УДК 621.391 DOI: 10.31673/2518-7678.2020.041624

Березнюк А.В., аспірант; **Лазебний С.Г.,** аспірант; **Шевцов П.В.,** аспірант;
Гринкевич Г.О., к.т.н.; **Макаренко А.О.,** д.т.н.

АНАЛІЗ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ НЕОБХІДНОГО ДЛЯ КОНТРОЛЮ ЗА ПАКЕТАМИ ДАНИХ В ПРОГРАМНО - КОНФІГУРОВАНИХ МЕРЕЖАХ

Bereznyuk A.V., Lazebnyi S.H., Shevtsov P.V., Grynkevych G.A., Makarenko A.O. Analysis software needed to ensure control over data packets in the software - configured networks.

The paper presents the concept of Software-Defined-Networking, and the history of programmable networks and associated review of the latest standards. One of them, the OpenFlow protocol, played an important role in promoting the concept of SDN, significantly influenced research and industry. Consider in more detail the basic principles of OpenFlow. To debug the software, it is necessary to ensure the control of packets in the network, although formally verification can be used to determine the correct flow. This is necessary in order to generate detailed network images and also to inspect function parameters under certain network conditions. From the standpoint of security, OpenFlow's potential for new security schemes has not been well understood. When analyzing packets in a Packet message on a network controller, it follows that they can be used to identify and detect the penetration of the connection before the stream is configured. While new attack vectors were investigated for network security, the centralized network controller and control channel remain partially unprotected.

Keywords: information network, software-configured network, information security, SDN, Linux, Ethernet, OpenFlow

Березнюк А.В., Лазебний С.Г., Шевцов П.В., Гринкевич Г.О., Макаренко А.О. Аналіз програмного забезпечення необхідного для контролю за пакетами даних в програмно - конфігурованих мережах.

У статті представлено концепцію Software-Defined-Networking, а також історію програмованих мереж і огляд пов'язаних з ними новітніх стандартів. Розглянуті механізми безпеки для захисту архітектури SDN від супротивників. Розглянуто більш детально основні принципи OpenFlow.

Ключові слова: інформаційні мережі, програмно-конфігуровані мережі, інформаційна безпека, SDN, Linux, Ethernet, OpenFlow

Березнюк А.В., Лазебный С.Г., Шевцов П.В., Гринкевич А.А., Макаренко А.А. Анализ программного обеспечения необходимого для контроля за пакетами данных в программно - конфигурируемых сетях.

В статье представлена концепция Software-Defined-Networking, а также история программируемых сетей и обзор связанных с ними новых стандартов. Один из них, протокол OpenFlow, сыграл важную роль в продвижении концепции SDN, значительно повлиял на научные исследования и промышленности.

Ключевые слова: информационные сети, программно-конфигурируемые сети информационная безопасность, SDN, Linux, Ethernet, OpenFlow

Вступ

Постановка задачі. Техніка захисту може відновити OpenFlow мережі в межах 50 мс. Для цього ніяких дій від мережевого контролера не потрібно, так як перемикач може безпосередньо реагувати на невдачі. У техніці реставрації контролер повинен взаємодіяти з мережевими пристроями, що займає більше часу і робить метод менш зручним для великомасштабних мереж з великою кількістю підвузлів.

Аналіз останніх досліджень і публікацій. Сучасний стан формування методів аналізу та синтезу програмно-конфігурованих мережах нерозривно пов’язано з роботами таких вчених як В.Б. Толубко, Л.Н. Беркман, С.В. Козелков, Д.В. Агеєв, О.І. Лисенко, В.І. Новіков, М.М. Климаш, А.П. Бондарчук, O. Sheyner, P. Ammann, X. Ou, L. Wang, A. Рой, H. Poolsappasit.

© Березнюк А.В., Лазебний С.Г., Шевцов П.В., Гринкевич Г.О., Макаренко А.О., 2020

Мета даної статті полягає в огляді нині існуючих програмних засобів, щоб забезпечити безпеку та контроль трафіку в програмно-конфігурованих мережах.

Викладення основного матеріалу дослідження

Загальні положення. Для того, щоб виявити неробоче посилання своєчасно, протоколи управління можуть бути використані для моніторингу підключення. Наприклад, неактивний порт комутатора може бути виявлений втратою сигналу події (ЛОС) відмови. Виявлення зламаного шляху між двома вузлами може бути здійсненим за допомогою двонаправленого Forwarding Detection (BFD), котрий заснований на простому протоколі Hello, визначеного в RFC 5880. В якості альтернативи можна використовувати Link Layer Discovery Protocol (LLDP), але це викликає високе навантаження на мережевому контролері і обмежує масштабованість, оскільки такі повідомлення моніторингу повинні бути оброблені на високій частоті. В літературі [1] пропонується розширення OpenFlow специфікації, для розгортання децентралізованого моніторингу мережі, в тому числі генерації пакетів і обробки на мережевих пристроях. Вони демонструють свій підхід до несправності на основі технології MPLS і концентрують увагу на відновленнях при збоях в межах 50 мс.

Для того, щоб зменшити кількість таких повідомлень до мінімуму, автори [1] пропонують тільки інформування вимикачів про збої зв'язку. Необхідний алгоритм діє на мережевих пристроях, що дозволяє забезпечити менший час відновлення в порівнянні з контролером на основі схеми повідомлених катіонів. Як і в попередній концепції, вона вимагає наявності додаткових функцій, які будуть додані до комутатора OpenFlow.

Вимірювання і моніторинг. Моніторинг мережі інформує оператора про поточний стан мережі, а також є основою для алгоритмів виявлення несправностей. Це вимагає наявності відповідних датчиків в мережі, наприклад – комутаторів, які забезпечують статистичні дані про поток записів.

NetFuse призначений для виявлення і пом'якшення перевантаження, яке може виникнути з різних причин, наприклад, через розподілену відмову в обслуговуванні (DDoS) атаки або в результаті запланованого резервного копіювання. Для того щоб виявити таку раптову несправність, NetFuse встановлюється в якості додаткового шару між мережевими пристроями і мережевим контролером для обробки OpenFlow керуючих повідомлень (наприклад, PacketIn, FlowMod, FlowRemoved). Для того щоб виявити перевантаження, багатовимірний алгоритм агрегації використовується для виявлення підозрілих потоків. Надалі вони управляються адаптивним механізмом управління, який модифікує правила контролю на комутаторах, щоб краще справлятися з мережевим перевантаженням.

На основі OpenFlow керуючих повідомлень Packet-in і FlowRemoved, FlowSense для обчислення лінії зв'язку використовують програми, котрі виконуються на верхній частині мережі контролеру. Спрощується розгортання в порівнянні з підходами, які вимагають додаткового шару (наприклад, NetFuse). Оскільки вимірювання оновлюється тільки після отримання повідомлення FlowRemoved, в потік, який містить правило байдужих полів в поєднанні з тривалим терміном дії, призводить до сповільненого результату. Для активної політики установки потоку повідомлення Packet може відбутися будь-коли, а це означає, що активне опитування буде необхідне для виявлення стану.

Робота OpenSketch розглядається програмно, що складається з простої конструкції для площини даних, які можуть бути реалізовані на апаратному забезпеченні, в той час як функції аналізу даних розташовані на площині управління. Виходячи з ідеї ескізів, які є компактними структурами даних, більш гнучкі зштовхування потоку можна порівняти з OpenFlow специфікацією. У поєднанні з контролером OpenSketch для ескізу бібліотеки,

можуть бути розроблені нові алгоритми вимірювання, що дозволяє автоматизовану конфігурацію комутаторів відповідно до вимог додатка вимірювань.

Установка додаткових потоків записів для цілей моніторингу розглядається в схемі. Ця схема можлива тому, що OpenFlow специфікації підтримують кілька етапів потоку записів. Мережевий контролер може зчитувати відповідні лічильники періодично і може виявити важливий Hitters через ієрархічний алгоритм. Сила цього підходу полягає у використанні апаратного забезпечення і низьких витрат на комутатор.

Виявлення атак DDoS засноване на аналізі потоку статистики за допомогою нейронних мереж, за допомогою підходів, реалізований на платформі контролера NOx і включаючий в себе наступні кроки:

-> [Flow Collector] -> [Feature Extractor] -> [класифікатор] -> Alarm
| ---------- <----------|
LoopDetection

По-перше, потік статистики з одного або декількох комутаторів витягується через певні проміжки часу за допомогою потоку колектора. На другому етапі, функція ідентифікує відповідні функції C, які вказують на атаку DDoS. Наприклад, темпи зростання окремих притоків в одному напрямку є індикатором початку такого нападу. Кількість перемикачів, що беруть участь в моніторингу, може бути адаптована до нових топологій, але збільшення числа комутаторів створює додаткові накладні витрати в зв'язку зі збільшенням кількості керуючих повідомлень.

Пряме вимірювання лічильників, пов'язаних з входом, необхідно для того, щоб побудувати матриці, які містять обсяг між пунктами відправлення та призначення пар в мережі. Але запити збільшують навантаження на комутатор і це небажано в мережі, що складається з декількох перемикачів і великої кількості притоків. OpenTM описує стратегії визначення того, якому комутатору для запиту уздовж шляху потоку слід зменшити накладні витрати. Автори пропонують рівномірний розподіл для запитів потоку лічильників серед всіх комутаторів в мережі, включаючи інформацію про маршрутизації від мережевого контролера для цієї мети.

Можна описати їх, як P2P трафік, що може класифікуватися на основі особливостей мережевого рівня. Це означає, що питання конфіденційності пов'язані з глибокою інспекцією невідомих пакетів (DPI) і що не потрібно спеціалізованого апаратного забезпечення, так як аналіз може бути досягнутим на мережевому контролері. Після визначення відповідних функцій мережевого рівня, які можуть відрізнити P2P від не-P2P трафіку, вони можуть бути реалізовані в якості додатку для контролера NOX і оцінені для загальнодоступного набору даних.

Налаштування, перевірка і тестування. Виявлення мережевих несправностей є серйозною проблемою для кожного оператора і спирається на людський фактор. Для того щоб зменшити час виявлення несправності, необхідні нові методи для SDN архітектур. Для налагодження мереж і використовується підхід, який включає в себе як мережевий контролер (програмне забезпечення), так і мережеві пристрої (апаратне забезпечення), що потребують знаходження основної причини помилки. Формальна перевірка програмного забезпечення контролера може визначити, чи гарантуються певні властивості коректності (наприклад, припинення циклів) і чи забезпечується SOFT [2]. Тестування комп'ютерних мереж потрібно для того, щоб гарантувати, що всі мережеві компоненти працюють як і очікувалося. Вони мають включати тестовий сценарій, котрий є якомога ближчий до операціональних ситуації. Ця проблема вирішується за допомогою ATPG [3], який генерує

тестові пакети, тоді як OFRewind [4] забезпечує можливість для повторного відтворення захоплених сценаріїв.

OpenFlow дозволяє використовувати стандартне обладнання, яке потребує правильного виконання агентів на всіх мережевих пристроях. ONF має визначити випробування з серією тестів, що є особливо корисно для виробників задля уникнення помилок в програмному забезпеченні на етапі розробки нової моделі комутатора. Аналогічний підхід приймається OFTest [3], який є частиною проекту Project Floodlight і дозволяє розробляти платформи SPECI тестових сценаріїв.

SOFT [5] використовує тести на сумісність, щоб забезпечити правильну реалізацію на всіх комутаторах в мережі. Беручи до уваги символічне виконання, всі можливі шляхи в програмі (firmware) здійснюють і визначають поведінку кожного мережевого пристрою. Згодом, здійснюється перехресна перевірка між мережевими пристроями з використанням головного пристрою, що виявляє невідповідності між вхідними наборами. Це дозволяє мережевим операторам знаходити помилки реалізації перед розгортанням і гарантує безпомилкову роботу мережевих пристроїв.

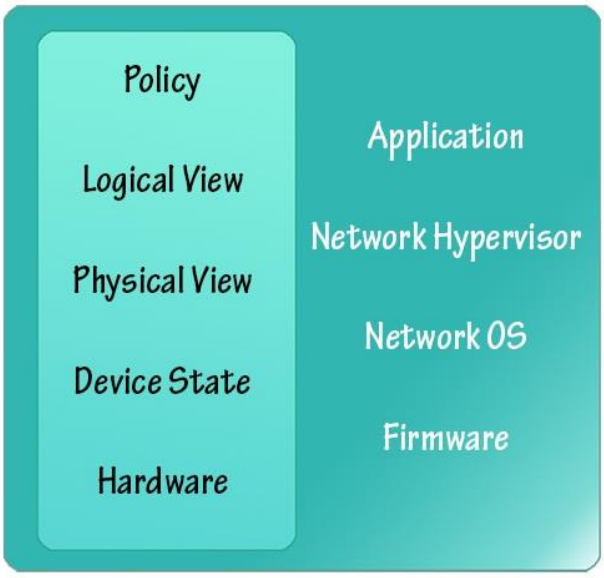
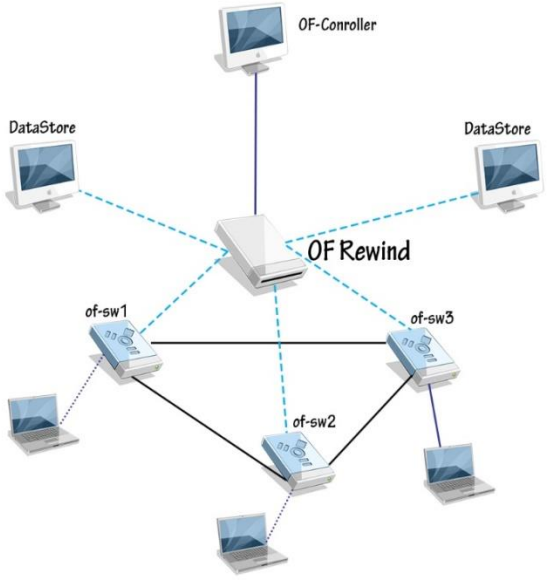


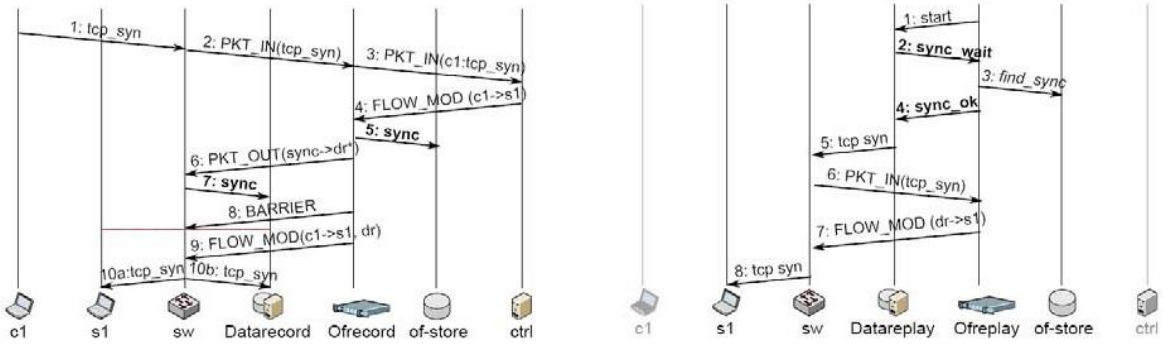
Рис. 1. Рівні стану та кодові рівні стеку SDN у разі безпомилкової роботи в мережі

Рівні стану можуть бути коректно відображені на будь-який інший рівень (еквівалентності). В іншому випадку, помилка може бути локалізована між рівнями, які відрізняються один від одного, і можуть бути виявлені в проміжному рівні коду.

Системний підхід до пошуку та усунення несправностей SDN наводиться в [2], в якому пропонується потік, який відокремлює стек SDN в рівні стану і кодових рівнях, як показано в таблиці 1. Рівні стану являють собою мережу конфігурації, в той час як рівні стеку описують відображення між двома рівнями стану. Методика дозволяє автоматичне визначення кодового рівня, де знаходиться несправність. Нормальна поведінка мережі призводить до еквівалентності між усіма рівнями стану таким чином, що кожен рівень може бути відображений на будь-який інший. У разі несправності мережі, помилка може бути локалізована в код рівня, розташованого між рівнями стану, які не є еквівалентними. Після того, як вона була розміщена, її причина може бути визначена за допомогою методів, які обговорюються раніше.



а) компоненти системи



б)Ofrecord c) Ofreplay

Рис. 2. Системні компоненти OFRewind: (а). Запит TCP SYN відправляється від хоста c1 до S1 може бути записаний за допомогою Ofrecord (б) і знову прочитаний за допомогою Ofreplay механізму (с) де запит (5: TCP SYN) відправляється з системи зберігання Datareplay від c1

Мережевий відладчик ndb [5] допомагає оператору визначити причину збоїв програмного забезпечення (або помилки) в мережі. Кожен комутатор посилає повідомлення, які спрацьовують, коли пакет з інформацію про співпадіння потоку вводу поступає на комутатор. Вони збираються з усіх перемикачів в централізований колектор, який може бути використаний для визначення пакетів, що мають відношення до пакету точки зупинки. Наприклад, для того, щоб досліджувати помилку досяжності для пакета посланого від хоста A до хоста B, наступний запит може бути використаний:

OFRewind [3] забезпечує запис і відтворення налагоджених засобів для SDN. Він вставляється в якості проксі-сервера між мережевим контролером і мережевими пристроями, а також дозволяє перехоплення і видозміну повідомлень управління для запису або відтворення фабричного калібрування. Для того, щоб зберігати трафік користувачів сховища даних, керовані компонентом OFRewind підключаються до комутаторів. Різні режими,

наприклад, Replay дозволяють різні тестові сценарії, наприклад, щоб отримати тільки керуючі повідомлення.

VeriFlow [3] забезпечує більш сфокусований вид на площині даних VERI. Це додає додатковий шар між мережевим контролером і мережевими пристроями, щоб перевірити нові правила в режимі реального часу перш ніж вони будуть розгорнуті в мережі. Правила мережі розділені на безліч класів еквівалентності (ЕКС), де подібні дії переадресації розташовані в ЕКС. Вони зберігаються в деревовидних структурах даних, котрі описують пересилання графіків пакетів в мережі. Нові правила аналізують VeriFlow шляхом виконання запитів на основі інваріантів для того, щоб виявити їх порушення. Такі інваріанти можуть бути встановлені через API і охоплювати широкий спектр умов, наприклад, досяжності, петлі-вільності і консистенції.

Anteater [6] також фокусується на площині даних аналізу. Вона являє собою зібрану топологію мережі і пересилання інформаційних баз (FIBs) з мережевих пристроїв в якості логічних функцій. Використовуючи цей метод, автори змогли виявити 23 помилки мережі. Перевага такого механізму в тому, що реальна поведінка системи може бути проаналізована без необхідності протоколів маршрутизації, внаслідок чого процедура набуває спрощення. Крім того, на відміну від площини управління на основі аналізу, цей підхід дозволяє виявляти помилки програмного забезпечення маршрутизатора.

Тема просторового аналізу (HSA) [4] зосереджується на виявленні помилок, таких як збої досяжності, циклів пересилання, трафіку з ізоляцією, проблеми витоку. Кожен заголовок пакета представлений як точка в просторі, в той час як мережеві пристрої і порти моделюються в ньому. Коли пакет перетинає мережу, вона трансформується з однієї ділянки до іншої. Поділ мережевого трафіку здійснюється через стрибок, що представлено як підмножина простору. Мережеві пристрої, такі як комутатори і маршрутизатори, додатково можуть бути змодельовані за допомогою відповідних функцій передачі даних. У поєднанні з заголовком простору це дозволяє виявляти порушення відповідних відмов. У той час як така структура здатна локалізувати джерело помилки (наприклад, непослідовні таблиці маршрутизації), він не може визначити причину, по якій сталася помилка.

Випробування нових застосувань контролера потрібно OpenFlow для належної конфігурацією контролера і перемикачів. Оскільки лише невелика кількість людей має доступ до таких ресурсів, системи зі складовими з віртуальними машинами (VM) є привабливою альтернативою. Через величезну витрату пам'яті, яка необхідна для створення великої мережі, масштабованість поширюється лише на кілька перемикачів. Більш легкий тип віртуалізації можливий з Mininet [5] мережевим емулятором, який дозволяє використовувати мер. Альтернативу для тестування на стандартному ноутбуці. Для створення нової мережі наступна команда дає приклад тесту на віртуальній мережі з топологією дерева глибини 2, розгалуження 8 і мережевий контролер на основі NOX:

mn --switch ovsk --controller nox --topo tree, depth=2,\\ fanout=8 --test pingAll

Всі номери, хости, комутатори і контролери емулюються і команди оболонки дозволяють використовувати основні засоби мережі (наприклад, пінг). Mininet надає можливість розробникам виконувати і тестувати нові програми контролера в різних мережевих топологіях. Обмеження цього підходу виникає з того, що продуктивність віртуалізації машини зазнає впливу високих навантажень, від швидкості передачі пакетів, складності O(n) в порівнянні з O (1) для таблиці пошуку в апаратному перемикачі на основі троїчної асоціативної пам'яті (TCAM). Завдяки своєму широкому використанню Mininet є найбільш поширеним емулятором для SDN та OpenFlow.

Мережа відкладки необхідна і для того, щоб виявити помилки, коли збої програмного забезпечення не є очевидними. Метод [6] може бути використаний для тестування програм контролера задля виявлення порушень властивостей коректності, викликаних помилками в програмному забезпеченні контролера. Перевірка моделі використовується для опису топології мережі і досліджує простір станів, який включає в себе контролер, перемикачі та господарів. Для того, щоб зменшити розмір вхідного простору, обробки подій на контролері виконуються автоматично символічним двигуном, який генерує тестові входи і впускає пакети в мережу. Метод прототип був оцінений на трьох реальних OpenFlow додатках, які були написані в Python для контролера NOX. Типовим прикладом є помилка в застосуванні комутатора MAC-навчання, який повинен посилати пакети конкретного пристрою, навіть якщо цей пристрій переміщається в нове місце в мережі. Якщо жорсткий тайм-аут нотифікації пропущений, то всі пакети раніше доставляються на старе місце і відповідні записи потоку не оновлюються з новими параметрами визначення місця розташування. Інструмент дозволяє виявити такі конструкції помилок реалізації, що є загальнодоступними.

Безпека. Архітектура SDN дозволяє реалізувати нові концепції безпеки, котрі неможливо було впровадити раніше. Наприклад, кожний мережевий пристрій може бути налаштований так, щоб блокувати пакети аналогічним чином до мережі Брандмауер. Хоча для виявлення вторгнень традиційно потрібні дорогі апаратні рішення. В роботі [7] показано, як алгоритми виявлення аномалій можуть бути адаптовані до мереж OpenFlow на основі реалізації в мережевому контролері. З іншого боку, архітектура SDN забезпечує новий простір для діяльності зловмисників і вимагає адекватних механізмів захисту. Наприклад, FortNOX [8] забезпечує механізм безпеки, який захищає потік установки механізму від супротивників.

Робота [9] розглядає виявлення аномалій OpenFlow і стверджує, що він повинен бути переміщений від ядра до домашньої мережі (близько до користувача), щоб отримати кращі результати виявлення. Автори адаптували чотири існуючі алгоритми для використання з OpenFlow, в тому числі виявлення скануючих інфекцій на хостах, що лімітує швидкість в умовах інфекції; виявлення аномалій з використанням максимальної ентропії і детектора аномального значення. Оскільки навантаження по обробці поширюється на домашніх користувачів, такий підхід знижує вимоги до обробки на постачальника послуг Інтернету (ISP) і знижує витрати. Крім того, розгортання алгоритмів виявлення на мережевому контролері не впливає на продуктивність пересилання пакетів на площині даних.

FRESCO [10] структура дозволяє розгортати служби безпеки для розімкнутої Flow. Вона включає в себе мову сценаріїв, що дозволяє досягнути розвитку служб безпеки, заснованих на API і бібліотек, що складаються з 16 багаторазових модулів. Сама структура FRESCO реалізована як додаток, побудований на контролері NOX.

Захист від атак IP-сканування описуєтьсяя в OpenFlow у випадковому вузлі комутації. [2]. Це засновано на ідеї, що статичні IP-адреси є легкою мішенню для зловмисників, але їх можна уникнути за допомогою проактивних методів, які змінюють IP-адреси господарів з плином часу (рухома мішень оборони (MTD)). У разі розімкнутого потоку, реальна IP-адреса зберігається господарем, але замінюється віртуальною IP-адресою, яка часто перепризначувалася до мережевих пристроїв за допомогою мережевого контролера. Для цього потрібен механізм трансляції адрес, а також гарантії обмежень, таких як мутації непередбачуваності. Підхід оцінювали з використанням MiniNet проти зовнішнього мережевого сканера (NMAP15) і атаки хробака.

FortNOX [4], нове виконання ядра політики безпеки підвищує захист потоку під час процедури налаштування в OpenFlow. Це може бути використаним суперником для того, щоб взяти під своє керування мережу. FortNOX вимагає цифрового підпису для авторизації.

Залежно від застосування, він визначає рівень пріоритету потоку з правилом в потік таблиці. Крім того, FortNOX виявляє, коли потік може обійти політику безпеки. Це можливо не тільки в разі перекриття діапазонів IP, але і, якщо правило встановлює новий заголовок пакета і пакет ред модіфіка – може досягти пункту призначення. В іншому випадку відбувається блокування.

Висновки

В даний час дослідження в області OpenFlow на основі SDN вже допомогли вирішити деякі питання до стадії оперативного розгортання самого проекту. Як ми вже казали, управління несправностями концентрується в основному на виявленні та виправленні переривання зв'язку, які можуть пошкодити не тільки дані, а й зв'язок з контролером. Для того, щоб запропонувати нові схеми виявлення несправностей для SDN, відповідні методи моніторингу для цієї архітектури повинні бути вивчені. Існує декілька підходів до вимірювання та моніторингу мережі, котрі грунтуються на вставці додаткових шарів між пристроями і контролером мережі. Це дозволяє забезпечити перевірку керуючих повідомлень OpenFlow, наприклад, щоб ідентифікувати трафік перевантаження. Так як додатковий шар відрізняється від оригінальної архітектури OpenFlow і вимагає додаткових апаратних засобів, аналізу потоку записів на мережі контролера – це може бути досягнуто шляхом розгортання алгоритмів моніторингу, що працюють на основі мережевого контролера. Аналізи потоку можуть бути використані для виявлення характеристик трафіку і побудови повної картини мережі шляхом включення вхідних сигналів від всіх мережевих пристроїв.

Для налагодження програмного забезпечення необхідно забезпечити контроль за пакетами в мережі, хоч формально перевірка може бути використана для визначення правильності потоку. Це необхідно для того, щоб генерувати докладні мережеві знімки, а також інспектувати параметри функції при певних умовах мережі. З точки зору безпеки, потенціал OpenFlow реалізованих на нових схемах безпеки не були добре вивчені. Аналізуючи пакети в повідомленні Packet на контролері мережі випливає, що вони можуть бути використані для ідентифікації і визначення проникнення зв'язку, перш ніж потік налаштується. У той час як для безпеки мережі були досліджені нові вектори атак, централізований контролер мережі і канал управління залишаються частково незахищеними.

Список використаної літератури

1. Metaswitch Networks, “PCE – an evolutionary approach to SDN,” [Електронний ресурс] // – Режим доступу: http://www.metaswitch.com/sites/default/files/metaswitch-white-paper-pce-an-evolutionary-approach-to-sdn.pdf

2. Смелянский Р.Л. Программно-конфигурируемые сети. Открытые системы. СУБД 9. 2012. с. 23–26.

3. Захаров А.А. Аспекты информационной безопасности архитектуры SDN [Електронний ресурс] / А. А. Захаров, Е. Ф. Попов, М. М. Фучко // – Режим доступу : http://vestnik.sibsutis.ru/uploads/1459328716_7845.pdf

4. Климаш М.М Розробка методу балансування навантаження в sdn мережах на основі модифікованого протоколу STP [Електронний ресурс] / М. М. Климаш., М.І. Бешлей., Ю. Л. Дещинський., О.М. Панченко // Комп’ютерні технології друкарства. – 2015. С.146-155 - Режим доступу : http://ctp.uad.lviv.ua/images//ktd/34_klymach.pdf

5. Шалимов А.В. Технологии SDN/OpenFlow [Електронний ресурс] / Шалимов А.В // – Режим доступу : http://lvk.cs.msu.su/~sveta/SDN_OpenFlow_basics_lecture1.pdf

6. Casado M. "Fabric: a retrospective on evolving SDN,". / M. Casado, T. Koponen, S. Shenker, and A. Tootoonchian. // in Proceedings of the First Workshop on Hot Topics in Software Defined Networks (HotSDN'12). - New York, NY, USA: ACM. – 2012. - pp. 85–90.

7. Hoelzle U. "OpenFlow @ Google," [Електронний ресурс] / U. Hoelzle. // – Режим доступу : http://opennetsummit.org/archives/apr12/hoelzle-tue-openflow.pdf (2012.)

8. HP Networking, "Software defined networks (SDN)." [Електронний ресурс] // – Режим доступу: http://h17007.www1.hp.com/us/en/mobile/solutions/tech/sdn.html

9. Juniper Networks, "OpenFlow Switch Application (OF-APP) for Juniper MXSeries Routers." [Електронний ресурс] // – Режим доступу: https://developer.juniper.net/ shared/jdn/docs/ProgrammableNetworks/OpenFLow APP JDN Overview.pdf

10. NOX: towards an operating system for networks [Електронний ресурс] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker // – Режим доступу : http://www.cs.yale.edu/homes/jf/nox.pdf

Автори статті

Березнюк Андрій Володимирович - аспірант, Державний університет телекомунікацій, Київ, Україна.

Лазебний Сергій Геннадійович - аспірант, Державний університет телекомунікацій, Київ, Україна.

Шевцов Павло Володимирович - аспірант, Державний університет телекомунікацій, Київ, Україна.

Гринкевич Ганна Олександрівна – кандидат технічних наук, доцент, доцент кафедри телекомунікаційних систем та мереж, Державний університет телекомунікацій, Київ, Україна.

Макаренко Анатолій Олександрович – доктор технічних наук, доцент, професор кафедри Мобільних та відеоінформаційних технологій, Державний університет телекомунікацій, Київ, Україна.

Authors of the article

Bereznyuk Andriy Volodymyrovych - postgraduate, State University of Telecommunications, Kyiv, Ukraine.

Lazebnyi Serhii Hennadiiovych - postgraduate, State University of Telecommunications, Kyiv, Ukraine.

Shevtsov Pavlo Volodymyrovych. - postgraduate, State University of Telecommunications, Kyiv, Ukraine.

Grynkevych Ganna Aleksandrovna – candidate of Science (technic), assistant professor, assistant professor of Department of Telecommunication systems and networks, State University of Telecommunications, Kyiv, Ukraine.

Makarenko Anatoliy Oleksandrovych – doctor of Science (technic), associate professor, professor of Department of Mobile and videoinformation technologies, State University of Telecommunications, Kyiv, Ukraine.

Дата надходження в редакцію 02.12.2020 р. Рецензент: д.т.н., професор В.Ф. Заїка