje informácie o vypočítanom stavovom priestore. Počet uzlov vyjadruje položka Nodes, položka Arcs počet hrán a pre nás najdôležitejšia položka Status indikuje, či bol vypočítaný plný stavový priestor (teda simulácia bola ukončená po vybudovaní susedstva a výmene smerovacej informácie).
Ako si môžeme všimnúť z výpisu štatistiky, stavový priestor obsahuje desaťtisíce položiek aj napriek domu, že model pozostáva iba z 22 miest a 14 prechodov. Dôvodom je, že v modeli prvej triedy sú vytvárané obe susedstvá súčasne, čo zvyšuje počty možných kombinácií aktívnych prechodov.
Medzi ďalšie zaujímavé položky patria údaje v odstavci Best Integer Bounds, kde pre konkrétne miesta vidíme, koľko značiek maximálne a minimálne sa v nich súčasne vyskytovalo.
Ako poslednú zaujímavú časť výpisu výpočtu stavového priestoru uvedieme položku Dead Markings, ktorá označuje uzly stavového priestoru, v ktorých simulácia končí. V prípade modelu prvej triedy sú uzly 3, čo je opäť spôsobené dvomi susedstvami vytvárajúcimi sa súčasne.

Keďže model druhej triedy vytvára iba jedno susedstvo a je jednoduchší než ostatné modely, jeho stavový priestor je menší, a taktiež obsahuje iba jeden uzol, v ktorom môže simulácia skončiť.

Stavový priestor tretej triedy je, čo sa týka vlastností, podobný ako model prvej triedy, keďže tiež vytvára dve susedstvá. Hlavný rozdiel je iba vo väčšom počte uzlov a hrán stavového priestoru.

Výpočet stavového priestoru štvrtej triedy prebehol rovnako úspešne ako u ostatných modelov. Aj napriek tomu, že vytvára jedno susedstvo, stavový priestor obsahuje viac uzlov a hrán ako v prípade modelu prvej triedy. Táto situácia je spôsobená riešením problému, ktorý sieťový protokol sa použije na výmenu smerovacích správ.

7 Porovnanie jednotlivých modelov

Aby bolo možné vytvoriť metodiku pre dizajn integrovaných smerovacích protokolov, museli sme jednotlivé modely tried riešenia protokolovej integrovanosti rigorózne porovnať. Jednou z najväčších výziev bolo práve výber vhodného kritéria, na základe ktorého bude možné výsledky porovnania využiť na vytvorenie metodiky pre dizajn integrovaných protokolov, ktorá bude nezávislá od konkrétnej budúcej implementácie, preto musí byť vybraté kritérium

dostatočne všeobecné. Preto sme vykonali analýzu dostupnej literatúry, ktorá sa zaoberá vytváraním modelov farbených Petriho sietí [12], [26], [32] ako aj rôznych vedeckých publikácii, zaoberajúcich sa využitím farbených Petriho sietí pre zlepšenie sieťových protokolov [10], [15], [38], [56]–[60], [16], [23]–[28], [34].

Ako prvé kritérium, nad ktorým sme uvažovali na základe jeho použitia pri simulácii rôznych modelov protokolov, je dĺžka nadviazania susedstva spolu s časom potrebným na následnú výmenu smerovacích informácií. Použitie časového kritéria predpokladá, že každý prechod v modeli má pridelenú svoju dĺžku trvania. Problém ale nastáva pri určovaní týchto časových intervalov, keďže dĺžka trvania je implementačne závislým parametrom. Ako príklad uvedieme prechod *Enkapsuluj L3*, ktorý v modeloch tretej a štvrtej triedy slúži na zabalenie smerovacej správy do IP paketu. Táto operácia môže byť v reálnej implementácii smerovača vykonávaná softvérovo, čo môže byť príklad lacných smerovačov, ktoré nie sú optimalizované na prevádzku vo vysokorýchlostných sieťach ale na cenu, pričom pri softvérovej implementácii môže trvať v rádoch mikrosekúnd. Naopak, táto operácia môže byť realizovaná hardvérovým prvkom, čo je prípad súčasných vysokorýchlostných smerovačov, kde táto operácia môže trvať v rádoch nanosekúnd až pikosekúnd. Ako vidíme, kritérium času je veľmi vhodné na porovnanie napríklad dvoch implementácií rovnakého smerovacieho protokolu, ale nie je dostatočne všeobecné pre porovnanie generických tried riešenia protokolovej integrovanosti.

Ako ďalšie možné kritérium porovnania modelov sa intuitívne naskytá použitie počtu miest alebo prechodov. Problémom tohto kritéria, je ale fakt, ktorý sme zistili na základe analýzy literatúry, že niektoré miesta alebo prechody mohli byť v modeli použité z dôvodu obmedzení modelovacieho jazyka. Problém môžeme ilustrovať aj na základe našej skúsenosti z modelovania prijímacej časti modelu štvrtej triedy. Na dekapsulovanie IP paketov bolo potrebné vytvoriť samostatné prechody *Dekapsuluj IPv4* a *Dekapsuluj IPv6*, pretože modelovací jazyk neumožňuje spracovanie dvoch rôznych prvkov štruktúry *union* na jednej hrane. Keďže nie je možné súčasné „*pretypovanie*“ a porovnanie jednotlivých podpoložiek „*pretypovanej*“ farby.

Ako tretie uvažované kritérium bolo porovnanie na základe počtu simulačných krokov potrebných na to, aby sa medzi smerovačmi vytvorilo susedstvo a preniesli sa smerovacie informácie. Jeden simulačný krok predstavuje vykonanie jedného prechodu v modeli. Toto kritérium by sa dalo považovať za zovšeobecnenie kritéria času, keďže nie je závislé od konkrétnej implementácie modelu, ale iba od vykonávania samotného modelu. Keďže modely všetkých štyroch tried boli vytvorené tak, aby odzrkadľovali jednotlivé významné odlišnosti jednotlivých tried a nevyhnutné variácie služieb podporujúcich vytvorenie susedstva a výmenu smerovacej informácie a zvyšné časti modelu boli zhodné, môžeme kritérium počtu simulačných krokov považovať za vhodné a aj dostatočne všeobecné pre porovnanie modelov a následné stanovenie odporúčaní pre dizajn integrovaných smerovacích protokolov.

Keďže vybraté kritérium porovnania (počet simulačných krokov) závisí od spôsobu vykonávania modelu, podmieňujúcim krokom bolo vykonanie verifikácie jednotlivých modelov, ktorá bola popísaná v kapitole . Výpočtom stavového priestoru sme zistili, že vykonávanie niektorých z modelov môže skončiť v rôznych uzloch stavového priestoru, predpokladali sme, že aj počet simulačných krokov sa môže pri rôznych simulačných behoch jedného modelu líšiť. Preto sme sa rozhodli vykonať väčší počet replikácii simulačného behu (v našom prípade 1000 replikácii). Nástroj CPN Tools umožňuje spustiť viaceré replikácie automatickej simulácie pomocou definície:

```
CPN'Replications.nreplications 1000
```

pričom výsledkom je textový súbor s výpisom informácii o každom simulačnom behu.

Dôležité položky výstupu každého simulačného behu sú číslo simulácie (*Simulation no.*), počet krokov, po vykonaní ktorých sa simulácia ukončila (*Steps*) a dôvod ukončenia simulácie (*Stop reason*). Keďže behov simulácie jednotlivých modelov je väčší počet, aby ich

bolo možné súhrnne porovnať, rozhodli sme sa použiť štatistické funkcie aritmetický priemer a modus aby bolo z výslednej štatistiky zrejmé aký bol najčastejší počet krokov simulácie ako aj vplyv všetkých behov na výsledný počet simulačných krokov. Výsledky sú zosumarizované v tabuľke (Tabuľka 1).

	Modus	Aritmetický priemer
Simulácie modelu prvej triedy	180	179,98
Simulácie modelu druhej triedy	90	90
Simulácie modelu tretej triedy	252	252
Simulácie modelu štvrtej triedy	131	131

Tabuľka 1: Výsledky simulačných behov modelov tried riešenia protokolovej integrovanosti

Ako môžeme vidieť z výsledkov v tabuľke, počet krokov všetkých simulačných behov bol zhodný v prípade druhej, tretej a štvrtej triedy, čo indikuje rovnaký medián aj aritmetický priemer. Naopak simulačné behy modelu prvej triedy pozostávali buď zo 180 alebo 175 krokov, pričom 180 krokov sa vyskytovalo najčastejšie.

Z tabuľky taktiež vyplýva, že najlepšou triedou riešenia protokolovej integrovanosti na základe porovnania podľa počtu simulačných krokov je druhá trieda, a teda vytváranie jedného susedstva spolu s posielaním smerovacích správ iba na linkovej vrstve.

8 Odporúčania pre dizajn integrovaných smerovacích protokolov

Hlavným cieľom tejto dizertačnej práce, je vytvorenie odporúčaní pre dizajn budúcich, alebo úpravu existujúcich integrovaných smerovacích protokolov. Preto sme vytvorili modely štyroch identifikovaných tried riešenia protokolovej integrovanosti z pohľadu počtu susedstiev a protokolu využitého na transport smerovacích správ. Na základe verifikácie jednotlivých modelov a následného porovnania pomocou kritéria počtu simulačných krokov sme zistili, že najvhodnejšou triedou riešenia protokolovej integrovanosti je trieda dva.

Preto na základe výskumu prezentovaného v tejto dizertačnej práci je možné sformulovať nasledujúce odporúčania pre dizajn integrovaných smerovacích protokolov z pohľadu počtu susedstiev a transportu:

Smerovacie protokoly, ktoré sú navrhované pre použitie v sieťach, kde sa využívajú viaceré sieťové protokoly, teda integrované smerovacie protokoly, by mali vytvárať iba jedno spoločné susedstvo pre všetky použité sieťové protokoly. Taktiež na prenos správ smerovacieho protokolu by mali využívať len protokoly linkovej vrstvy a teda enkapsulovať smerovacie správy priamo do rámcov linkovej vrstvy. Ako základ smerovacích správ je vhodné využiť TLV štruktúry, ktoré umožňujú veľkú flexibilitu pre budúce rozšírenie samotného protokolu. Potvrdzovanie prijatia je potrebné riešiť v réžii smerovacieho protokolu, keďže linková vrstva takéto služby neposkytuje.

Ako ďalší z prínosov modelu druhej triedy vidíme jeho jednoduchosť čo potvrdil aj výpočet stavového priestoru, keďže stavový priestor bol najmenší z modelov všetkých tried riešenia protokolovej integrovanosti. Keďže je model dostatočne jednoduchý, môže byť po pridaní vhodných rozširujúcich prvkov použitý aj na verifikáciu dizajnu konkrétnej implementácie smerovacieho protokolu založenej na formulovaných odporúčaniach po prečítaní popisu postupu vytvárania ako aj funkcionalít modelov jednotlivých tried v kapitole 5. Relatívne malý stavový priestor vytvoreného modelu môže potenciálne signalizovať výhodu

tohto modelu aj v prípade porovnávania na základe iných kritérií alebo iných častí smerovacieho protokolu navrhnutého na základe týchto odporúčaní. Túto hypotézu by ale bolo potrebné overiť ďalším výskumom.

Výhodou modelu druhej triedy je aj fakt, že nie je potrebné riešiť problém, pomocou ktorého protokolu sa má vytvoriť susedstvo ako v prípade štvrtej triedy, popísanej v kapitole 5.2.1. Aj napriek tomu, že oba modely vytvárajú iba jedno susedstvo, keďže model druhej triedy používa na prenos smerovacích správ protokol linkovej vrstvy, podporované sieťové protokoly je dostatočné indikovať parametrom v hlavičke smerovacej správy.

Keď sa pozrieme na aktuálne využívané integrované smerovacie protokoly, jediným protokolom, ktorého dizajn zapadá do druhej triedy riešenia protokolovej integrovanosti je IS-IS, ktorý ale paradoxne nebol pôvodne vyvinutý pre IP siete. Jeho dizajn, nezávislý od použitých protokolov, mu umožnil byť adaptovaný aj v úplne iných typoch sietí, pre aké bol navrhnutý.

Keďže tretia trieda riešenia protokolovej integrovanosti vyšla z porovnania najhoršie, súčasné integrované smerovacie protokoly, ktoré zaraďujeme na základe ich vlastností do tejto triedy by mohli ťažiť z adaptovania vlastností druhej triedy protokolovej integrovanosti. Preto by cieľom ďalšieho výskumu mohlo byť napríklad prepracovanie protokolov ako sú EIGRP a OSPFv3 do druhej triedy protokolovej integrovanosti a následné porovnanie prepracovaného protokolu s originálnou predlohou.

Záver

Objavovanie sa nových trendov v oblasti počítačových sietí, ako aj používanie smerovacích protokolov v oblastiach iných, než sú klasické počítačové siete, vyvoláva množstvo otázok z hľadiska dizajnu smerovacích protokolov. Náš záujem upútala hlavne otázka podpory viacerých sieťových protokolov v smerovacom protokole a spôsob riešenia transportu na prenos smerovacích informácií. Keďže má naša katedra skúsenosti s vývojom otvorenej implementácie smerovacieho protokolu EIGRP, najväčšia inšpirácia pochádza práve z návrhu tohto protokolu ako aj z iných unikátnych smerovacích protokolov ako IS-IS.

V úvode práce sme načrtli dôležitosť smerovacích protokolov, ich význam a zdôraznili sme fakt, že smerovacie protokoly aj napriek tomu, že sú pre bežného používateľa neviditeľné, zabezpečujú dôležitú úlohu v riadení sieťovej prevádzky v prostredí internetu ako aj vo firemných, operátorských a iných rozľahlých sieťach.

V prvej kapitole sme sa venovali popisu princípu smerovania paketov v počítačových sieťach. Potom sme popísali princíp statických smerovacích záznamov, ich výhody a oblasť použitia. Tieto informácie sme rozšírili o poznatky z činnosti dynamických smerovacích protokolov. Následne sme popísali vlastnosti konkrétnych dynamických smerovacích protokolov s prihliadnutím na ich unikátne vlastnosti a ako bol ich dizajn ovplyvnený v prípade viacerých adresových rodín.

V druhej kapitole sme popísali služby nevyhnutné pre nadviazanie susedstva a posielanie smerovacích informácií vzhľadom na smerovacie protokoly podporujúce viaceré sieťové protokoly, ako aj identifikované triedy riešenia protokolovej integrovanosti.
V tretej kapitole boli popísané nástroje na vytvárane formálnych popisov systémov, ich výhody a nedostatky. Veľké množstvo zdrojov uvádzalo ako vhodný nástroj na modelovanie a následnú verifikáciu a analýzu sieťových protokolov, konkrétne aj smerovacích protokolov, farbené Petriho siete, ku ktorým sme uviedli aj príklad grafického znázornenia jednoduchého modelu.

Celá štvrtá kapitola je venovaná detailnému popisu smerovacieho protokolu EIGRP, jeho vlastnostiam popísaným v RFC špecifikácii ako aj funkciám, ktoré v tejto špecifikácii chýbajú. Následne sú porovnané dostupné otvorené implementácie smerovacieho protokolu

EIGRP voči RFC. V poslednej časti druhej kapitoly sme sa zamerali na testovanie popísaných funkcií otvorených implementácií v spoločnej virtualizovanej topológii. EIGRP bol venovaný v práci väčší priestor pre jeho výber ako základ pre modelovanie tretej triedy protokolovej integrovanosti.

Piata kapitola popisuje ciele dizertačnej práce, prečo sú dôležité a vyčleňuje čiastkové ciele potrebné pre splnenie načrtnutého cieľa celej práce.

Šiesta kapitola sa venuje jednotlivým modelom tried riešenia protokolovej integrovanosti vytvoreným pomocou farbených Petriho sietí a nástroja CPN Tools. Detailne popisuje prvky, z ktorých sa modely skladajú, načrtáva ich funkčnosť a popisuje problémy, ktoré bolo potrebné vyriešiť pre dosiahnutie požadovanej funkčnosti jednotlivých modelov tried.

V siedmej kapitole sme načrtli problémy verifikácie vytvorených modelov, postup identifikácie zdroja problémov a ich riešenie. V neposlednom rade siedma kapitola obsahuje výsledky verifikácie jednotlivých modelov.

Ôsma kapitola pojednáva o rôznych kritériách, ktoré je možné použiť na porovnanie viacerých modelov a vyberá najvhodnejšie z nich, ktoré boli zvolené na vytvorenie odporúčaní. V poslednej deviatej kapitole je predložená metodika pre dizajn integrovaných smerovacích protokolov na základe výsledkov porovnania jednotlivých modelov tried riešenia protokolovej integrovanosti a sú v nej popísané ďalšie výhody najlepšieho z modelov.

Ako ďalšie možnosti rozšírenia a budúceho výskumu odvíjajúceho sa od výsledkov tejto práce vidíme možnosť aplikácie odporúčaní pre dizajn integrovaných smerovacích protokolov do zmeny dizajnu súčasných smerovacích protokolov patriacich do tretej triedy protokolovej integrovanosti ako sú EIGRP a OSPFv3. Po zmene ich dizajnu a následnej implementácii by bolo možné porovnať prínosy odporúčaní stanovených na základe výskumu v tejto dizertačnej práci na reálnej implementácii smerovacieho protokolu. Taktiež je možné využiť postupy a princípy načrtnuté a preverené v tejto práci na vylepšenie iných častí integrovaných smerovacích protokolov.

Zoznam použitej literatúry

[1] Cisco Networking Academy, *Routing protocols companion guide.* Cisco Press, 2014.

[2] R. Graziani and A. Johnson, *Routing Protocols and concepts, CCNA exploration companion guide*. Cisco Press, 2007.

[3] N. Kocharians and P. Palúch, *CCIE Routing and Switching v5.0 Official Cert Guide, Volume 1*, 5th ed. Indianapolis: Cisco Press, 2014.

[4] K. Wallace, *CCNP Routing and Switching ROUTE 300-101 Official Cert Guide*. Indianapolis: Cisco Press, 2015.

[5] N. Kocharians, *CCIE Routing and Switching v5.1 Foundations: Bridging the Gap Between CCNP and CCIE*, 1st ed. Cisco Press, 2017.

[6] "CIDR Report." [Online]. Available: https://www.cidr-report.org/as2.0/. [Accessed: 01-Aug-2018].

[7] N. Kocharians and T. Vinson, *CCIE Routing and Switching v5.0 Official Cert Guide , Volume 2*. 2015.

[8] A. Lindem, S. Mirtorabi, A. Roy, M. Barnes, and R. Aggarwal, "Support of Address Families in OSPFv3," 2010.

[9] R. Callon, "Use of OSI IS-IS for routing in TCP/IP and dual environments," 1990.

[10] S. Aly, K. M.-D. U. July, and U. 2003, "Protocol verification and analysis using Colored Petri Nets," *Citeseer*, 2003.

[11] S. Šimoňák, "Modelovanie a analýza systémov s využitím integrácie formálnych metód [Habilitačná práca]," Technická univerzita v Košiciach, 2018.

[12] K. Jensen and L. M. Kristensen, *Coloured Petri Nets: Modelling and Validation of*

Concurrent Systems. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009.
[13] G. J. Holzmann and American Telephone and Telegraph Company., *Design and validation of computer protocols*. Prentice Hall, 1991.
[14] "Carl Adam Petri • IEEE Computer Society." [Online]. Available: https://www.computer.org/web/awards/pioneer-carl-petri. [Accessed: 01-Aug-2018].
[15] G. Hareesh and S. Chinara, "Dynamic Modeling of Routing Protocol Using Colored Petri Nets," p. 47, 2015.
[16] D. Li, Y. Cui, K. Xu, and J. Wu, "Improvement of Multicast Routing Protocol Using Petri Nets," pp. 634–643, 2005, doi: 10.1007/11548706_67.
[17] D. A. Zaitsev, "Verification of Protocol BGP via Decomposition of Petri Net Model into Functional Subnets," *Proc. Des. Anal. Simul. Distrib. Syst. Symp.*, pp. 5–7, 2005.
[18] H. Chen, C. Zhou, Y. Qin, A. Vandenberg, A. V. Vasilakos, and N. Xiong, "Petri Net Modeling of the Reconfigurable Protocol Stack for Cloud Computing Control Systems," in *2010 IEEE Second International Conference on Cloud Computing Technology and Science*, 2010, pp. 393–400, doi: 10.1109/CloudCom.2010.81.
[19] D. Losch and J. RoBmann, "Visual Programming and Development of Manufacturing Processes Based on Hierarchical Petri Nets," in *2016 3rd International Conference on Soft Computing & Machine Intelligence (ISCMI)*, 2016, pp. 154–158, doi: 10.1109/ISCMI.2016.12.
[20] M. Naedele and J. W. Janneck, "Design patterns in Petri net system modeling," in *Proceedings. Fourth IEEE International Conference on Engineering of Complex Computer Systems (Cat. No.98EX193)*, 1998, pp. 47–54, doi: 10.1109/ICECCS.1998.706655.
[21] C. Seatzu, "Modeling, analysis, and control of automated manufacturing systems using Petri nets," in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2019, pp. 27–30, doi: 10.1109/ETFA.2019.8869012.
[22] Zhicao Dong and Xinhui Zhao, "Application of Petri Nets in Material Supply Systems," in *2010 Fourth International Conference on Genetic and Evolutionary Computing*, 2010, pp. 63–66, doi: 10.1109/ICGEC.2010.24.
[23] S. Niari, A. J.-P. of the 6th I. Conference, and U. 2013, "Verification of OSPF vulnerabilities by colored Petri net," *Proc. 6th Int. Conf. Secur. Inf. Networks*, pp. 102–109, 2013.
[24] D. Ibrahim, E. Sallam, … T. E.-… C. on C., and U. 2014, "Coloured petri net model for vector-based forwarding routing protocol," *Proc. Int. Conf. Comput. Technol. Inf. Manag.*, 2014.
[25] Z. J. Wang, J. Yang, G. Xie, and Mingtian, "OSPFv3 protocol simulation with colored Petri nets," in *International Conference on Communication Technology Proceedings, 2003. ICCT 2003.*, 2003, vol. 1, no. 60273070, doi: 10.1109/ICCT.2003.1209078.
[26] K. Jensen, L. M. Kristensen, and L. Wells, "Coloured Petri Nets and CPN Tools for modelling and validation of concurrent systems," *Int. J. Softw. Tools Technol. Transf.*, vol. 9, no. 3–4, pp. 213–254, May 2007, doi: 10.1007/s10009-007-0038-x.
[27] K. Farah, K. Chabir, and M. N. Abdelkrim, "Colored Petri nets for modeling of networked control systems," in *2019 19th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA)*, 2019, pp. 226–230, doi: 10.1109/STA.2019.8717215.
[28] I. S. Brito and J. P. Barros, "Coloured Petri net model of the bCMS system using CPN tools," in *2013 3rd International Workshop on Comparing Requirements Modeling Approaches (CMA@RE)*, 2013, pp. 7–12, doi: 10.1109/CMA-RE.2013.6664178.
[29] B. Mikolajczak and A. Singh, "TransCPN - Software Tool for Transformation of Colored Petri Nets," in *2009 Sixth International Conference on Information Technology: New*

Generations, 2009, pp. 211–216, doi: 10.1109/ITNG.2009.211.
[30] E. Mirzaeian, S. Ghaderi Mojaveri, H. Motameni, and A. Farahi, "An optimized approach to generate object oriented software test case by Colored Petri Net," in *2010 2nd International Conference on Software Technology and Engineering*, 2010, doi: 10.1109/ICSTE.2010.5608812.
[31] "Colored Petri Nets group Website." [Online]. Available: https://cs.au.dk/cpnets/. [Accessed: 06-Oct-2019].
[32] "CPN Tools Website." [Online]. Available: http://cpntools.org/. [Accessed: 06-Oct-2019].
[33] M. Zhou, F. DiCesare, and A. A. Desrochers, "A top-down approach to systematic synthesis of Petri net models for manufacturing systems," in *Proceedings, 1989 International Conference on Robotics and Automation*, pp. 534–539, doi: 10.1109/ROBOT.1989.100041.
[34] H. T. Jung and S. H. Joo, "Transformation of an activity model into a Colored Petri Net model," in *Trendz in Information Sciences & Computing(TISC2010)*, 2010, pp. 32–37, doi: 10.1109/TISC.2010.5714602.
[35] Yazhou Chen, "Research on a Top-Down collaborative modeling system," in *2008 12th International Conference on Computer Supported Cooperative Work in Design*, 2008, pp. 171–176, doi: 10.1109/CSCWD.2008.4536976.
[36] Y. Yang, S. Zhang, and X. Cheng, "Process Modeling of Top-down Collaborative Assembly Design Based on Petri Net," in *2007 10th IEEE International Conference on Computer-Aided Design and Computer Graphics*, 2007, pp. 401–406, doi: 10.1109/CADCG.2007.4407916.
[37] L. Ferrarini and M. Trioni, "Modeling shared resources with generalized synchronization within a Petri net bottom-up approach," in *Proceedings of IECON'94 - 20th Annual Conference of IEEE Industrial Electronics*, vol. 2, pp. 1105–1110, doi: 10.1109/IECON.1994.397946.
[38] C. M. M. Junior, R. M. S. Julia, and S. Julia, "Modeling Recursive Search Algorithms by Means of Hierarchical Colored Petri Nets and CPN Tools," in *2015 12th International Conference on Information Technology - New Generations*, 2015, pp. 788–791, doi: 10.1109/ITNG.2015.143.
[39] "Enhanced Interior Gateway Routing Protocol (EIGRP) Informational RFC Frequently Asked Questions - Cisco." [Online]. Available: https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/enhanced-interior-gateway-routing-protocol-eigrp/qa_C67-726299.html. [Accessed: 26-Aug-2018].
[40] J. Expósito, V. Trujillo, and E. Gamess, "Easy-EIGRP: A Didactic Application for Teaching and Learning of the Enhanced Interior Gateway Routing Protocol," in *2010 Sixth International Conference on Networking and Services*, 2010, pp. 340–345, doi: 10.1109/ICNS.2010.53.
[41] D. Savage, J. Ng, S. Moore, D. Slice, P. Paluch, and R. White, "Cisco's Enhanced Interior Gateway Routing Protocol (EIGRP)," RFC Editor, 2016.
[42] J. J. Garcia-Luna-Aceves, "Loop-free routing using diffusing computations," *IEEE/ACM Trans. Netw.*, vol. 1, no. 1, pp. 130–141, 1993, doi: 10.1109/90.222913.
[43] R. Albrightson, J. J. Garcia-Luna-Aceves, and J. Boyle, "EIGRP - A fast routing protocl based on distance vectors," in *Proc. Networld/Interop*, 1994, vol. 94, pp. 136–147.
[44] P. Jakma and D. Lamparter, "Introduction to the quagga routing suite," *IEEE Netw.*, vol. 28, no. 2, pp. 42–48, Mar. 2014, doi: 10.1109/MNET.2014.6786612.
[45] "Quagga Routing Suite Manual." [Online]. Available: http://www.nongnu.org/quagga/docs/docs-info.html. [Accessed: 01-Aug-2018].

[46] M. Kontšek, “Implementácia mechanizmov pre reštart susedských relácií do smerovacieho protokolu Quagga EIGRP [Diplomová práca],” University of Zilina, 2016.
[47] “Quagga-EIGRP source repository.” [Online]. Available: https://github.com/janovic/Quagga-EIGRP. [Accessed: 01-Aug-2018].
[48] J. Janovic, “Implementácia smerovacieho protokolu EIGRP v balíku Quagga, transportná časť [Diplomová práca],” University of Zilina, 2015.
[49] M. Perina, “Implementácia smerovacieho protokolu EIGRP v balíku Quagga, časť DUAL [Diplomová práca],” University of Zilina, 2015.
[50] P. Orság, “Implementácia smerovacieho protokolu EIGRP v balíku Quagga, časť riadenia [Diplomová práca],” University of Zilina, 2015.
[51] T. Hvorkový, “Implementácia filtrovania obsahu smerovacej informácie do smerovacieho protokolu Quagga EIGRP [Diplomová práca],” University of Zilina, 2016.
[52] “FreeRangeRouting - A new Quagga fork with more open development.” [Online]. Available: https://www.slideshare.net/apnic/freerangerouting-a-new-quagga-fork-with-more-open-development. [Accessed: 01-Aug-2018].
[53] “Welcoming FRRouting to The Linux Foundation.” [Online]. Available: https://www.linux.com/news/2017/4/welcoming-frrouting-linux-foundation. [Accessed: 01-Aug-2018].
[54] “FRRouting source repository.” [Online]. Available: https://github.com/FRRouting/frr. [Accessed: 01-Aug-2018].
[55] “EIGRP implementation for OpenBSD.” [Online]. Available: https://github.com/rwestphal/openbsd-eigrpd. [Accessed: 01-Aug-2018].
[56] Y. He, T. Liu, H. Zhong, and Q. Xiong, “EA-CPNsim: A CPN-Based Simulation Platform for Analysis and Defense Design of Internet End-Systems Targeted Attacks,” in *2009 Fourth International Conference on Frontier of Computer Science and Technology*, 2009, pp. 548–552, doi: 10.1109/FCST.2009.68.
[57] F. Yutao, S. Guiping, H. Liu, and Z. Siyu, “Study on a CPN-Based Auto-Analysis Tool for Security Protocols,” in *2012 Fourth International Symposium on Information Science and Engineering*, 2012, pp. 179–182, doi: 10.1109/ISISE.2012.45.
[58] T. R. Shmeleva, “Measuring subnets of colored Petri net models: Online technique for networks perfomance evaluation,” in *2017 International Conference on Information and Telecommunication Technologies and Radio Electronics (UkrMiCo)*, 2017, pp. 1–5, doi: 10.1109/UkrMiCo.2017.8095417.
[59] Z. Abderrazzak and E. Ahmed, “Cloud SaaS using MDA approach on a multiview models generate a SaaS from a colored Petri Net using view PNML,” in *2015 International Conference on Cloud Technologies and Applications (CloudTech)*, 2015, pp. 1–5, doi: 10.1109/CloudTech.2015.7336977.
[60] S. H. Askari, S. A. Khan, M. Haris, and M. Shoaib, “Pattern Based Model Reuse Using Colored Petri Nets,” in *2019 19th International Conference on Computational Science and Its Applications (ICCSA)*, 2019, pp. 32–38, doi: 10.1109/ICCSA.2019.000-7.

Zoznam vlastných publikácií

1. Marek Moravčík et al.: Teaching cloud computing in cloud computing, in ICETA 2017 15th IEEE international conference on Emerging eLearning technologies and applications, November 26-27, 2017, Starý Smokovec, The High Tatras, Slovakia, 2017, ISBN 978-1-5386-3296-3, s 319-324.

2. Jakub Hrabovský et al.: Influence of positive additive noise on classification performance of convolutional neural networks, in DISA 2018 : IEEE World Symposium on Digital Intelligence for Systems and Machines : proceedings. - 1. vyd. - Danver: Institute of Electrical and Electronics Engineers, 2018. - ISBN 978-1-5386-5102-5. - s. 175-181
3. Marek Moravčík et al.: Overview of cloud computing standards, in ICETA 2018 16th IEEE international conference on Emerging eLearning technologies and applications, November 15-16, 2018, Starý Smokovec, The High Tatras, Slovakia, 2018, ISBN 978-1-5386-7914-2, s 395-402.
4. Jana Uramová et al.: Infrastructure for Generating New IDS Dataset, in ICETA 2018 16th IEEE international conference on Emerging eLearning technologies and applications, November 15-16, 2018, Starý Smokovec, The High Tatras, Slovakia, 2018, ISBN 978-1-5386-7914-2, s 603-610.
5. Jozef Papán et al.: Overview of IP Fast Reroute Solutions, in ICETA 2018 16th IEEE international conference on Emerging eLearning technologies and applications, November 15-16, 2018, Starý Smokovec, The High Tatras, Slovakia, 2018, ISBN 978-1-5386-7914-2, s 417-424.
6. Martin Kontšek et al.: A Survey of the EIGRP Standard and Following Open-Source Implementations, in ICETA 2018 16th IEEE international conference on Emerging eLearning technologies and applications, November 15-16, 2018, Starý Smokovec, The High Tatras, Slovakia, 2018, ISBN 978-1-5386-7914-2, s 297-304.
7. Martin Kontšek et al.: Testing of the Current Open-source EIGRP Implementations, in ICETA 2018 16th IEEE international conference on Emerging eLearning technologies and applications, November 15-16, 2018, Starý Smokovec, The High Tatras, Slovakia, 2018, ISBN 978-1-5386-7914-2, s 291-296.
8. Ondrej Šuch et al.: Neural pairwise classification models created by ignoring irrelevant alternatives, in ITAT 2019: proceedings of the 19th conference Information Technologies - Applications and Theory. - ISSN 1613-0073. - 1. vyd. - [S.l]: CEUR-WS, 2019. - s. 66-70
9. Martin Kontšek et al.: Approaches and tools for network protocol modeling, in ICETA 2019 17th IEEE international conference on Emerging eLearning technologies and applications, November 21-22, 2019, Starý Smokovec, The High Tatras, Slovakia, 2019, ISBN 978-1-7281-4967-7, s 419-424.
10. Pavel Segeč et al.: Architecture design in private Cloud Computing, in ICETA 2019 17th IEEE international conference on Emerging eLearning technologies and applications, November 21-22, 2019, Starý Smokovec, The High Tatras, Slovakia, 2019, ISBN 978-1-7281-4967-7, s 709-714.
11. Pavel Segeč et al.: Network virtualization tools – analysis and application in higher education, in ICETA 2019 17th IEEE international conference on Emerging eLearning technologies and applications, November 21-22, 2019, Starý Smokovec, The High Tatras, Slovakia, 2019, ISBN 978-1-7281-4967-7, s 699-708.
12. Marek Moravčík et al.: Proposal of VoIP infrastructure and services for academia - case study, in ICETA 2019 17th IEEE international conference on Emerging eLearning technologies and applications, November 21-22, 2019, Starý Smokovec, The High Tatras, Slovakia, 2019, ISBN 978-1-7281-4967-7, s 540-545.
13. Jozef Papán et al.: Fast ReRoute error detection – implementation of BFD mechanism, in ICETA 2019 17th IEEE international conference on Emerging eLearning technologies and applications, November 21-22, 2019, Starý Smokovec, The High Tatras, Slovakia, 2019, ISBN 978-1-7281-4967-7, s 593-599.
14. Jana Uramová et al.: Best practise for creating Packet Tracer activities for distance learning and assessment of practical skills, in ICETA 2019 17th IEEE international conference on Emerging eLearning technologies and applications, November 21-22, 2019, Starý Smokovec, The High Tatras, Slovakia, 2019, ISBN 978-1-7281-4967-7, s 784-790.