Implementácia grafického používateľského rozhrania pre distribuovaný firewall založenom na OpenFlow

Dávid BEŇO, Šimon HARVAN

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova 2, 842 16 Bratislava, Slovakia
xbenod@stuba.sk, xharvan@stuba.sk

Abstract. Softvérovo definované siete (SDN) sú technológia, ktorá bude jadrom sietí ďalších generácií. Veľa spoločností a organizácií začalo používať SDN aplikácie. Toto dáva administrátorom flexibilitu pri implementovaní vlastných sieti. Ale zároveň vyvstávajú nové bezpečnostné problémy. Aby sme mohli zabezpečiť SDN siete potrebujeme silný firewall. Aktuálne už existujú firewally, ale majú určité nevýhody. Jeden z hlavných nedostatkov existujúcich riešení je, že sú umiestnené na jednom centrálnom zariadení a celý firewall zlyhá ak zlyhá jedno zariadenie. Ďalší nedostatok existujúcich riešení je, že väčšina z nich sú firewall-y druhej vrstvy. V tomto článku popisujeme implementáciu grafického používateľského rozhrania pre distribuovaný firewall, kde sa každý prepínač v sieti môže správať ako firewall. Používame RYU riadič založený na Python-e. RYU už obsahuje firewall, s ktorým sa dá komunikovať pomocou REST funkcii a takto nastavovať jeho pravidlá. Tento firewall a k nemu nami vytvorené GUI sme testovali pomocou emulátora Mininet Emulátor, ktorý je nainštalovaný v Ubuntu 14.04, ktoré je nainštalované pod VirtualBox.

1 Úvod

V tradičných sieťach je veľmi ťažké dynamicky upravovať smerovače, prepínače, vyrovnávače zaťaženia, konfigurácie IDS a IPS podľa požiadaviek. Avšak využitie SDN toto dovoľuje pretože oddeľuje prepravnú rovinu od riadiacej roviny. SDN umožňuje centralizovanému ovládaču dynamicky spravovať všetky zariadenia. V tradičných sieťach musí byť každé zariadenie konfigurované individuálne. Riadiaca rovina a prepravná rovina komunikujú pomocou protokolu OpenFlow.

Hlavnou funkciou SDN je určiť, čo sa má urobiť s paketom, ktorý bol prijatý prepínačom OpenFlow, a to konzultáciou s tabuľkou vstupov. Keď sa pakety dostanú na prepínač, polia hlavičiek paketov sa porovnajú s položkami toku tabuľky. Ak nastane zhoda, činnosť sa vykoná podľa špecifikácie v položke toku. Ak nenastane žiadna zhoda, tak paket bude odoslaný do riadiacej jednotky podľa vstupného prietoku. Toto sa tiež nazýva správa *Packet_In*. Potom riadič rozhodne, čo sa má urobiť s paketom podľa aplikačnej logiky. Následne môže poveriť prepínač, aby preposielal paket alebo môže pridať tok do tabuľky toku prepínača, aby sa tak s podobnými typmi paketov mohlo v budúcnosti rýchlejšie pracovať.

2 Analýza

2.1 Mininet

Mininet je emulátor siete, ktorý vytvára sieť virtuálnych staníc (ďalej hostov), prepínačov, a ovládačov. Mininet hostovia prevádzkujú štandardný sieťový softvér Linux a jeho prepínače podporujú OpenFlow pre vysoko flexibilné vlastné smerovanie a softvérové siete. Spoločnosť Mininet podporuje výskum, vývoj, učenie, prototypovanie, testovanie, ladenie a všetky ostatné úlohy, ktoré by mohli mať prospech z úplnej experimentálnej siete na prenosnom alebo inom počítači.

Mininet poskytuje jednoduchý spôsob, ako dosiahnuť správne správanie systému (a v rozsahu podporovanom vaším hardvérom, výkonom) a experimentovať s topológiami. Mininetové siete pracujú so skutočným kódom vrátane štandardných sieťových aplikácií Unix / Linux, ako aj skutočného jadra Linuxu a sieťového zásobníka.

Z tohto dôvodu sa kód, ktorý vyvíjame a otestujeme v službe Mininet, pre radič

OpenFlow, zmenený prepínač alebo host, môže prejsť na skutočný systém s minimálnymi zmenami, pre testovanie v reálnom svete, hodnotenie výkonnosti a nasadenie. Dôležité je to, že návrh, ktorý pracuje v programe Mininet, sa zvyčajne môže presunúť priamo na hardvérové prepínače pre presmerovanie paketov.

2.2 Firewall

Brána firewall je sieťový bezpečnostný systém založený na hardvéri alebo softvéri, ktorý používa pravidlá na kontrolu prichádzajúcej a odchádzajúcej sieťovej prevádzky.

Brána firewall funguje ako bariéra medzi dôveryhodnou a nedôveryhodnou sieťou. Firewall kontroluje prístup k zdrojom siete prostredníctvom pozitívneho modelu kontroly. To znamená, že jediná prevádzka povolená na sieť je definovaná v pravidlách brány firewall - všetka ďalšia prevádzka bola zamietnutá.

2.3 SDN

Softvérovo definovaná sieť (SDN) je zastrešujúcim výrazom, ktorý zahŕňa niekoľko druhov sieťových technológií zameraných na to, aby bola sieť agilná a flexibilná ako virtualizovaná serverová a ukladacia infraštruktúra moderného dátového centra. Cieľom SDN je umožniť sieťovým inžinierom a administrátorom rýchlo reagovať na meniace sa požiadavky.

V sieťach definovaných softvérom môže správca siete tvarovať prevádzku z centralizovanej ovládacej konzoly bez toho, aby sa musel dotýkať jednotlivých prepínačov a môže poskytovať služby všade tam, kde sú potrebné v sieti, bez ohľadu na to, aké konkrétne zariadenia server alebo iné hardvérové súčasti sú pripojené k sieti. Kľúčovými technológiami implementácie SDN sú funkčné oddelenie, sieťová virtualizácia a automatizácia prostredníctvom programovateľnosti.

2.4 Ryu

Ryu kontrolór je otvorený kontrolór pre SDN siete, ktorý je navrhnutý tak, aby zvyšoval agilitu siete tým, že umožňuje jednoduchú správu a spôsob kontrolovania prevádzky. Vo všeobecnosti je kontrolór „mozog“ prostredia SDN, oznamuje informácie smerom k prepínačom a smerovačom ale aj aplikáciám a API na severi.

Ryu poskytuje softvérové komponenty s dobre definovanými aplikačnými programovými rozhraniami (API), ktoré uľahčujú vývojárom vytváranie nových aplikácií pre správu a riadenie siete. Tento prístup komponentov pomáha organizáciám prispôsobiť nasadenia tak, aby vyhovovali ich špecifickým potrebám prevádzky. Vývojári môžu rýchlo a ľahko upraviť existujúce komponenty alebo implementovať svoje vlastné, aby zabezpečili, že základná sieť môže spĺňať meniace sa požiadavky svojich aplikácií.

Zdrojové kódy ryu sú dostupné na serveri github a sú spravované Ryu komunitou. Ryu je napísaný v jazyku Python a je pod Apache 2.0 licenciou, takže je dostupný pre každého. Ryu plne podporuje OpenFlow, čo je jeden z prvých a častou používaných komunikačných štandardov. OpenFlow sa využíva práve na komunikáciu s prepínačmi a smerovačmi – pre nastavenie správy prevádzky siete.

2.5 OpenFlow

OpenFlow je považovaný za jeden z prvých SDN štandardov. Definuje komunikačný protokol v prostredí SDN, ktorý umožňuje SDN riadiču priamo komunikovať s prepínačmi a smerovačmi, fyzickými aj virtuálnymi, aby sa mohli lepšie prispôsobiť meniacim sa obchodným požiadavkám.

2.6 REST API

REST (Representational state transfer) je webový štandard založený na architektúre a používaní protokolu HTTP pre komunikáciu. REST navrhol v roku 2000 Roy Fielding (spoluautor protokolu HTTP) v rámci svojej dizertačnej práce. Rozhranie REST je použiteľné pre jednotný a jednoduchý prístup k zdrojom. To môžu byť dáta rovnako ako napríklad stavy aplikácie. Na rozdiel od iných prístupov ako napríklad SOAP nie je REST orientovaný procedurálne ale dátovo. Každý zdroj má vlastný identifikátor URI a REST definuje štyri základné metódy na prístup k nim – CRUD (Create, Read, Update, Delete). Konkrétne sa jedná o HTTP metódy GET, PUT, POST, DELETE.

3 Grafické používateľské rozhranie

Pre lepšiu vizualizáciu aktuálneho stavu firewallu na prepínačoch, bola vytvorená grafická aplikácia. Je určená pre akýkoľvek počítač pripojený k stroju, na ktorom je spustený riadič siete. Aplikácia je založená na frameworku

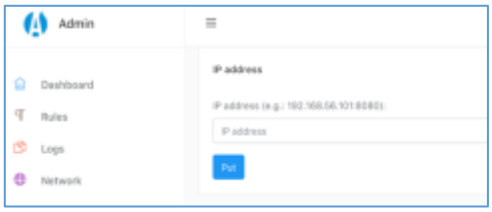
Bootstrap a všetky závislosti sú manažované balíčkovým manažérom NPM. Primárnym cieľom frameworku Bootstrap je responzívnosť a použiteľnosť aplikácií na rôznych platformách.

Nami pripravené web rozhranie bude mať nasledovné funkcie:

- Získanie IP stroja, na ktorom je spustený kontrolér siete.
- Prezeranie statusu všetkých firewall prepínačov
- Zapnutie/Vypnutie firewall prepínačov
- Prezeranie firewall pravidiel všetkých prepínačov
- Pridanie pravidla pre špecifický prepínač
- Odobranie pravidla pre špecifický prepínač

Získanie IP stroja, na ktorom je spustený riadič siete

Pri prvom spustení aplikácie, používateľ zadá IP adresu stroja, na ktorom beží controlér siete a spravuje prepínače a routre pomocou protokolu OpenFlow.



Obr. 1.: *Zadanie IP*

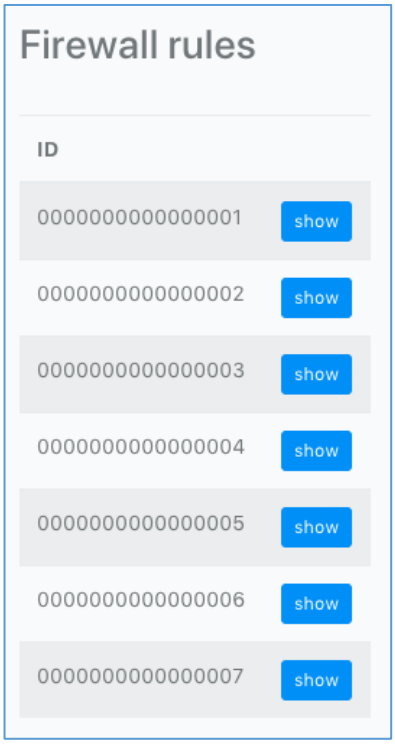
Prezeranie statusu všetkých firewall prepínačov

Po štarte aplikácie sa zobrazí zoznam všetkých prepínačov v sieti a aktuálny stav firewalu na nich. Firewall na každom prepínači sa dá zapnúť alebo vypnúť. Tiež sa dajú zapnúť/vypnúť firewally na všetkých prepínačoch naraz.

Obr. 2.: *Zoznam prepínačov*

Prezeranie pravidiel firewallu na všetkých prepínačoch

Aplikácia ponúka pre používateľa možnosť zobrazenia pravidiel na všetkých prepínačoch. Používateľ si zo zoznamu všetkých prepínačov vyberie ten, na ktorom si chce prezrieť pravidlá firewallu. Pravidlá sa zobrazia po stlačení tlačidla *Show*.

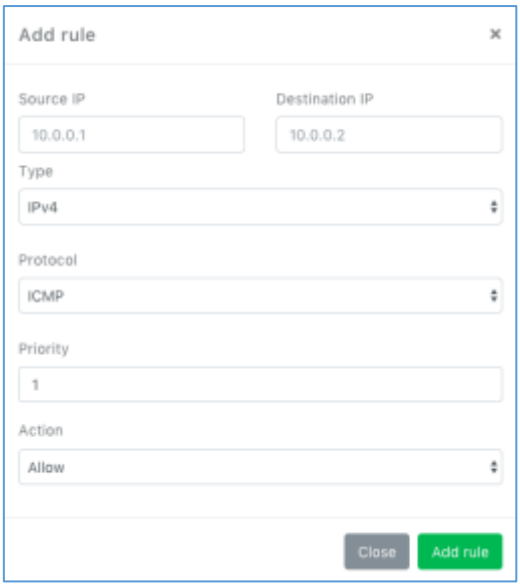


Obr. 3.: Vyber prepínaču

Pridanie pravidla pre špecifický prepínač

Z hlavného menu si používateľ vyberie prepínač, s ktorým chce pracovať. Po stlačení tlačidla *Add* rule aplikácia používateľovi zobrazí formulár pre vytvorenie pravidla pre daný prepínač.

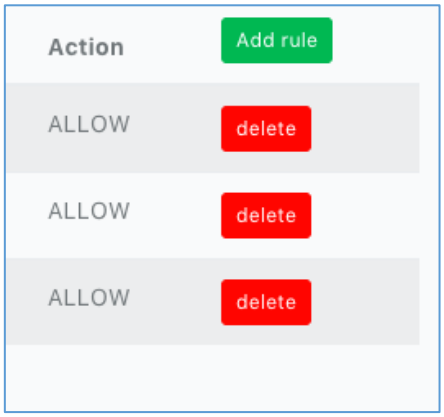
Formulár obsahuje pole pre zadanie zdrojovej a cieľovej IP hostu. Pole pre číslo protokolu a tiež pole pre port, na ktorom sa komunikuje daným protokolom.



Obr. 4.: *Pridanie pravidla*

Odobranie pravidla zo špecifického prepínaču

Pre daný prepínač sa zobrazí zoznam všetkých pravidiel. Pri každom pravidle je tlačidlo, ktoré používateľovi ponúka možnosť odobrať pravidlo.

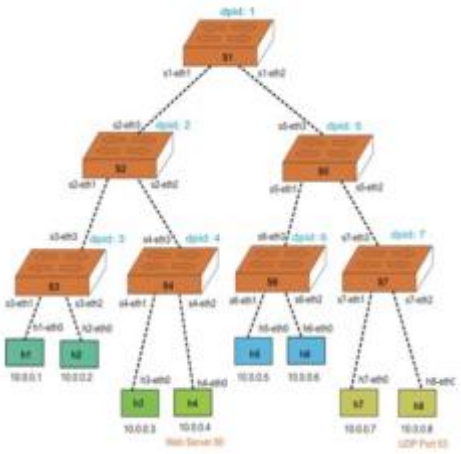


Obr. 5.: *Odobranie pravidla*

4 Experiment

Ako testovacie prostredie sme použili Ubuntu 16.04 spustené pomocou programu VirtualBox. Pre simulovanie SDN siete sme použili simulovací nástroj Mininet. Ako riadič pre SDN sieť sme si vybrali RYU a to z dôvodu, že už obsahuje firewall, pre ktorý sme dorobili grafické používateľské rozhranie. Pre experiment sme vytvorili stromovú topológiu príkazom:

sudo mn --topo tree,3 --mac --switch ovsk protocols=OpenFlow13 --controller remote -x



Obr. 6.: *Topológia siete*

Vytvorená sieť pozostáva zo siedmich prepínačov a ôsmych hostiteľských počítačov (staníc?). Každá stanica má pridelenú IP v rozsahu od 10.0.0.1 po 10.0.0.8. Ďalej na stanici Host 4 je spustený web server na port 80 pomocou programu „SimpleHTTPServer“ a na stanici Host 8 je spustený UDP server pomocou programu „nc“.

5 Testovanie

Pre dokázanie funkčnosti nášho riešenia, sme vytvorili štyri testovacie scenáre. Naša GUI aplikácia ponúka možnosti pre nastavenie takých pravidiel, ktoré zodpovedajú nasledujúcim testovacím scenárom:

1. test – h1 pingne h2
2. test – h1 pingne h7
3. test – h1 pristúpi na web server na h4
4. test – h7 pristúpi na UDP server na h8

V prípade, že po nastavení pravidiel cez našu aplikáciu, bude komunikácia v sieti fungovať tak ako je v scenároch popísaná, bude to znamenať, že naše riešenie je správne a funkčné.

Pre správne nastavenie pravidiel na základe testovacích scenárov, sme vytvorili tabuľku pravidiel:

Prepínač	Zdroj. IP	Cieľ IP	Protokol	Port
S3	10.0.0.1	10.0.0.2	1 ICMP	-
S3, S2, S1, S5, S7	10.0.0.1	10.0.0.7	1 ICMP	-
S3, S2, S4	10.0.0.1	10.0.0.4	6 TCP	80
S7	10.0.0.7	10.0.0.8	17 UDP	53

Tabuľka 1.: *Pravidlá siete*

Prvý scenár nastavíme tak, že na prepínači S3 povolíme komunikáciu cez ICMP medzi IP adresami h1 a h2.

Druhý scenár je zložitejší ako prvý a to z dôvodu, že bude potrebné nastaviť pravidlá až na piatich prepínačoch. Nastavujú sa podobne ako pri prvom scenári

Tretí scenár už pracuje aj s nastavovaním portu. Na h4 sme spustili jednoduchý HTTP server a z h1 naň pristupujeme príkazom curl. Pravidlá pre povolenie danej komunikácie je nutné nastaviť na troch prepínačoch.

Štvrtý scenár je podobný ako tretí. Na h8 sme spustili jednoduchý UDP receiver a z h7 naň posielame reťazec znakov. Pravidlo je nutné nastaviť na prepínači S7. Tiež je potrebné nastaviť port pre UDP komunikáciu.

6 Záver

Distribuovaný firewall funguje správne a je schopný zvládnuť prevádzku protokolu ICMP, TCP a UDP. Do všetkých prepínačov alebo konkrétnych prepínačov môžeme vložiť pravidlá na základe našich požiadaviek. To znamená, že GUI funguje tiež správne a správne komunikuje s jednotlivými prepínačmi. Budúca práca môže zahŕňať testovanie tohto firewallu na skutočnom hardvéri.

References

[1] Hu, Fei, Qi Hao, and Ke Bao, "A survey on software-defined network and openflow: from concept to implementation," IEEE Communications Surveys & Tutorials 16, no. 4 (2014): 2181-2206.

[2] Mccauley, J. "Pox: A python-based openflow controller." (2014).

[3] Jarraya, Yosr, Taous Madi, and Mourad Debbabi, "A survey and a layered taxonomy of software-defined networking," IEEE Communications Surveys & Tutorials 16, no. 4 (2014): 1955-1980.

[4] Fernandez, Marcial P, "Comparing openflow controller paradigms scalability: Reactive and proactive," In Advanced Information Networking and Applications (AINA), 2013 IEEE 27th International Conference on, pp. 1009-1016. IEEE, 2013.

[5] Javid, Tariq, Tehseen Riaz, and Asad Rasheed, "A layer2 firewall for software defined network," In Information Assurance and Cyber Security (CIACS), 2014 Conference on, pp. 39-42. IEEE, 2014.