Educational Matters
Научно-методически статии

УПРАВЛЕНИЕ НА СОФТУЕРНО ДЕФИНИРАНИТЕ МРЕЖИ С FLOODLIGHT CONTROLLER

Румен Тодоров
Софийски университет „Св. Климент Охридски“

Резюме. Основната идея при софтуерно дефинираните мрежи SDN (Software Defined Networking) е разделянето на системите, които управляват мрежата (control plane), от устройствата, които предават данните (data plane). По този начин се дефинира слой за програмно управление и контрол на цялата мрежа (control layer). SDN контролерът управлява цялата мрежа, получава указания и инструкции от приложния (application layer) слой и ги препредава към компонентите на мрежата. Контролерът извлича информация за мрежата от мрежови устройства – инфраструктурен слой (infrastructure layer) – и я връща обратно към приложния слой. В тази статия ще представим основната архитектура и характеристиките на Floodlight контролера. Тестваме различни топологии на мрежовия симулатор Mininet, взаимодействие между виртуални машини, добавяне и прихващане на OpenFlow потоци от мрежовия анализатор Wireshark.

Keywords: Software Defined Networking; Floodlight Controller; OpenFlow

1. Въведение

Floodlight е контролер с отворен код, който съдържа съвкупност от независими модули, написани на Java. Архитектурата му се основава на контролера Becon, създаден от David Erickson[1)] от Станфордския университет през 2010 година. Чрез контролера Floodlight могат да се управляват физически и виртуални комутатори, маршрутизатори, различни точки на достъпи др. Floodlight взаимодейства с мрежовите устройства през Southbound API слоя, като използва OpenFlow протокол (протокол за предаване на потоци от данни в софтуерно дефинираните мрежи)[2)]. Floodlight контролерът открива и поддържа топологията на мрежата, управлява движението на потоците, изчислява най-кратките пътища между възлите в мрежата, добавя нови правила за управлението в комутаторите, поддържа статистическата информация за състоянието на мрежата. Той взаимодейства с външни приложения, като използва Northbound API

слой. Модулната му архитектура улеснява добавянето на нови модули и промени във вече съществуващите модули. Разпространява се под Apache лиценз[3)].

2. Архитектура на Floodlight контролера

Архитектурата на Floodlight се състои от **пет** основни компонента[4)]:

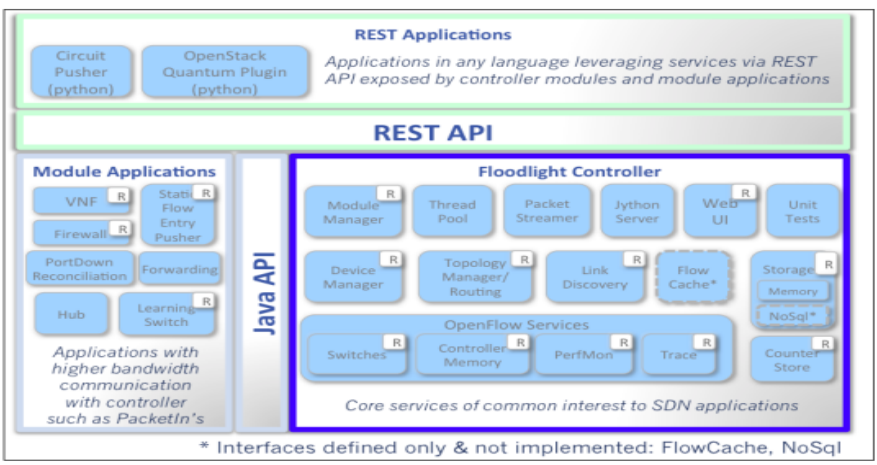
– модули на контролера (**controller modules**);

– модулни приложения (**module applications**);

– приложения, написани на различни езици, които използат **REST** (Representational State Transfer) **Applications** за обмен на данни;

– приложения за дистанционно управление на мрежовите устройства, които използват **REST API**;

– приложения, написани на езика Java, които използват API (Application Programming Interface) – **Java API**.



Source:https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/pages/1343548/The+Controller

Фигура 1. Обща архитектура на Floodlight контролера

2.1. Модулите на контролера (controller modules)

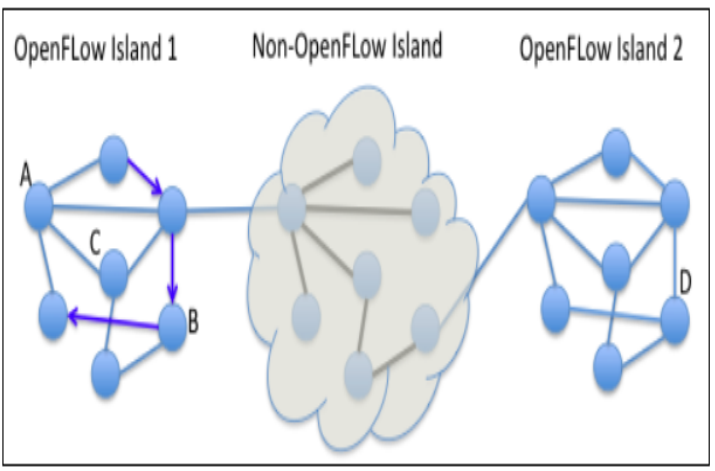
Контролерът съдържа основните модули. Тези модули трябва да приемат пристигащите пакети, да инсталират необходимите правила за разпределение и управление на потоците и след това да ги предават на комутаторите.

Кратко представяне на модулите.

– **LinkDiscoveryManager** – открива, управлява и поддържа връзките между комутаторите в OpenFlow мрежата, като използва LLDP (Link Layer

Discovery Protocols) и BDDPs (Broadcast Domain Discovery Protocol) за откриване на връзките.

– **TopologyService** – поддържа информация за топологията на мрежата, за откриване на свързаните суичове и проследяване на връзките. OpenFlow островите могат да взаимодействат с участъци от мрежата, които не поддържат OpenFlow протокола, но връзката се осъществява по един канал, управлявана от един контролер.



source: https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/pages/1343517/Supported+Topologies

Фигура 2. Топология, поддържана от Floodlight

– **RestApiServer** – сървър, който дава възможност на REST API (REpresentational State Transfer Application Programming Interface) приложенията да осъществяват заявки и обмен на данни, като използват HTTP (Hypertext Transfer Protocol) протокола и стандартни TCP портове.
– **ThreadPool** – Java модул, който дава възможност на някои нишки да се изпълняват по определен план със забавяне или да се изпълняват периодично.
– **The MemoryStorageSource** – дава възможност за съхранение на данните в табличен вид (NoSQL, Cassandra-based database).
– **High Availability Support** – осигурява надеждно споделяне и обмен на информация между няколко контролера.
– **Packetstreamer** – услуга, която може да формира потоци от OpenFlow пакети по определени критерии, които да бъдат обменяни между комутаторите и контролера.

2.2. Модулни приложения (module applications)

Module applications (модулни приложения) съдържат модули, създадени на Java, компилирани и след това включени в контролера Floodlight[5].

Представям основните модули:

– Virtual Network Filter (VNF) – изолиране на мрежата на базата на MAC адресите (layer 2);

– Static Flow Entry Pusher – модул, който дава възможност на потребителя самостоятелно да добавя потоци;

– Firewall – защитната стена също е създадена като модул; проверява реактивно входящите потоци;

– Port Down Reconciliation – модул, който се използва за съгласуване на потоците;

– Forwarding – осъществява препращането (пренасочването) на пакети от едно устройство към друго;

– **Learning Switch** – модул, който изпълнява функциите на суич (комутатор). Обикновен L2 суич. Препраща пакетите на съответните портове въз основа на MAC адресите, включени в пристигащите пакети;

– **Access Control List (ACL)** – прилага правилата за контрол на достъпа върху входящите потоци по проактивен начин.

2.3. REST Applications

Приложения, които могат да бъдат написани на различни езици за програмиране. Използват REST API за разработка на приложения. Могат да се създават приложения за управление, за следене състоянието на мрежата и др. Приложенията дават възможност на администраторите да контролират цялата мрежа от едно-единствено устройство. В настоящия момент REST Applications съдържа две приложения.

– **Circuit Pusher** – приложение, което създава двупосочна връзка между две IP устройства с определен приоритет чрез добавяне на записи във всички комутатори, които съдържат тази връзка.

– **Open Stack** – Floodlight може да бъде използван от OpenStack, като се използва плъгингът Neutron[6]. Neutron е част от проект на OpenStack, който предоставя NaaS ("networking as a service") – предоставя мрежови ресурси, услуги и приложения като продукт между различни интерфейсни устройства[7]. Neutron използва създадения от Floodlight REST API за предоставяне на тази услуга (NaaS)[8].

2.4. REST API

Приложения за дистанционно управление на мрежовите устройства, които използат **REST API**. Последните години за дистанционно управление и конфигуриране на мрежовите устройства на разработчиците се

препоръчва използването на REST API (REpresentational State Transfer Application Programming Interface)[9]. REST дава възможност на API приложения да осъществяват връзка и обмен на данни с външни потребители в мрежата. Използват стандартни TCP сокети и HTTP протоколи за изпращане и получаване на съобщения в реално време, без да са необходими специални конфигурационни настройки. REST е разработен така, че да позволява на външни приложения да извличат статистическа информация от мрежата, да изчисляват маршрут на едно устройство от друго, да инсталират правила в защитната стена, да откриват свързаните комутатори, устройства/хостове в определена топология от мрежата. По подразбиране API е достъпен на порт 8080[10]. По желание номерът на порта може да се промени или да използва HTTPS за сигурна (криптирана) връзка. Floodlight предоставя възможността за създаване на собствено REST API в зависимост от конкретните нужди на потребителя[11].

2.5. Java API

Класовете, включени в Java API, улесняват създаването на нови модули, написани на Java, които да бъдат включени като част от Floodlight контролера[12]. Като за развойна среда може да се използва **Eclipse**.

3. Експериментална част

3.1. Инсталиране на Floodlight контролер

Floodlight може да се изтегли от Github и да се създаде чрез Ant[13]java базиран инструмент за изграждане на Java проекти.

Отваряме терминален прозорец и пишем[14]:

```
git clone git://github.com/floodlight/floodlight.git
cd floodlight
git submodule init
git submodule update
ant
sudo mkdir /var/lib/floodlight
sudo chmod 777 /var/lib/floodlight
./floodlight.sh
```

Би трябвало при успешно инсталиране да ни изпише:

```
BUILD SUCCESSFUL
Total time: 37 seconds
```

3.2. Тестване на мрежовия симулатор Mininet

Mininet е софтуерен емулатор за създаване на виртуални мрежи[15]. Могат да бъдат симулирани мрежови топологии с големи размери, след това да бъдат прехвърляни на реални хардуерни устройства с минимални из-

менения. Mininet поддържа протокола OpenFlow и софтуерния комутатор OpenvSwitch.

sudo mn -c – команда за инициализиране (изчистване) на първоначалните настройки.

– Тестваме с минимална топология на две виртуални машини (VM1, VM2) на Oracle VM VirtualBox операционна система Ubuntu 14 на виртуалните машини. На първата *VM1 с мрежови настройки IP adress: 10.0.2.15* Primary DNS: 10.0.2.3 стартираме Mininet:

sudo mn --controller=remote,ip=192.168.56.101,port=6653 --switch ovsk,protocols=OpenFlow13

Стартираме топология от два хоста (h1, h2) и един суич (s1), управлявани от отдалечен контролер (c0) с IP адрес: 192.168.56.101. Поддържа OpenFlow 1.3 и комутатор Open vSwitch (ovsk).

Втората VM2 е с мрежови насройки: Network Host-only Adapter vboxnet0 с установен IP адрес: 192.168.56.101. На тази виртуална машина стартираме контролера Floodlight.

Влизаме в директорията, в която е инсталиран:

cd floodlight

Стартираме контролера:

java -jar ./target/floodlight.jar

Тестване на връзката

mininet>**h1 ping h2** – проверява за връзка с командата ping

mininet>**nodes** – показва възловите точки

mininet>**net** – показва съществуващите връзки

mininet>**dump** – дава подробна информация за съществуващите връзки

mininet>**exit** – изход

– Ако извършваме теста на локален хост (localhost), преди да стартираме, на Floodlight контролер трябва да стартираме Open vSwitch (OVS) комутатора. Open vSwitch съдържа малък сървър за база данни, който трябва да се конфигурирал.

Стартираме първи терминален прозорец и пишем:

sudo ovsdb-server -v --remote=punix:/usr/local/var/run/openvswitch/db.sock --remote=db:Open_vSwitch,Open_vSwitch,manager_options --pidfile --detach --log-file

Инициализиране на OVS:

ovs-vsctl --no-wait init

ovs-vswitchd --pidfile --detach

Стартиране на OpenvSwitch:

/etc/init.d/openvswitch-switch start

Би трябвало да ни изпише:

[ok] Starting openvswitch-switch (via systemctl): openvswitch-switch.service.

След това стартираме Mininet със следната топология:

sudo mn --arp --topo single,3 --mac --switch ovsk --controller=remote,ip=127.0.0.1,port=6653

Топология съдържа три хоста (h1,h2,h3) и един суич (s1), управлявани от отдалечен контролер (c0) с IP адрес:127.0.0.1. Портът по подразбиране на Floodlight контролера е 6653. Опцията --mac показва, че спрямо поредния номер на хоста се присвоява мас адресът. Опцията --arp означава, че Mininet може да попълва статични ARP записи в таблици с Erthernet адресите, отговарящи на заявките[16], които могат да бъдат използвани за бъдеща употреба.

Ако в този момент проверим връзката с командата ping:

mininet>**h1 ping h2** – проверява за връзка с командата ping, ще ни изведе съобщението: Destination Host Unreachable.

Трябва да стартираме контролера, който да създаде и след това да управлява топологията на мрежата.

Отваряме втори терминален прозорец и стартиране на Floodlight. Ако сте root потребител, влизате в терминалния прозорец:

cd..

След това в директорията, в която е инсталиран: **cd floodlight**

Стартираме контролера по следния начин: **java -jar ./target/floodlight.jar**

При проверка с командата ping би трябвало да има достижимост до хост:

mininet> **h1 ping h2**

Добавяме потоци в mininet:

sh ovs-ofctl add-flow s1 in_port=2,actions=output:1

sh ovs-ofctl add-flow s1 in_port=1,actions=output:2

sh ovs-ofctl add-flow s1

priority=1111,in_port=1,actions=output:3

sh ovs-ofctl dump-flows s1 – показва таблицата на потоците на суич s1;

ovs-ofctl show s1 – разширено представяне на таблицата;

sh ovs-ofctl del-flows s1 – изтрива потоци;

3.3 Предоставя web базиран интерфейс

Floodlight контролерът предоставя удобен графичен интерфейс (GUI), който ни дава информация за състоянието на контролера, топо-

логията на мрежата, съдържанието на таблиците с потоците и др. Графичният интерфейс е достъпен, след като напишем в браузър следния URL адрес:

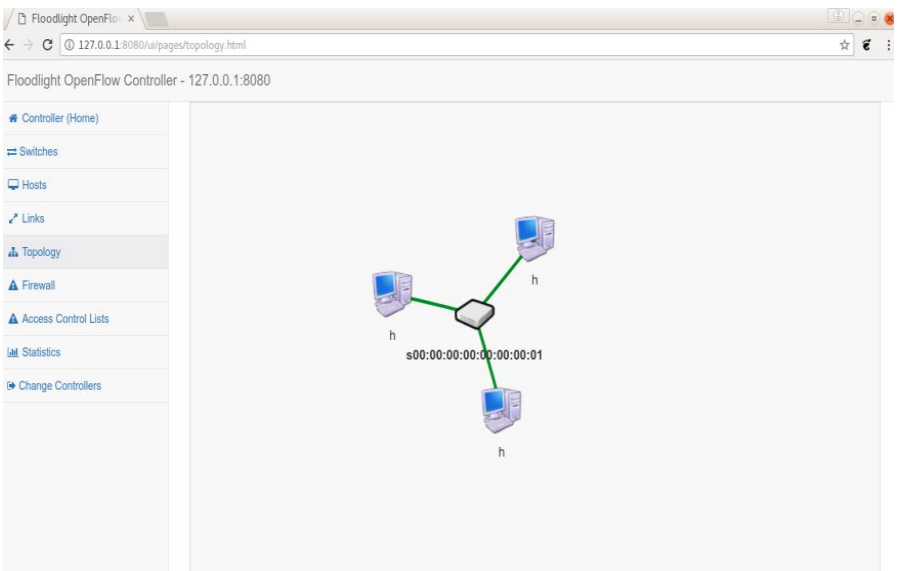
http://<controller-ip>:8080/ui/index.html

където <controller-ip> е IP адресът или името на хоста на Floodlight контролера, ако контролерът е стартиран на същия компютър, на който работим. За достъп до web GUI трябва да изпишем:

http://localhost:8080/ui/index.html

или

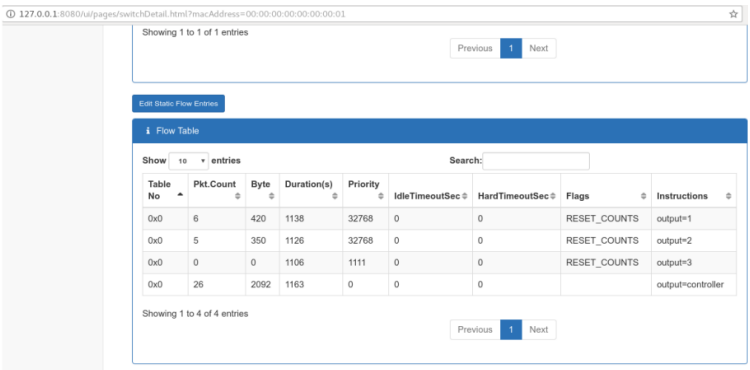
http://127.0.0.1:8080/ui/index.html



Фигура 3. Представяне на топология от web интерфейса

Можем да получим информация на въведени потоци във flow таблицата на комутатор s1 от предишната глава и в графичен интерфейс на адрес:

http://127.0.0.1:8080/ui/pages/switchDetail.html?macAddress=00:00:00:00:00:00:00:01



Фигура 4. Таблицата с потоците от web интерфейс

4. Мониторинг на мрежата с Wireshark

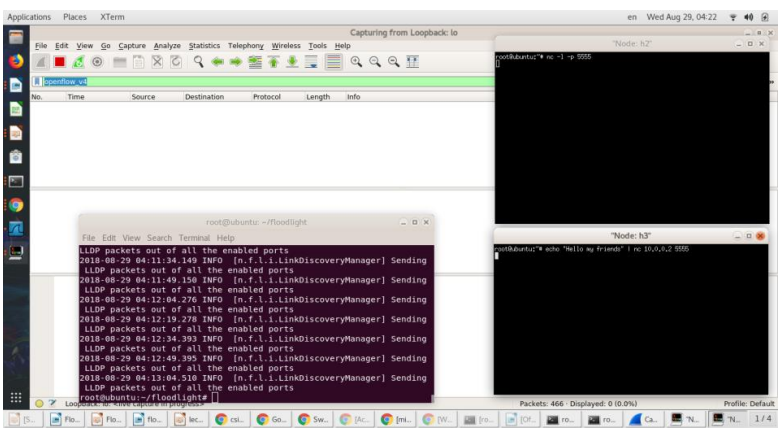
Използваме програмата Wireshark за прихващане на OpenFlow пакети в създадената топология. Стартираме Wireshark в нов терминален прозорец:

sudo wireshark &

От програмата избираме да осъществява мониторинг на локалния хост: loopback:lo

В полето “Filter“: можем да осъществим филтриране по различни критерии. Изписваме:

openflow_v4 – по този начин задаваме критерии за прихващане OpenFlow v1.3 пакети.



Фигура 5. Изображение преди стартиране на контролера Floodlight

Стартираме Mininet със следната топология:

sudo mn --arp --topo single,3 --mac --switch ovsk --controller=remote,ip=127.0.0.1,port=6653

Използваме многофункционалния инструмент Netcat (Shema&Johnson, 2004) за предаване на съобщението “Hello my friends” от хост h3[17]: 10.0.0.3 към хост h2: 10.0.0.2.

В mininet отваряме терминален прозорец на h3:

xterm h3

Задаваме статичен arp адрес 10.0.0.11:

h3 > arp -s 10.0.0.11 00:00:00:00:00:11

Осъществяваме проверка с командата:

h3 arp

Отваряме терминален прозорец на h2:

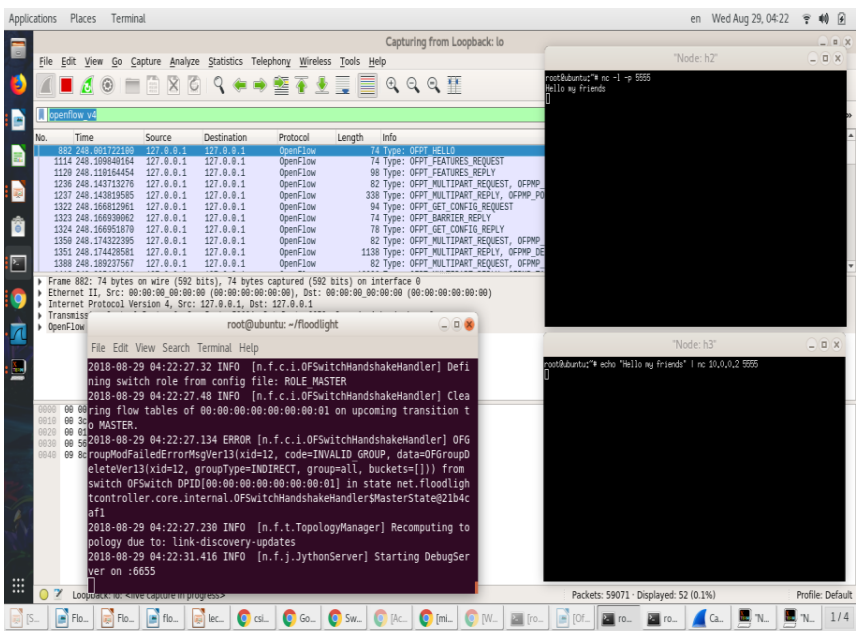
xterm h2

Стартираме netcat на h2 в режим слушане на порт 5555:

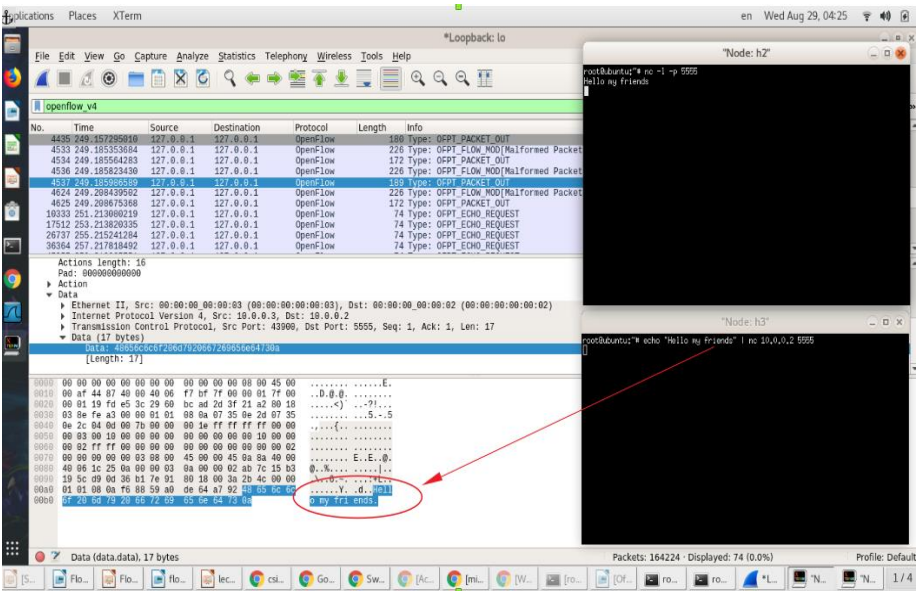
mininet h2> nc -l -p 5555

От терминален прозорец h3 изпращаме съобщението:

mininet h3> echo “Hello my friends” | nc 10.0.0.2 5555



Фигура 6. Изображение след стартиране на контролера Floodlight



Фигура 7. Изображение след прихващане на низа "**Hello my friends**" от Wireshark

Заключение

Floodlight контролерът е удобен за изследователски цели, лесен за инсталиране и употреба. Поддържа се от опен сорс разработчиците, както и от Big Switch Networks. Модулната архитектура му дава възможност за лесно добавяне и редактиране на модули. Предоставя web базиран интерфейс, който дава информация за топологията на мрежата, състоянието на контролера, свързаните хостове, суичове, таблиците на потоците и т.н. Тестването от много разработчици дава възможност за добавяне на нови модули (услуги), повишаване на надеждността. Автоматизирано добавяне на нови потребители. Достъп за управление на SDN от различни устройства и различни точки в мрежата.

NOTES/БЕЛЕЖКИ

1. David Erickson, "The Beacon OpenFlow Controller", Hong Kong, China – August, 2013.

http://yuba.stanford.edu/~derickso/docs/hotsdn15-erickson.pdf (last visit 19.09.2018).

2. Софтуерно дефинирани мрежи: SDN [Software-Defined Networking: SDN]. http://www.programist.bg/virtual/v2.html (last visit 19.09.2018).

3. SDN Series Part Five: Floodlight, an OpenFlow Controller, 6 Jan 2015. https://thenewstack.io/sdn-series-part-v-loodlight/ (last visit 19.09.2018).
4. Eduardo Berrueta, "Testing tool of SDN controllers' performance", June 2016. http://academica-.unavarra.es/bitstream/handle/2454/22442/TFE_BerruetaIrigoyen.pdf?sequence=1 (last visit 19.09.2018).
5. Module Applications, https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/pages/1343509/Module+Applications (last visit 19.09.2018).
6. OpenStack, https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/pages/1343489/OpenStack (last visit 19.09.2018).
7. Neutron-OpenStack Networking, https://wiki.openstack.org/wiki/Neutron (last visit 19.09.2018).
8. Antti Tolonen, Miika Komu, Software-defined Networking, 30.10.2013, http://www.cse.tkk.fi/fi/opinnot/T-110.5121/2013/luennot-files/SDN_lecture.pdf (last visit 19.09.2018).
9. Hakan Akcay, Derya Yiltas-Kaplan, "Web-Based User Interface for the Floodlight SDN Controller", March 2017, http://www.ijana.in/papers/V8I5-2.pdf (last visit 19.09.2018)
10. Floodlight REST API, https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/pages/1343539/Floodlight+REST+API (last visit 19.09.2018).
11. How to add a REST API to a Module, https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/pages/15040589/How+to+add+a+REST+API+to+a+Module (last visit 19.09.2018).
12. Just what is the Java API anyway? https://www.javaworld.com/article/2077392/java-se/just-what-is-the-java-api-anyway.html (last visit 19.09.2018).
13. Diarmuid O'Briain, "Building a Network Training Emulator (NTE) Software Defined Networking (SDN) Virtual Machine (VM)", Feb 2017, http://www.obriain.com/training/TEL3214/odt/TEL3214-Appendix_02-Build_SDN_VM_odt.pdf (last visit 19.09.2018).
14. Installation Guide, https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/pages/1343544/Installation+Guide (last visit 19.09.2018).
15. Mininet: Rapid Prototyping for Software Defined Networks, https://github.com/mininet/mininet (last visit 19.09.2018).
16. Brendan Tschaen, "Floodlight Tutorial", September 2015, https://www2.cs.duke.edu/courses/fall15/cps214/FloodlightTutorial.pdf (last visit 19.09.2018).
17. Kallianiotis Nikolaos, "Software Defined Networks Reactive Flow Programming and Load Balance switching", 2017, http://dione.lib.unipi.gr/xmlui/bitstream/handle/unipi/10680/Kallianiotis_Nikolaos.pdf?sequence=1 (last visit 19.09.2018).
18. LIU WEI, "Constructing SDN with FloodLight Controller in Mininet and Further Thinking", 2013, http://www.cs.sjtu.edu.cn/~wang-b/wireless_new/material/projects/Junior_normal_2013/%E5%88%98%E4%BC%9F.pdf (last visit 19.09.2018).

REFERENCES/ЛИТЕРАТУРА

Shema, M. & B. Johnson (2004). *Anti-Hacker Tool Kit* (pp. 30 – 52). Sofia: SoftPress Ltd.

MANAGEMENT OF SOFTWARE DEFINED NETWORKS WITH FLOODLIGHT CONTROLLER

Abstract. The basic idea for software-defined networks (SDN) is the separation of systems that control the network from systems the data plane. This defines a layer of program control and control of the whole network (control layer). The SDN controller manages the entire network, receives instructions and instructions from the application layer and relays them to the network components. The controller retrieves network infrastructure information – a infrastructure layer – and returns it back to the application layer. In this article we will present the basic architecture and features of the Floodlight controller. We test different topologies of the Mininet network simulator, interaction between virtual machines, adding and intercepting OpenFlow flows from the Wireshark network analyzer.

✉ **Mr. Rumen Todorov, PhD student**
University of Sofia
Sofia, Bulgaria
E-mail: rumentodorov@programist.bg