# Monitorovanie softvérovo definovanej siete

Martin Baláž, Lukáš Vrba

***Abstract*—Monitorovanie ako aj virtualizácia SDN (software define network) sietí nie je až tak prebádaná oblasť. Tvorcovia pôvodného článku sa zameriavali vytvorením monitorovacieho nástroja SOFTmon. Pôvodnou myšlienkou článku bolo vytvorenie nezávislého monitorovacieho nástroja od NOS (network operating system). V našej práci sa dočítate ako sme nástroj otestovali na virtuálnej sieti mininet. Taktiež ako sme konfigurovali jednotlivé topológie a aké technológie a prostredie sme využívali. Okrem samotnej funkcionality SOFTmonu sme sa zaoberali aj jeho architektúrou a porovnaním s inými monitoravacími nástrojmi, kde sme sa snažili identifikovať ich podstatné vlastnosti. Počas oboznámenia sa s nástrojom sme identifikovali jeho silné a slabé stránky. Medzi slabé stránky patrila vizualizácia topológie monitorovanej siete a práve preto sa v článku zaoberáme aj na nami doimplementovanú vizualizáciu topológie. Okrem našej implementácie sa jemne venujeme aj implementácii SOFTmonu od pôvodných autorov.**

***Keywords*—*SOFTmon, mininet, floodlight, OpenFlow, SDN, moniroting, tools.***

## I. Introduction

Pojem SDN (software defined network) tu bol už dávnejšie, no v poslednej dobe sa stáva stále viac populárnejší. SDN je novým prístupom k CC (cloud computing), ktorý zefektívňuje správu siete a umožňuje programovo efektívnu konfiguráciu siete s cieľom zlepšiť výkon a monitorovanie siete. Aká je vlastne výhoda SDN sietí oproti normálnym sieťam? Veľkou výhodou je práve samostatná kontrolná vrstva. Predstavme si situáciu v ktorej chceme aktualizovať sieť alebo potrebujeme vykonať isté zmeny. V prípade, že by sme operovali na klasickej sieti bolo by potrebné zozbierať všetky smerovacie zariadenie a aktualizovať ich každé osobitne. Dôvodom takejto akcie by bol fakt, že v klasickej sieti má každé smerovacie zariadenie vlastnú riadiacu jednotku, ktorá sa stará o to, či sa spojenie nadviaže alebo nie. Na druhej strane pokiaľ vykonávame zmeny v SDN sieti, tak máme samostatnú riadiacu jednotku, ktorá v sieti kontroluje všetky smerovacie zariadenia. Teda ak chceme vykonať aktualizácie alebo isté zmeny stačí nám aktualizovať iba spomínanú kontrolnú jednotku. Tým ušetríme čas aj peniaze a to pridáva SDN na atraktivite.

V našej práci sa zaoberáme monitorovaním SDN sietí. Bližšie sa venujeme open source monitorovaciemu nástroju SOFTmon. Autori poukazujú na fakt, že aj keď monitorovacie riešenia existujú a sú celkovo komplexné, žiadny z nich sa nevenuje monitorovaniu SDN siete. Hlavným cieľom práce je preto analýza samotného nástroja SOFTmon; testovanie a overenie správnosti implementácie riešenia. Ďalej sme sa rozhodli v rámci spoznávania nástroja SOFTmon o jeho rozšírenie v rámci zobrazenia topológie siete v grafickej podobe.

### A. Motivácia a existujúce riešenia

Načo je práve monitorovanie siete dobré? Jednoduchá odpoveď je, že jednoducho chceme zistiť či je sieť dostatočné výkonná. V rámci monitorovania vieme sledovať viacero aspektov. Jedným z aspektov je meranie latencie založené na klientovi prostredníctvom dotazov založených na protokole ICMP (internet message control protocol) alebo sieťových uzlov pomocou SNMP (simple network management protocol). Taktiež vieme sledovať slabé stránky siete, inak povedané, v ktorých uzloch je prenos dát pomalší. Medzi iné aspekty patrí priepustnosť siete. Je to maximálna rýchlosť, za ktorú je možné niečo spracovať. Najvyššou hodnotou je maximálna rýchlosť spracovania. V klasických sieťach sa na takéto monitorovanie používajú nástroje ako je Zabbix alebo Nagios. V nasledujúcej tabuľke (Fig.1) je uvedené porovnanie základných vlastností klasických riešení v porovnaní so SOFTmon. [5] [6]

### B. SOFTmon architektúra

Tvorcovia SOFTmon monitorovacieho nástroja sa snažili o vytvorenie nástroja nézávislého na NOS (network operating system). Snažili sa v rámci jednotlivých NOS pridať metódy na meranie sieťovej premávky ako aj zabezpečiť ich vizualizáciu v grafovej podobe v reálnom čase.

SOFTmon interaguje s Northbound NOS rozhraním, ktoré umožňuje zložke siete komunikovať s komponentami vyššej úrovne akou je napríklad sieťová riadiaca jednotka. SOFTmon je podniková aplikácia, ktorá oplýva vrstevnou architektúrou. Najnižšou vrstvou je údajová vrstva. Môžu sa v nej nachádzať IO súbory ako aj databáza. Okrem spomínaného poskytuje podporu REST komunikácie. Prečo práve REST komunikácia? Je to z dôvodu, že v čase keď autori SOFTmon vyvíjali, používala väčšina open-source sieťových operačných systémov na dátovú komunikáciu práve REST.

Tvorcovia sa snažili vytvoriť nástroj, ktorý bude nezávislý na jednotlivých NOS a preto použili aj oni REST komunikáciu. To im ušetrilo čas, ktorý by inak museli vynaložiť na vytvorenie medzikomunikačných modulov na transformáciu dát. Tým pádom ak by bol záujem využiť iný NOS ako použili autori nie je časovo nákladné a implementačne náročné využívať SOFTmon monitorovanie aj s iným NOS. Tým sa postarali o nezávislosť SOFTmon na konkrétnom NOS. Ďalšou vrstvou je abstraktná vrstva REST-Connector, ktorá prostredníctvom NOS konektora alebo príslušného komunikačného modulu získa dátový model a metódy, ktoré sa v sieťovom operačnou systéme využívajú. Tento dátový model si SOFTmon uloží a ďalej využíva na spracovanie a vykonávané výpočty potrebne pre zobrazovania monitorovania v reálnom čase spôsobom, že dátový model vypočíta metrické údaje výkonnosti pomocou štatistík poskytnutých zo sieťového operačného systému, ktorý opäť získal tieto informácie cez OpenFlow zo sieťových uzlov.

| Zabbix | Nagios | SOFTmon | Nami upravený SOFTmon |
|---|---|---|---|
| Zber dát | Zber dát | Stav prepínačov | Stav prepínačov |
| Flexibilné vstupné definície | Rozšírený správca konfigurácie (verzia XI) | Stav portov | Stav portov |
| Vysoko konfigurovateľné upozornenia | Konfigurovateľné používateľské upozornenia | Stav tokov | Stav tokov |
| Vytváranie grafov v reálnom čase | Grafy plánovania výkonnosti a kapacity | Vytváranie grafov v reálnom čase | Vytváranie grafov v reálnom čase |
| Rozsiahle možnosti vizualizácie (zahŕňa vizualizáciu topológie) | Konfigurovateľný dashboard | | Vizualizácia topológie |
| Automatické vyhľadávanie sieťových zariadení | Automatické vyhľadávanie sieťových zariadení | Automatické vyhľadávanie sieťových zariadení | Automatické vyhľadávanie sieťových zariadení |
| Vlastné API | | Northbound API | Northbound API |
| Plnohodnotný a ľahko rozšíriteľný agent | Plnohodnotný a ľahko rozšíriteľný agent | Plnohodnotný a ľahko rozšíriteľný agent | Plnohodnotný a ľahko rozšíriteľný agent |
| REST API | | REST API | REST API |
| Web rozhranie | Textové súbory, Webové rozhranie (verzia XI) | Desktop rozhranie | Desktop rozhranie |
| PHP | C | Java | Java |
| SSH, DNS, DHCP, FTP, IMAP, POP3, TCP, UDP | SSH, DNS, DHCP, FTP, IMAP, POP3, TCP, UDP | SSH, DNS, DHCP, FTP, IMAP, POP3, TCP, UDP | SSH, DNS, DHCP, FTP, IMAP, POP3, TCP, UDP |
| Možnosti webového monitorovania | Množstvo pluginov | | |
| Ukladanie historických údajov | Ukladanie historických údajov | | |
| Použitie šablón | | | |
| Systém povolení | Správy o dostupnosti SLA | | |
| Binárny systémový daemon (jazyk C) | | | |
| SNMP, ICMP | SNMP, ICMP | | |

Fig. 1: Porovnanie monitorovacích nástrojov.

Samotný dátový model sa však skladá z troch hlavných prvkov. [8] Tými sú:

- Topológia
- Počítadlá
- Metriky

Topológia sa skladá zo všetkých sieťových zariadení, ktoré sú zvyčajne prepínače a ich prepojenia. Obsahuje aj počítadlá, ktoré uchovávajú štatistické údaje. Posledným prvkom sú metriky, ktoré sú potrebné na vizualizáciu výkonu siete. Model topológie a počítadlo objektov je založený predovšetkým na špecifikácii OpenFlow v1.3. [3]

Na najvyššej vrstve sa nachádza používateľské rozhranie. Toto rozhranie ponúka základné ovládacie prvky ako aj rozhranie na vizuálnu prezentáciu výkonnosti siete prostredníctvom grafov.

Autori sa už ďalej nezamýšľali nad prípadom, že zaujímavou informáciou je pre používateľa aj vizualizácia topológie siete s ktorou používateľ pracuje. SOFTmon síce ponúka v rámci grafického rozhrania zoznam nájdených sieťových komponentov ale slúži skôr ako ovládací prvok než vizuálna pomôcka.

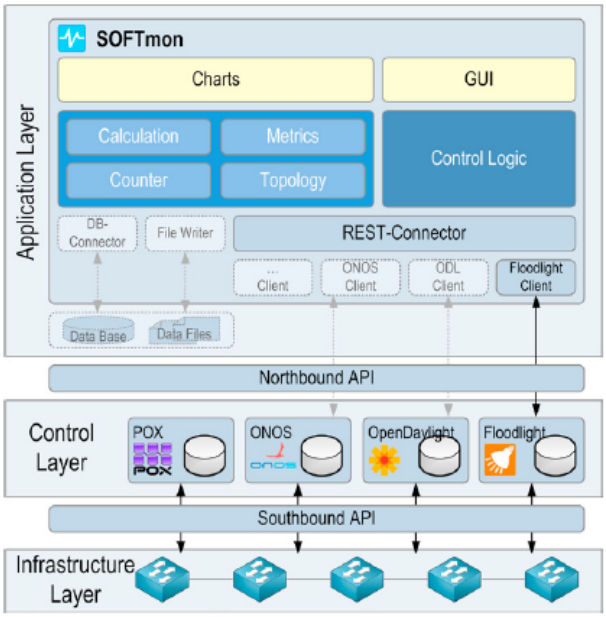


Fig. 2: SOFTmon vrstvová architektúra.

Pôvodne sme spomínali, že tvorcovia SOFTmon nástroja sa snažili vytvoriť nezávislý monitorovací nástroj pre rôzne NOS. Napriek tejto myšlienke by sme radi spomenuli fakt, že implementovaný bol iba Floodlight riadiaca jednotka. Preto je tento nástroj aj napriek skutočnosti, že je NOS nezávislí, v aktuálnej verzii závislí práve na jednom NOS. Plán autorov je v budúcnosti doplniť SOFTmon o nasledujúce NOS:

- Pox
- Onos
- OpenDaylight

Tieto NOS nieje možné v aktuálnej verzii využiť, aj keď sú v práci spomínané, tým ale chceli tvorcovia poukázať na skutočnosť, že samotné pridanie klienta môže byť z konfiguračného hľadiska nenáročné. Na druhej strane z programátorského hľadiska je vytvorenie takéhoto klienta celkom náročná robota, teda vieme pochopiť prečo implementovali iba Floodlight na prezentačné účely. Autori poukázali aj na fak, že pre Floodlight sa rozhodli z dôvodu, že je veľmi dobre zdokumentovaný a je rozšírený vo vedeckej komunite. [1]

### C. *Implementácia*

Vyššie uvedená architektúra naznačuje implementovanie monitorovacieho nástroja SOFTmon ako modulárneho riešenia, ktoré je jednoduché doplňať dodatočnou funkcionalitou. Monitorovací nástroj SOFTmon využíva na komunikáciu so sieťovým operačným systémom spojenie REST API rozhranie, ktoré vie rozpoznávať jednotlivé identifikátory zdrojov (URI). V týchto zdrojoch sa nachádzajú potrebné informácie uložené v údajovej štruktúre JSON (Java Script Object Notation). Táto štruktúra je na programovej úrovni preložená do príslušných tried. Triedy sa využívajú ako Data Model, s ktorým sa následne pracuje v aplikácii.

Hodnoty počítadiel sú k dispozícii len v časovo-diskrétnej forme. Preto možno výpočet metriky aproximovať pomocou príslušného časového intervalu t:

| Type | Metric | Unit | Counter | Unit |
|---|---|---|---|---|
| Switch Stats. | Flow Count | n | Active Entries (Tables) | n |
| | Packet Rate | n/s | Received Packets (Flows) | n |
| | Byte Rate | Byte/s | Received Bytes (Flows) | Byte |
| Port Stats. | RX Packet Rate | n/s | Received Packets | n |
| | TX Packet Rate | n/s | Transmitted Packets | n |
| | RX Byte Rate | Byte/s | Received Bytes | Byte |
| | TX Byte Rate | Byte/s | Transmitted Bytes | Byte |
| | RX Port Usage | % | Received Bytes | Byte |
| | TX Port Usage | % | Transmitted Bytes | Byte |
| Flow Stats. | Packet Rate | n/s | Received Packets | n |
| | Byte Rate | Byte/s | Received Bytes | Byte |

Fig. 3: Metriky.

Výpočet výkonnostných metrík vychádza zo štatistiky portov a tokov definovaných v štandarde OpenFlow. Vyššie uvedená tabuľka zobrazuje vypočítané metriky vo vzťahu k príslušným štatistikám (počítadlám). Počítadlá pre prepínače predstavujú agregované hodnoty, ktoré nie sú súčasťou špecifikácie OpenFlow, zatiaľ čo agregácia je vykonávaná prostredníctvom NOS. Pretože metrika výkonnosti m(t) je časovo závislá hodnota, môže sa vypočítať pomocou časového odvodenia zodpovedajúceho počítadla závislého od času c(t):

$$m(t) = \frac{d}{dt}c(t)$$

$$m(t) = \frac{c(t) - c(t - \Delta t)}{\Delta t}$$

Fig. 4: Vzorce.

Špecifikácia OpenFlow definuje počítadlá trvania štatistiky portov, ako aj štatistiky tokov (od OpenFlow verzie 1.3), ktoré by mohli byť použité ako časová základňa pre časovo závislé hodnoty počítadla. To by pomohlo dosiahnuť merania s teoretickou presnosťou až do jednej nanosekundy. Avšak počítadlo pre nanosekundovú časť trvania je označené ako voliteľné v špecifikácii OpenFlow. Maximálne možné a zaručené časové rozlíšenie je teda jedna sekunda. Okrem toho počítadlá trvania portu neexistujú v starších verziách OpenFlow. Bohužiaľ, jedna sekundu nestačí na dosiahnutie plynulej vizualizácie v reálnom čase. Riešením tohto problému bolo vytvorenie dodatočnej časovej základne generovaním a pridaním systémovej časovej pečiatky k hodnotám počítadla na základe času príchodu zodpovedajúceho objektu JSON prijatého od NOS. Je to tiež nevyhnutné pre pridanie funkcie zobrazovania historických hodnôt. [8]

### *D. Grafické rozhranie*

Monitorovaní nástroj SOFTmon ponúka grafické rozhranie pre koncového používateľa. Toto používateľské rozhranie je rozdelené do štyroch hlavných častí zobrazených na fig. č.5.

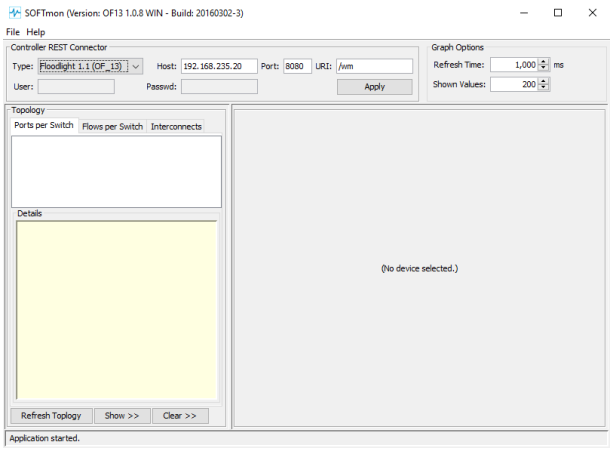


Fig. 5: SOFTmon grafické rozhranie.

1) REST Connector - Slúži na voľbu riadiacej jednotky SDN:
   a) Typ riadiacej jednotky;
   b) Ipv4 adresa riadiacej jednotky;
   c) Port riadiacej jednotky;
   d) Uri prefix;
   e) Apply tlačidlo na spustenie pripojenia.
2) Nastavenie parametrov merania:
   a) Rýchlosť zobrazovania;
   b) Rozpätie zobrazovania.
3) Stromová topológia
   a) Meranie Switch portov;
   b) Meranie Switch tokov;
   c) Medzispojenia;
   d) Zobrazenie podrobností o zvolenej jednotke.
4) Grafické okno zobrazujúce graf monitorovania.
   a) Graf monitorovanej jednotky;
   b) Podrobné informácie monitorovanej jednotky.

### *E. Rozšírenie*

V našej práci sme sa zamerali na rozšírenie funckionality monitorovacieho nástroja SOFTmon o vizualizáciu topológie monitorovanej siete. Doplnok programu sme zakomponovali do pôvodného rozloženia okna. V rámci vizualizácie topológie dostupnej siete predstavuje okno aj základný ovládací prvok monitorovacieho nástroja SOFmon. Voľbou Switch uzla sa spustí monitorovanie nad touto jednotkou, ak chceme vidieť monitorovanie konkrétneho portu switch uzla je potrebné mať označený switch aj príslušnú hranu, ktorá predstavuje zapojenie portu. Doplnok je zobrazený na fig. č.6. Na vizualizáciu je použitá doplnková knižnica JUNG. [7]

## II. Overenie práce

Na overenie pôvodného článku sme nakonfigurovali testovacie prostredie tak, aby sa zhodovalo so zdrojovým článkom v čo najväčšej miere a to v každom bode.

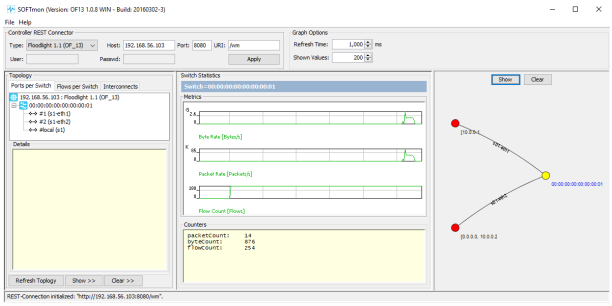

Fig. 6: SOFTmon rozšírenie o vizualizáciu topológie siete.

### A. Realizácia testovania v zdrojovom článku

Autori zdrojového článku realizovali vývoj a testovanie pomocou MININET emulačného nástroja, ktorý spúšťali na virtuálnom stroji. Vo svojej práci použili verziu MININET 2.2.1 a riadiacu jednotku Floodlight verziu 1.1. Záznam testovania z vývoja vo virtuálnom prostredí v publikácii neuviedli. Uviedli ale testovacie výsledky z reálneho prostredia lokálnej SDN sieťového zhluku s názvom ASOK, ktoré mali k dispozícii. V tabuľke č.* je zoznam použitých komponentov. Aby boli schopný monitorovať sieť, vytvorili pomocou Iperf testovacieho nástroja sieťovú premávku. [2] [4]

| System | Hardware | OS/Firmware |
|---|---|---|
| Cluster Node | 2 Intel Xeon Quad-Core 2,66 GHz, 32GB RAM | Ubuntu Server 14.04.3 LTS 64 bit |
| NOS Node | Pentium Dual-Core E5500 2,80GHz, 4GB RAM | Ubuntu Desktop 14.04.3 LTS 64 bit |
| Monitoring PC | Intel Core i7 2,8 GHz, 8GB RAM | Windows 7 Professional 64 bit |
| Switches | NEC IP8800/S3640-48T | OS-F3L Ver. 11.1.C.Af |

Fig. 7: Hardvér použitý na testovanie SOFTmon.

V článku uvádzajú postup testovania v dvoch hlavných testovacích scenároch. Prvý testovací scenár pojednáva o testovaní monitorovania na topológii zobrazenej na obrázku č. *, konkrétne prebieha monitorovanie na nec 1-1 switch porte 19 s nastavenou priepustnosťou 100 MBit/s. Iperf testovacím nástrojom nastavili server na stroji ASOK 13, a ako klient slúžil stroj ASOK04. Graf na obrázku č.*, zobrazuje priemernú rýchlosť 12,5 MByte/s. Nastavená rýchlosť na 100MBitov/s bola z dôvodu zabránenia zahltenia premávky na sieti. Druhý testovací scenár predstavoval sumu monitorovania premávky troch klientov, ASOK 04 až ASOK 06, na nec 3-1 switch porte 18. Priepustnosť aj server boli nastavené obdobne prvému testu. Graf na obrázku č.*, zobrazuje priemernú rýchlosť 37 MByte/s. Výsledky uvádzajú vo forme záznamu z okna monitorovacieho nástroja SOFTmon. Autroi sa odvolávajú na overenie rýchlosti priepustnosti v prvom a v druhom teste, k teoretickej priepustnosti, ktorá vyplýva z inicializačných nastavení priepustnosti. Sledovali, že pri prvom teste bola monitorovaná priepustnosť necc 1-1 switch portu 19 priemerne 12,5 MByte/s a druhého testu v priemere 37,5MByte/s čo odpovedá možnostiam priepustnosti pri zvolených nastaveniach. Autori ďalej uvádzajú, že zobrazované monitorovanie nie je v reálnom čase, ale s oneskorením, ktoré nepresahuje 25 milisekúnd, čo v konečnom dôsledku neovplyvňuje schopnosť používateľa nástroj využívať na monitorovanie sieťovej premávky a tak je toto oneskorenie z ich pohľadu zanedbateľné.

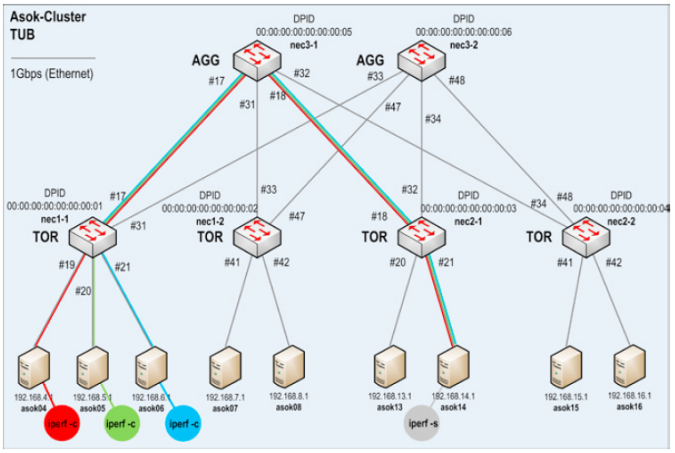


Fig. 8: Asok topológia.

### B. Overenie testovania zdrojového článku

V našej práci sme sa rozhodli priblížiť sa testovacím podmienkam čo najbližšie aby sme mohli vyviesť závery, ktoré majú svoje opodstatnenie. Pri realizácii sme sa ale stretli s problémami, ktoré nás donútili isté veci zmeniť.

Ako prvú s týchto vecí by sme radi spomenuli emulačný nástroj MININET. V zdrojovom článku ako je vyššie uvádzané autori využili verziu MININET 2.2.1, my sme na druhej strane v našom testovacom prostredí využívali MINININET verziu 2.2.2, keďže staršia veria už nebola kompatibilná s riadiacou jednotkou Floodlight verziou 1.1. Príčinou boli nedostupné funkcie, ktoré bránili pripojenie emulovanej siete na riadiacu jednotku. To by ale nemalo nijako ovplyvniť správny beh programu SOFTmon, keďže autori uvádzajú ako jediné podmienky, spustenia nástroja Floodlight verziu 1.1 a verziu protokolu OpenFlow 1.3, ktoré sme v našom overovaní využívali.

Pre dosiahnutie optimálneho testovania sme vytvorili rovnakú topológiu siete ako využívali autori vo svojej práci. Nemali sme k dispozícii datacentrum ASOK, ktoré autori využívali vo svojej práci na overenie správnosti riešenia, a tak sme sa rozhodli použitú topológiu simulovať vo virtuálnej sieti MININET. Obdobným spôsobom sme monitorovanie realizovali aj na testovacej topológii, fig. č. 8, ktorú autori využívali vo vývojovom prostredí. Na záver by sme radi uviedli aj základnú topológiu mininetu ktorá pozostáva z dvoch klientských zariadení a jedného switch zariadenia, na ktorom sme nastavovali vývojové prostredie nástroja SOFTmon, v ktorom sme následne testovali základnú funkcionalitu programu.

Radi by sme preto spomenuli, že jediný rozdiel medzi výsledkami dosiahnutými naším testovaním a testovaním autorov je v testovaní na reálnych strojoch, ktoré sme na rozdiel od autorov nemali k dispozícii.

### C. Priebeh testovania

Ako prvé bolo potrebné nakonfigurovať MININETVM, v ktorom sme simulovali sieť. Do MININETVM, bola stiahnutá, nainštalovaná a nakonfigurovaná SDN riadiaca jednotka Floodlight v1.1, ktorú autori uvádzajú ako potrebnú pre správne prepojenie medzi SOFTmon nástrojom a riadiacou jednotkou. Podrobný postup ako nakonfigurovať vývojové prostredie nájdete na github [9] stránke.

Druhý krok predstavovalo vytvorenie sieťovej topológie na overenie správnosti riešenia autorov zdrojového článku. Pre tento účel sme zvolili konfiguráciu pomocou python scriptu, ktorú nájdete na spomínanej github stránke. Pripojenie riadiacej jednotky na topológiu siete je potrebné vykonať s prepínačom, ktorý vymedzí verziu OpenFlow na verziu 1.3, ktorú autori článku uvádzajú ako nutnú podmienku správneho fungovania monitorovacieho nástroja SOFTmon.

Posledný krok predstavuje samotné monitorovanie sieťovej premávky vytvorenej pomocou Iperf testovacieho nástroja, ktorý nastaví priepustnosť na 100 Mbit/s. Výsledky autorov naznačujú priemernú rýchlosť prenosu na 12,5M Byte/s pri prenose z jedného zariadenia ASOK 04 na server zariadenie ASOK 13 a priemernú rýchlosť 37,5 MByte/s pre meranie priepustnosti portu 18 pre switch nec3-1, v ktorom sa združuje sieťová premávka pre Iperf nástroj všetkých 3 klientov (asok04, asok05, asok06).Tento fakt koreluje s nastavenou prenosovou rýchlosťou 300 Mbit/s.

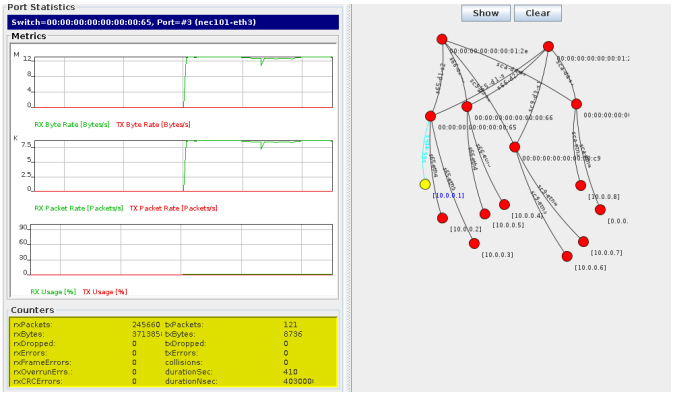


Fig. 9: Test č.1: asok14 server, asok04 client, switch nec101 port:03, bandwidth 100 MBit/s

### D. Dodatočné testovanie

Okrem hlavného testovania na overenie tvrdení autorov článku sme testovali SOFTmon aj na vlastných topológiach. Testy sú prezentované chronologicky podľa zložitosti topológie.

Prvá topológia, na ktorej sme testovali bola základná topológia, ktorá pozostáva z dvoch klientov h1 a h2 a swtich zariadenia s1.

Test prebiehal s rovnakými nastaveniami aké boli použité pri overení tvrdení autorov článku. Pomocou Iperf nástroja sa vytvoril server na h1 a klient na h2, s obmedzenou priepustnosťou na 100 MBit/s. Výsledky testu je možné pozorovať

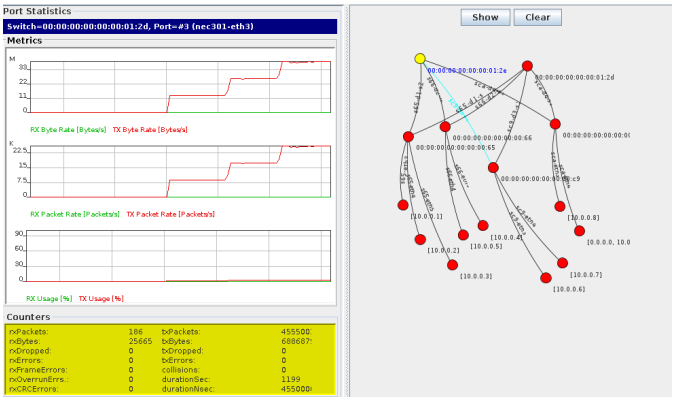


Fig. 10: Test č.2: asok14 server, sum: asok04-asok06 client, switch nec301 port:03, bandwidth 100 MBit/s

na fig. č 10, test č.3. Test bol zopakovaný zrkadlovo, na overenie správnosti monitorovacieho záznamu. Tento test je možné pozorovať na fig č.11, test č.4.

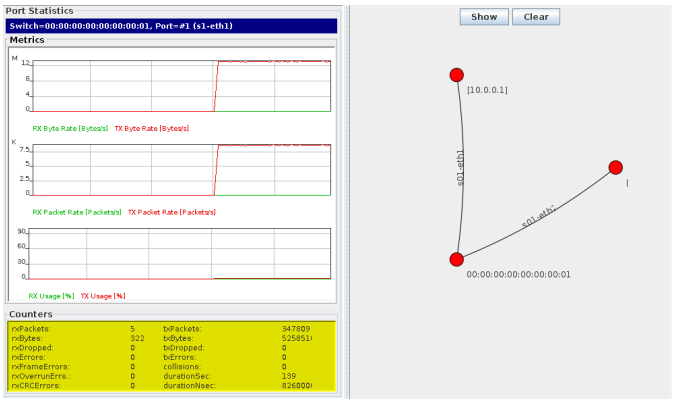


Fig. 11: Test č.3: h1 server, h2 client, s1 port:1, bandwidth 100 MBit/s

Druhou testovanou topológiou bola topológia ktorú autori použili pri vývoji monitorovacieho nástroja SOFTmon. Záznam z testovania je možné pozorovať na fig č.13, test č5.

Posledný test sme sa rozhodli realizovať na topológii s hĺbkou 3 a vetvením 2. Priepustnosť siete bola nastavená znova na 100 MBit/s. V práci sme neuvádzali všetky záznamy z testovania, tie sú dostupné na stránke projektu na githube, aj s podrobnými nastaveniami Iperf nástroja, tak isto formou snímky z obrazovky.

## III. Vyhodnotenie

V práci sme sa venovali podrobnejšej analýze monitorovacieho nástoja SOFTmon, ktorý slúži na monitorovanie SDN siete. Overovali sme tvrdenia autorov pomocou replikácie testu, na ktorý sa autori odvolávajú ako na test potvrdzujúci správnosť ich riešenia. Testovali sme aj vlastné testovacie

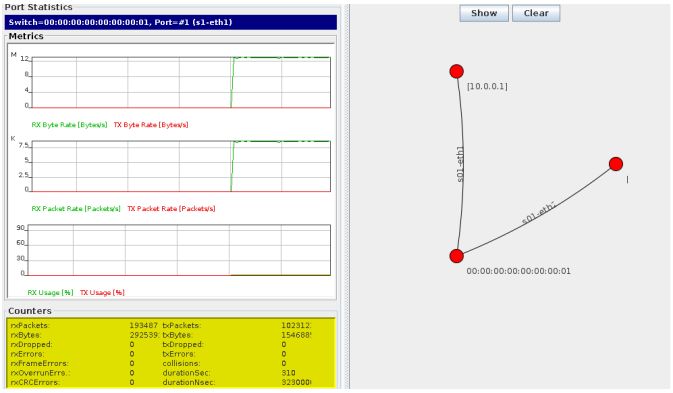


Fig. 12: Test č.4: h2 server, h1 client, s1 port:1, bandwidth 100 MBit/s

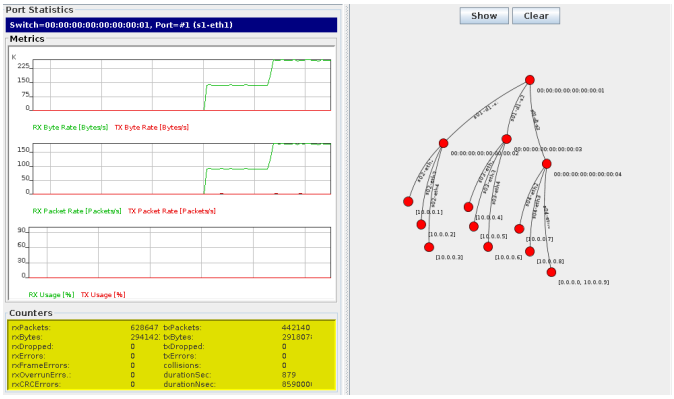


Fig. 13: Test č.5: h5 server, h1 and h2 client, s1 port: p1, bandwidth 100 MBit/s

scenáre, ktorými sme chceli potvrdiť alebo vyvrátiť ich tvrdenia. Podarilo sa nám zrekapitulovať inicializačné nastavenia s jediným rozdielom verzie MININET emulačného nástroja a faktu, že z dôvodu nedostupnosti testovacieho prostredia autorov sme sa rozhodli toto testovacie prostredie simulovať v MININETVM.

Po úspešnom nasadení testovacieho nástroja SOFTmon a zapojenia všetkých potrebných komponentov sa nám podarilo dokázať, že monitorovací nástroj, ktorý autori prezentujú funguje správne podľa predošlých tvrdení autorov. Je ale nutné poukázať na fakt, že sme narazili na niekoľko problémov spojených s nástrojom SOFTmon. Funkcionalita nie je úplná a aj napriek tvrdeniam autorov je nasadenie nástroja problematická úloha, ktorú odporúčame len používateľom, ktorí sa cítia schopnejší v počítačovej oblasti. Keďže sme museli upraviť zdrojový kód, ktorý bol zodpovedný za správne načítavanie JSON súborov z REST api, ktorú Floodlight ponúka, nemôžme tvrdiť, že tento monitorovací nástroj je pripravený na nasadenie na trh, nie je totižto možné program stiahnuť a nasadiť do produkcie bez programatorských zručností spojených s prepísaním zdrojového kódu.

Na druhu stranu by sme radi poukázali na fakt, že SOFT-

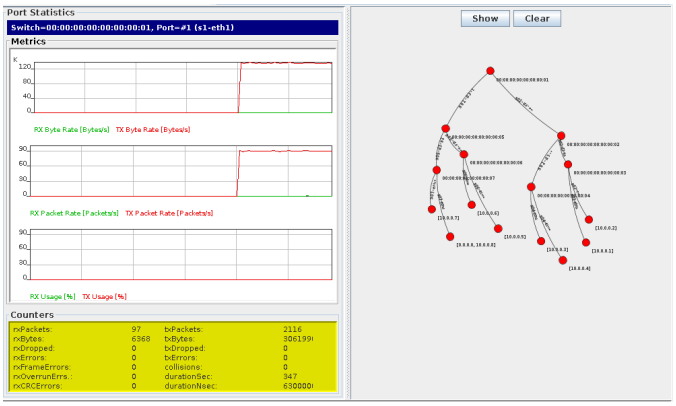


Fig. 14: Test č.6: h2 server, h7 client, s1 port:1 bandwidth 100 MBit/s

mon je naozaj modulárne naprogramovaný a nebolo problém zorientovať sa v zdrojovom kóde monitorovacieho nástroja SOFTmon. Toto tvrdenie máme podložené nasadením vizualizačného okna topológie, ktorým by sme radi prispeli k budúcemu vývoju SOFTmon.

Na záver by sme zhrnuli, že aj keď základná funkcionalita, ktorú autori v monitorovacom nástroji SOFTmon vyvinuli funguje správne, je nutné na tomto programe ešte zapracovať. Open source komunita je rozsiahla a preto dúfame, že tak isto ako sme prejavili záujem o tento nástroj my, nájdu sa aj ďalší nadšenci, ktorí dostanú monitorovací nástroj SOFTmon do produkčnej podoby.

## Poďakovanie

Radi by sme pri tejto príležitosti poďakovali autorom článku za vytvorenie monitorovacieho nástroja SOFTmon ako aj za fakt, že tento nástroj ponúkli ako OpenSource. Tak isto by sme radi poďakovali Ing. P. Helebrandtovi, PhD. za vedenie a pomoc poskytnutú pri riešení tejto práce.

## References

[1] Floodlight, P, Floodlight controller - REST api, 2017, https://floodlight.atlassian.net

[2] iperf2, 2017, https://sourceforge.net/projects/iperf2/

[3] OpenNetworkingFoundation, OpenFlowswitchspecification1.3.0., 2012, https://www.opennetworking.org

[4] Project floodlight, 2016, http://www.projectfloodlight.org/floodlight/

[5] Zabbix::Theenterprise-classmonitoringsolutionforeveryone, 2017, http://www.zabbix.com

[6] Nagios–theindustrystandardinitinfrastructuremonitoring, 2017, https://www.nagios.org

[7] Jung–Java Universal Network/Graph Framework, 2010, http://jung.sourceforge.net/

[8] M. Hartung, M. Körner, SOFTmon - Traffic Monitoring for SDN, In Procedia Computer, ScienceDirect, p. 516-523, 2017, ISSN 1877-0509

[9] 2017, https://github.com/aks-2017/semestralne-zadania-semestralne-zadanie-xbalazm1-xvrbal