25th International Scientific Conference UNITECH

INTERNATIONAL SCIENTIFIC CONFERENCE
20-22 November 2025
GABROVO

ИЗГРАЖДАНЕ НА МРЕЖОВА ТОПОЛОГИЯ НА СОФТУЕРНО ДЕФИНИРАНА МРЕЖА С КОНТРОЛЕР OPENDAYLIGHT

Делян Генков*, Мирослав Славов

Технически университет - Габрово, ул. Хаджи Димитър 4, Габрово, България
**кореспондиращ автор: dgenkov@tugab.bg*

CREATING NETWORK TOPOLOGY OF A SOFTWARE DEFINED NETWORK WITH OPENDAYLIGHT CONTROLLER

Delyan Genkov*, Miroslav Slavov

Technical University - Gabrovo, 4, Hadzhi Dimitar Str., Gabrovo, Bulgaria
**Corresponding author: dgenkov@tugab.bg*

Abstract
Software Defined Networks become a very important solution in the modern networking world. It allows centralized monitoring and control of all the network devices through a specialized software, called controller. Although many controllers, both commercial and open source exists, most of them do not offer a graphical user interface, but only some form of application programming interface to interact with. The overall goal of our team is to create a universal application for software defined networks that can work with many existing controllers and provide many different functions. This paper presents the way intended to communicate with one widespread controller – OpenDaylight in order to learn and create the existing network topology.
Keywords: Software Defined network; SDN; Controller; OpenDayLight; Network Topology.

ВЪВЕДЕНИЕ

Софтуерно дефинираните мрежи са важна архитектура, широко използвана в съвременния мрежов свят. Много водещи производители на мрежов хардуер и софтуер, между които Cisco Systems, Juniper Networks (вече част от Hewlett Packard Enterprise), IBM и др.

Традиционните мрежи предполагат конфигуриране на всяко мрежово устройство поотделно, което прави прилагането на промени в мрежовата услуга трудно и предполага възможност за неправилно функциониране на мрежата при конфигурационна грешка.

При софтуерно дефинираните мрежи конфигурациите се правят централизирано на специализиран софтуер, наречен контролер. По този начин промяната се изпълнява едновременно на всички устройства, което пести време и усилия, както и предпазва от грешни конфигурации.

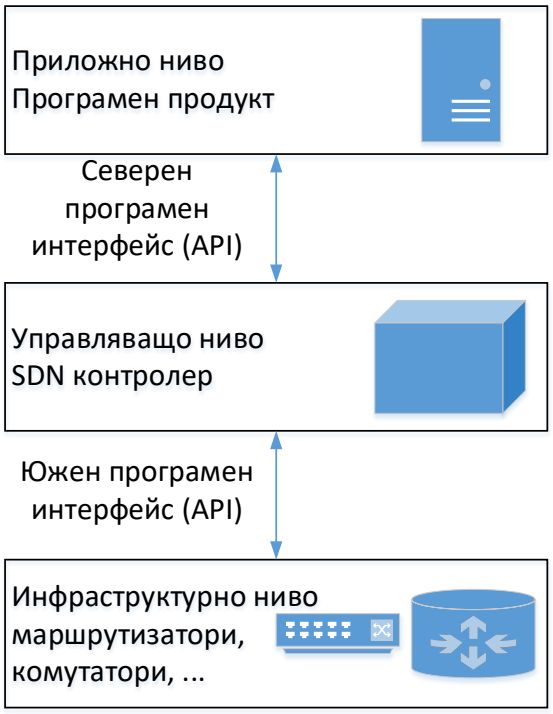
Контролерът обикновено е специализиран софтуер, който няма графичен интерфейс за наблюдение на мрежата или за управление на устройствата. Взаимодействието с него често става чрез REST приложен програмен интерфейс, който е единствения достъпен инструмент. Съществуват много контролери, някои са с отворен код, други са комерсиални. Обикновено всеки производител на мрежови технологии работи със собствен контролер, което прави работата с устрой-

Print ISSN: 1313-230X, Online ISSN: 3033-1404

ства и мрежови технологии от различни производители трудна или невъзможна.

ИЗЛОЖЕНИЕ

Типична архитектура на софтуерно дефинирана мрежа е показана на фигура 1.



***Фиг. 1**. Архитектура на софтуерно дефинирана мрежа*

На инфраструктурното ниво работят мрежовите устройства, които са съвместими със софтуерно дефинираната мрежа (SDN). Те комуникират с управляващото ниво чрез стандартен протокол, наричан южен програмен интерфейс. Съществуват множество такива протоколи, но най-разпространеният и считан за основа на софтуерно дефинираните мрежи е OpenFlow [1].

На управляващото ниво работи специализираният софтуер – контролер. В реалните мрежови топологии поради опасност от отказ на контролера и спиране на цялата мрежа, обикновено се използват повече контролери за разпределяне на натоварването и отказоустойчивост.

Съществува голямо многообразие от контролери за софтуерно дефинирани мрежи, някои от които с отворен код. Някои от най-популярните са:

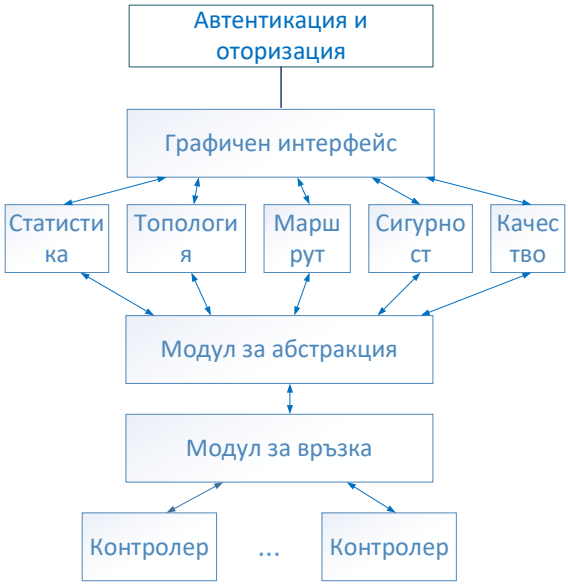
- NOX - оригиналният OpenFlow контролер [2]. Той служи като платформа за мрежово управление, която осигурява програмен интерфейс на високо ниво за управление и разработване на приложения за мрежов контрол. NOX първоначално е разработен от Nicira Networks, сега собственост на VMware. Съвременната версия на NOX поддържа само C++.
- POX - предоставя рамка за комуникация със SDN комутатори, използвайки протокола OpenFlow или OVSDB, използвайки езика за програмиране Python. Това е популярен инструмент за преподаване и изследване на софтуерно дефинирани мрежи и програмиране на мрежови приложения. [3].
- Ryu е компонентно-базирана рамка за софтуерно дефинирани мрежи с отворен код на Python. Ryu предоставя софтуерни компоненти с добре дефиниран програмен интерфейс, който улеснява разработчиците да създават нови приложения за управление и контрол на мрежата. Ryu поддържа различни протоколи за управление на мрежови устройства, като OpenFlow, Netconf и др. За OpenFlow, Ryu поддържа напълно 1.0, 1.2, 1.3, 1.4, 1.5 и Nicira Extensions. Целият код е свободно достъпен под лицензна Apache 2.0 [4]. За съжаление има не особено добра производителност.
- OpenDaylight е сложен и разширяем проект на Java, широко разпространен в индустрията, с интеграция за OpenStack и много облачни платформи [5]. Той се използва в някои от реализациите на Cisco Systems, в решения на RedHat и други комерсиални приложения. Настоящата разработка се базира на OpenDaylight.

Приложното ниво на софтуерно дефинираните мрежи, както и северния програмен интерфейс, чрез който то си комуникира с контролера не са стандартни. Тъй като съществуват много конт-

ролери и много различни приложения, обикновено всеки разработчик създава собствено приложение, което комуникира с един вид контролер.

Екипът на проекта има за цел да разработи многофункционално приложение, което да може да работи с голямо многообразие от контролери и устройства от много производители. За да стане възможно това, трябва да се изучат често използваните контролери и да се установят начините за обмен на информация с тях.

Архитектурата на приложението е описана в [6] и е показана на фигура 2.



***Фиг. 2**. Архитектура на приложение за софтуерно дефинирани мрежи*

Предмет на настоящата разработка е начинът на работа и комуникация с контролера OpenDaylight за целите на топологичния модул и разпознаването на топологията на мрежата чрез комуникация с контролера, както и установяването на връзка с него за целите на модула за връзка.

ОПИТНА ПОСТАНОВКА

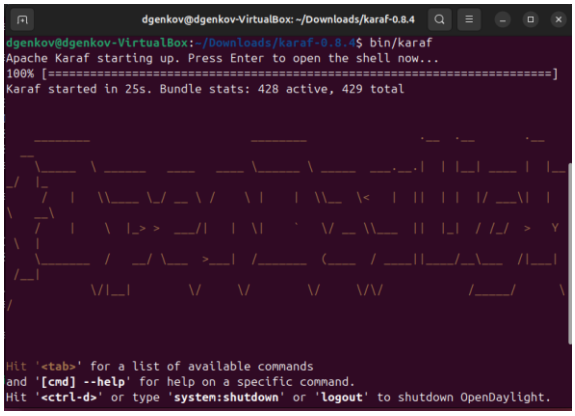
За създаването на опитна постановка са инсталирани две виртуални машини, под управлението на виртуализационната платформа Oracle VirtualBox [7].

Върху първата виртуална машина като операционна система е инсталиран Ubuntu 24.04.3 LTS [8]. Върху него е инсталирана версията на контролера OpenDaylight 0.8.4 с кодово име Nitrogen [9]. Той работи в среда на Apache Karaf [10].

За създаване на мрежова топология е използван симулатор на софтуерно дефинирана мрежа – Mininet [11]. Той се изтегля като предварително инсталирана виртуална машина и се импортира в средата VirtualBox.

За да комуникират помежду си двете виртуални машини, за всяка от тях е създаден виртуален мрежов адаптер от тип Host Only, а за връзка към Интернет през основния компютър за целите на инсталация на софтуерни пакети е създаден втори виртуален адаптер от тип NAT.

Стартира се виртуалната машина с OpenDaylight и по подразбиране получава от DHCP сървъра на VirtualBox IP адрес 192.168.56.101. Това може да бъде променено от настройките на VirtualBox или от самата машина. От командния ред се стартира контролера и се получава достъп до неговата конзола, както е показано на фигура 3.



***Фиг. 3**. Конзола на контролера*

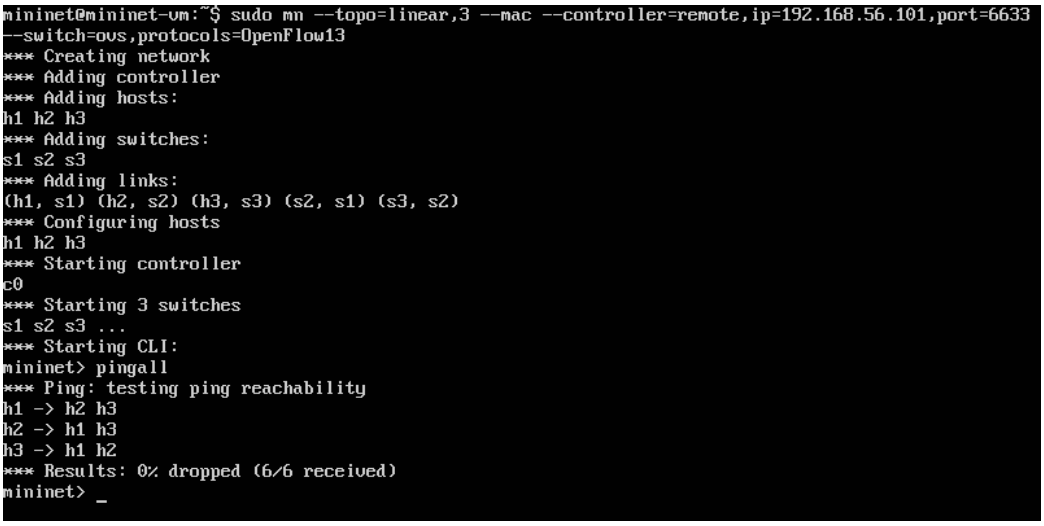
Тук еднократно трябва да бъдат инсталирани използваните модули, използвайки командата:

```
feature:install    odl-restconf-all
odl-l2switch-switch    odl-mdsal-all
features-dlux-all
```

Използваните модули са: odl-restconf-all за REST интерфейса, който ще се използва за комуникация с контролера, odl-l2switch-switch – за комуникация с традиционен комутатор, odl-mdsal-all – за модула за съобщения, данни и отдалечени процедури и features-dlux-all – за web-базираната графична среда. Проведените експерименти показаха, че инсталирането на модулите един по един понякога не позволява функционалността на някои модул и е препоръчително инсталирането на всички модули с една команда, както е показано по-горе. По подразбиране контролерът работи на TCP порт 6633.

За да се симулира софтуерно дефинирана мрежа, се стартира втората виртуална машина със симулатора Mininet. По подразбиране тя получава IP адрес 192.168.56.102.

В конзолата се стартира симулацията, като се избира топологията и параметрите за връзка с контролера, както е показано на фигура 4.



***Фиг. 4**. Стартирана мрежова симулация*

Използваната команда в примера е:

```
sudo mn --topo=linear,3 --mac –
controller=remote,ip=192.168.56.101
,port=6633
switch=ovs,protocols=OpenFlow13
```

Параметрите означават:

- sudo – изпълни командата с администраторски права;
- mn – команда mininet, стартира симулатора;
- topo single,3 – изгради топология с един комутатор и три компютъра. Могат да се изграждат и други топологии, в настоящия пример се използва опростена топология;
- mac – използва малки числа за MAC адреси за лесно помнене;
- switch ovs – да се използва комутатор Open vSwitch (OVS), като се задава и използвания протокол OpenFlow 1.3;
- controller remote – да се използва външен контролер, вместо вградения в комутатора. Като параметри се задават IP адреса и порта на контролера.

С командата pingall се проверява правилната функционалност на постановката, че могат да достигат пакети от и до всички компютри в симулираната мрежа.

По подразбиране REST интерфейсът на контролера се достъпва по HTTP протокол на порт 8181 с потребителско име admin и парола admin и се намира в пап-

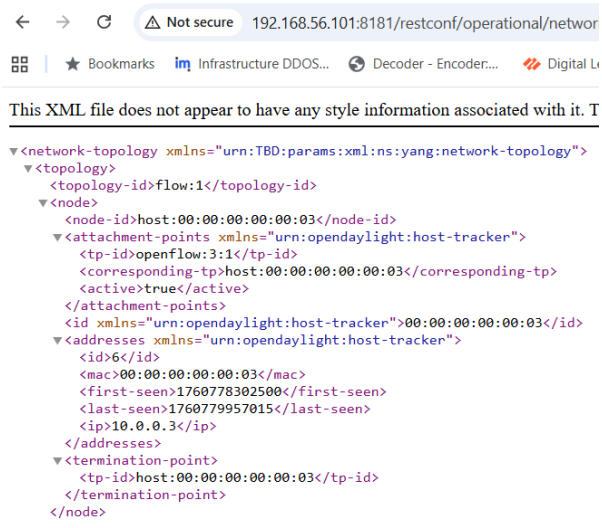
ка restconf. За начална автентикация може да се използва следния URL адрес:

```
http://admin:admin@192.168.56.101:8181/restconf/
```

След успешна автентикация могат да се използват различни команди на програмния интерфейс. За целите на настоящата разработка е използвана командата:

```
GET
http://192.168.56.101:8181/restconf/operational/network-topology:network-topology
```

Резултатът се връща под формата на XML файл, част от който е показана на фигура 5.



***Фиг. 5**. Фрагмент от XML файла*

В него се съдържат следните компоненти:

```
object{1}
   network-topology{2}
      topology{3}
         topology-id:flow:1
         node[6]
         link[10]
```

Топологията включва 6 възела – три компютъра и три порта на комутатор, както и 10 връзки – 5 двупосочни, по една за всеки от трите компютъра и две между портовете на комутатора.

За всеки възел се съдържа следната информация:

```
node-id: host:00:00:00:00:00:03
attachment-points        {4}
id                       {2}
addresses                {6}
  id: 6
  mac:       00:00:00:00:00:03
  first-seen:      1760778302500
  last-seen:       1760779957015
  ip:        10.0.0.3
termination-point        {1}
  tp-id: host:00:00:00:00:00:03
```

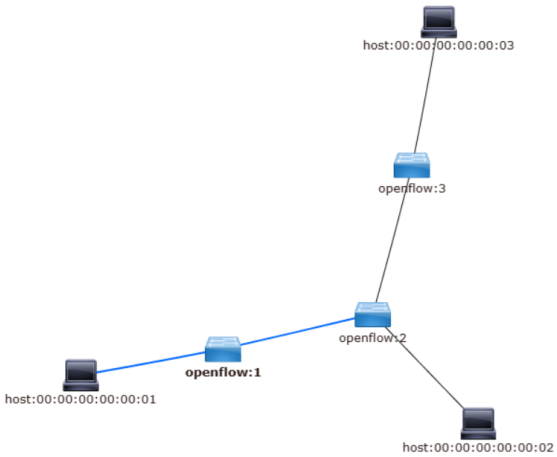
От тук за всеки възел се получава следната информация: IP и MAC адреси, идентификатор, начално и крайно време на свързване към мрежата във формат timestamp (брой секунди, изминали от 01.01.1970 г.).

За всяка връзка във файла се съдържа следната информация:

```
link-id:
host:00:00:00:00:00:01/openflow:1:1
source                   {2}
source-tp:  host:00:00:00:00:00:01
source-node:host:00:00:00:00:00:01
destination              {2}
dest-tp:    openflow:1:1
dest-node:  openflow:1
```

От тук се вижда, че показаната в примера връзка е от компютъра с MAC адрес 00:00:00:00:00:01 до порт 1 на OpenFlow комутатор номер 1. Разбира се, за да има двупосочна комуникация, съществува друга връзка в обратна посока.

Така от получената информация може да се изгради текущата мрежова топология, показана на фигура 6.



***Фиг. 6**. Топология на мрежата*

Така топологията на мрежата е изградена и подадена към топологичния модул, който я подава на графичния интерфейс за визуализация, както и на модулите за маршрутизация, сигурност и качество на обслужването, които могат да взимат решения и да дават възможност на администратора да задава правила за поведение спрямо текущата топология.

ЗАКЛЮЧЕНИЕ

В настоящия документ е описан начинът на изграждане на мрежовата топология на софтуерно дефинирана мрежа, управлявана от контролер OpenDaylight. Тази разработка е част от по-голям проект, имащ за цел изграждане на модулно приложение за наблюдение и управление на софтуерно дефинирани мрежи от различни производители, управлявани от различни контролери.

Бъдещите планове включват поддръжка на други контролери, като NorthStar на Juniper Networks, Cisco OnePK и Trema на NEC.

Допълнителните модули за маршрутизация, сигурност и качество на обслужването ще осигурят допълнителна функционалност за работа със софтуерно дефинирани мрежи.

Източник на финансиране: Настоящият документ е изготвен с финансовата помощ на договор № НИП2025-9 за провеждане на научни изследвания по проект на тема: „Изследване и изграждане на софтуерно дефинирани мрежи” към Технически университет – Габрово.

ЛИТЕРАТУРА

[1] Open Networking Foundation. OpenFlow. https://opennetworking.org/sdn-resources/customer-case-studies/openflow/ Date of usage 05.10.2025.

[2] Sridhar Rao. SDN Series Part Three: NOX, the Original OpenFlow Controller, https://thenewstack.io/sdn-series-part-iii-nox-the-original-openflow-controller/, Date of usage 05.10.2025.

[3] Brian Linkletter, Using the POX SDN controller, https://brianlinkletter.com/2015/04/using-the-pox-sdn-controller/, date of usage: 02.10.2025.

[4] Ryu SDN Framework Community, Build SDN Agilely. https://ryu-sdn.org/. Date of usage 05.10.2025.

[5] OpenDaylight Project, OpenDaylight, https://www.opendaylight.org/, Date of usage, 05.10.2025.

[6] Genkov, D., T. Raykov, M. Slavov, H. Kilifarev, Architecture of an Application for Software Defined Network, International Conference AUTOMATICS AND INFORMATICS`2025, October 09 - 11, 2025, Varna, Bulgaria (ICAI'25)

[7] Oracle. Virtualbox Powerful open source virtualization, https://www.virtualbox.org/, Date of usage 18.10.2025.

[8] Canonical, Ubuntu: Enterprise Open Source and Linux, https://ubuntu.com/, Date of usage 05.07.2025.

[9] OpenDaylight, OpenDaylight downloads, https://docs.opendaylight.org/en/latest/downloads.html, Date of usage 05.07.2025

[10] Apache Software Foundation, Karaf – The modulith runtime, https://karaf.apache.org/, Date of usage 18.10.2025.

[11] Minimet, Mininet - An Instant Virtual Network on your Laptop (or other PC). http://mininet.org/, Date of usage 08.10.2024.