Технологично училище “Електронни системи” към
Технически Университет-София

ДИПЛОМНА РАБОТА

Тема: Проектиране и изграждане на прототип за решение за Internet Exchange Point базирано на Софтуерно Дефинирана Мрежа

Дипломант: *Маргарита Илиева*

Дипломен ръководител: *маг. инж. Здравко Николов*

София
2017г.

Увод

За дълго време не е имало коренна промяна в мрежите и мрежовите технологии, които ние днес познаваме. Повечето от най-използваните протоколи са от 70-80 -те години на миналия век. В днешния динамичен свят, в който технологиите се променят с всеки ден, мрежовите технологии трябва да предложат бързината, мащабируемостта и лесната в управлението, от която света има нужда.

С цел да отговорим на тези изисквания ние качваме мрежите на софтуерно ниво. В историята на информационните и компютърните технологии, нещата доста често са били създавани на ниво софтуер и веднъж разработено се сваля на хардуерно ниво. По този начин иновациите са вървяли напред. За да дадем нов тласък на мрежовите технологии, те трябва отново да бъдат качени на софтуерно ниво. Така се зарежда идеята за Софтуерно Дефинираната Мрежа.

Глава първа

1.1. Internet Exchange Points (IXP)

Internet Exchange Point (IXP) [1] представлява инфраструктура, която предоставя достъп до мрежа, чрез която интернет доставчици и CDN (Content Delivery Networks) [2] обменят интернет трафик между своите мрежи (автономни системи). Автономна система представлява група от мрежи под управлението на една компания с общи правила за маршрутизация от и към Интернет.

IXP нямалява количеството трафик, който трябва да бъде доставен през upstream доставчиците на Internet Service Provider (ISP)[3], като по този начин намалява средната цена, per-bit доставяне на техните услуги. Чрез многото връзки в IXP, се повишава ефективноста и отказоустойчивостта за всеки един клиент, който е вързан в мрежата. Главният фокус на IXP е да улесни свързаността чрез споделени точки за достъп.

Предимствата на IXP:

- Позволява високо скоростен трансфер
- Намалява закъснението

[1] IXP - Internet Exchange Point
[2] CDN - мрежа за предоставяне на съдържание. Най-често се използват за оптимизирано доставяне на статични ресурси, като например изображения и видео
[3] ISP - Интернет доставчик

- Осигурява отказоустойчивост
- Подобрява ефективността на маршрутизарането и пропусквателна способност

Физическата инфраструктура включва множество високоскоростни Ethernet комутатори. Точките за обмен обикновено са в големи изчислителни центрове, където ISP с които трябва да обменят мрежи също имат оборудване (Point of presence). Това прави свързването далеч по-лесно, но също така проблем в една сграда може да навреди на голям брой мрежи.

Обмяната на трафик в IXP по принцип се усъществява на базата на Border Gateway Protocol (BGP). Обикновено ISP задават пътищата, който искат да бъдат споделени. Могат да изберат да маршрутизират с техни адреси или такива предоставени от IXP. В някои случаи, Internet Exchange Point може да служи като резервна връзка, в случаи че директна свързаност на ISP отпадне.

Първите IXP са били споделени, оперативните разходи се покриват и споделят от всички свързани ISP. В усъвършенствени и частни такива, ISP са таксувани месечно или годишно на база брой портове и/или трафик.

Възможно е и свързаност между множество IXP, като по този начин осигуряват свързаност за техните клиенти. В последно време Content delivery Providers (CDN) използват IXP за да се свържат с ISP и други CDN, като по този начин предоставят по-високо скоростна комуникация за техните крайни клиенти.

1.1.1. Видове IXP

- Комерсиални IXP (CIX) [4]

Commercial Internet eXchange (CIX) е IXP, което позволява безплатен обмен на на TCP/IP трафик, включително и комерсиален трафик между ISP. Това е спомогнало за създаването на Интернет какъвто го познаваме днес.

Пример за това е Deutscher Commercial Internet Exchange (DECIX)[5] - неутрален ISP, намиращ се във Франкфурт (Германия). Той е най-големият в световен мащаб от гледна точка на пиков трафик и максимална пропускателна способност от 5 Tbps.

DE-CIX имат над 700 връзки с ISPs, включително и доставчици на широколентов достъп, CDNs, уеб хостинг, както и утвърдени оператори в над 20 изчислителни центрове.

[4] CIX - Комерсиален Internet Exchange Point
[5] DECIX - Немски комерсиален Internet Exchange Point

- Регионални Internet Exchange Point

Пример за това е BIX, който е първият неутрален Internet Exchange Point (IXP) в България. Идеята зад регионален IXP е подобряването на връзките в страната и по-високо скоростен Интернет.

Друг пример е Balkan Internet Exchange Point, който има по няколко локации във всички държави от Балканския регион.

1.1.2. Услуги предоставяни от Internet Exchange point

- Публична Свързаност

Публичен пиъринг е свързаност на множество участници в IXP, чрез който си обменят трафик и е достъпен за нови за свързаности. Реализара се на базата на мрежова инфраструктура, изградена на базата на Ethernet технология. Клиентите могат да се държат през един или няколко на брой обединени портове и да реализират свързаност с други участници. Тази свързаност може да се осъществи чрез изграждане на връзка към някоя от колокациите на IXP.

Peering услугата се изгражда и ползва чрез определен VLAN и чрез предоставени и/или лични IPv4 и/или IPv6 адреси, като е достъпна за различни капацитети започващи от 1GE до 100GE.

- Частна свързаност

Частният peering представлява директна свързаност на два участника в IXP, без тя да е достъпна за други участници.

- Мултикаст Пренос

Мултикаст услугата дава възможност участниците да използват своите портове за пренос на видео и аудио съдържание, като се реализира чрез multicast VLAN-и.
Необходими са две страни:
• Multicast източник– Участник, който притежава всички авторски права върху съдържанието (multicast source VLAN)
• Multicast получател– Участник, който иска да получи съдържанието (multicast receiver VLAN)

- Blackholing[6]

Използва се за митигация на DDOS (Distributed Denial-of-Service Attack). В случай, че клиент на IXP засече атака насочена към него, може да съобщи на Exchange Point-а префиксите от които идва нападението и то да бъде блокирано още в платформата на IXP. Има възможност да се създадат политики за филтриран трафик, които могат да бъдат изпратени на група група от свързани клиенти.

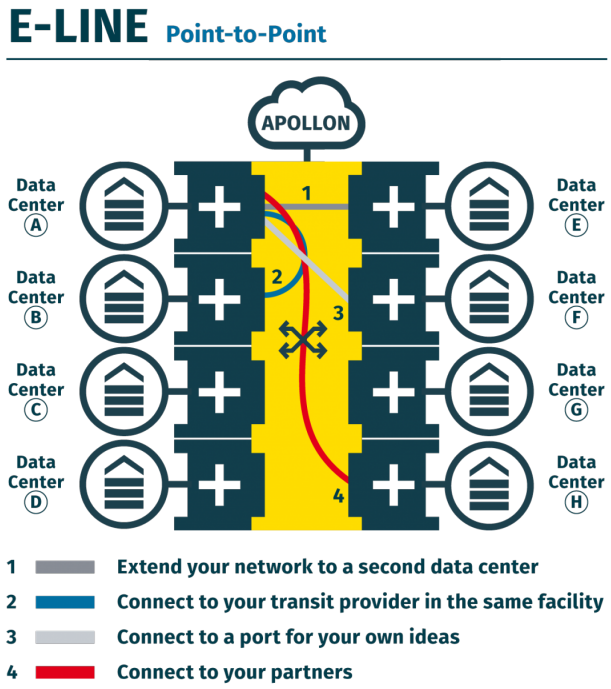
[6] Blackholing - при DDOS атака мрежите се блокират директно от IXP

- DirectCloud[7]

IXP може да предложи директна свързаност към cloud услуги като Azure и AWS.

- E-Line - virtual point-to-point private line[8]

Чрез point-to-point услугата, клиентите могат да конфигурират един или множество VLAN на техните портове, което им осигурява свързаност до други клиенти свързани в същата платформа.

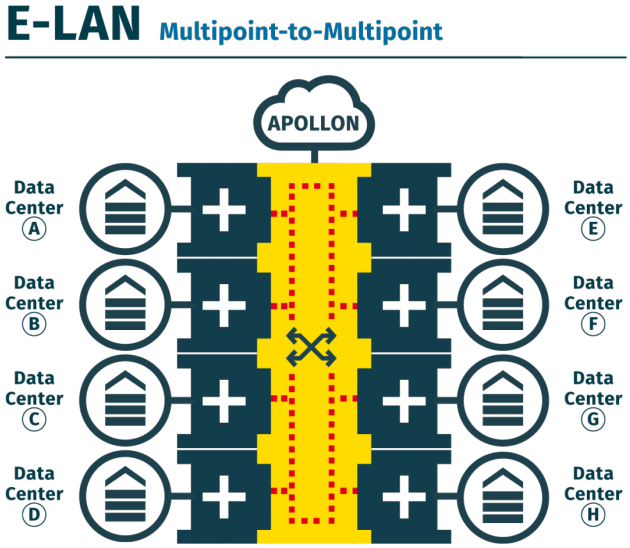


Фиг. 1.1. E-Line Point-to-Point свързаност

[7] Direct Cloud - IXP предлага директна свързаност с някой от големите доставчици на облачни услуги
[8] E-Line Point-to-point - Vlan за директна свързаност между клиенти

- E-LAN[9]

Предоставя multipoint-to-multipoint услуги, подобно на традиционен VLAN, който се свързва към множество портове.



Фиг. 1.2 E-LAN Multipoint-to-Multipoint свързаност

1.1.3. Използвано оборудване и протоколи в IXP

В типичен IXP се използват един или множество мрежови комутатори. В началото са били използвани fiber-optic inter-repeater link (FOIRL)

[9] E-LAN - споделяне на множество портове между клиенти

концентратори или Fiber Distributed Data Interface (FDDI) пръстени, като на по-късен етап е преминато на Ethernet.

Част от първите IXP са имплементирали ATM[10] комутатори, но Ethernet се е наложил като много по-евтин и достъпен за изграждане. Масово портовете днешно време в IXP са от 1Gbit/s до 10Gbit/s в по-големите изчислителни центрове, като в AMS-IX и DE-CIX, се предлага 100Gbit/s.

Интернет трафик обмена между два участника в IXP се случва в повечето случаи на базата на Border Gateway Protocol (BGP) конфигурирано между тях. Те избират кои пътища да споделят през директната си връзка - било то пътища от техните мрежи или пътища до други IXP, в които те са свързани. От другата страна, другия участник може да приложи филтриране на получените пътища, дали ще приеме пътищата и да ги рутира или да ги игнорира и да използва други пътища аз да достигне дадени адреси.

Transit exchange за разлика от Local Exchange, няма абонати и за това не играе ролята на източник на трафика в мрежата, той само събира и пренасочва трафика по локалните обмени.

1.1.4. Проблеми срещани в момента

Най-често срещаните проблеми в момента в едно IXP са свързани с менажирането на мрежата. Инсталирането на нови мрежови устройства

[10] ATM - Asynchronous Transfer Mode

изисква конфигурация, която трябва да бъде извършена от квалифицирани мрежови инженери. В случай, че клиент иска различни видове свързаност, било директна към друг участник или към публичен peering, трябва да се правят мрежови, а и понякога хардуерни промени.

Повечето от тези проблеми могат да бъдат решени със софтуерно дефинирана мрежа (СДМ). Добавянето на нов клиент или смяната на услуга, може да става за секунди. СДМ дава възможността за управление на цялата мрежа на много нива (L2; L3; L4 от OSI модела) от един контролер и могат да бъдат добавяни рестрикции по MAC адреси, IP адреси и портове. В legacy мрежа това би изисквало конфигурации на комутатори, маршрутизатори и защитни стени, докато в СДМ всичко това се случва в общ контролер.

1.2. Софтуерно дефинирана мрежа (СДМ)

Софтуерно Дефинирана Мрежа е технология с високо ниво на адаптация, лесна за управление и изключително динамична. Архитектурата на СДМ разделя Forwarding Plane от Operational Plane. По този начин мрежовия контрол става програмируем, а инфраструктурата отдолу да бъде абстрактна за приложенията и мрежовите услуги. OpenFlow протокола е съществена част от конструирането на Software Defined Network (SDN). Архитектурно ключови аспекти са разгледани в rfc7426.

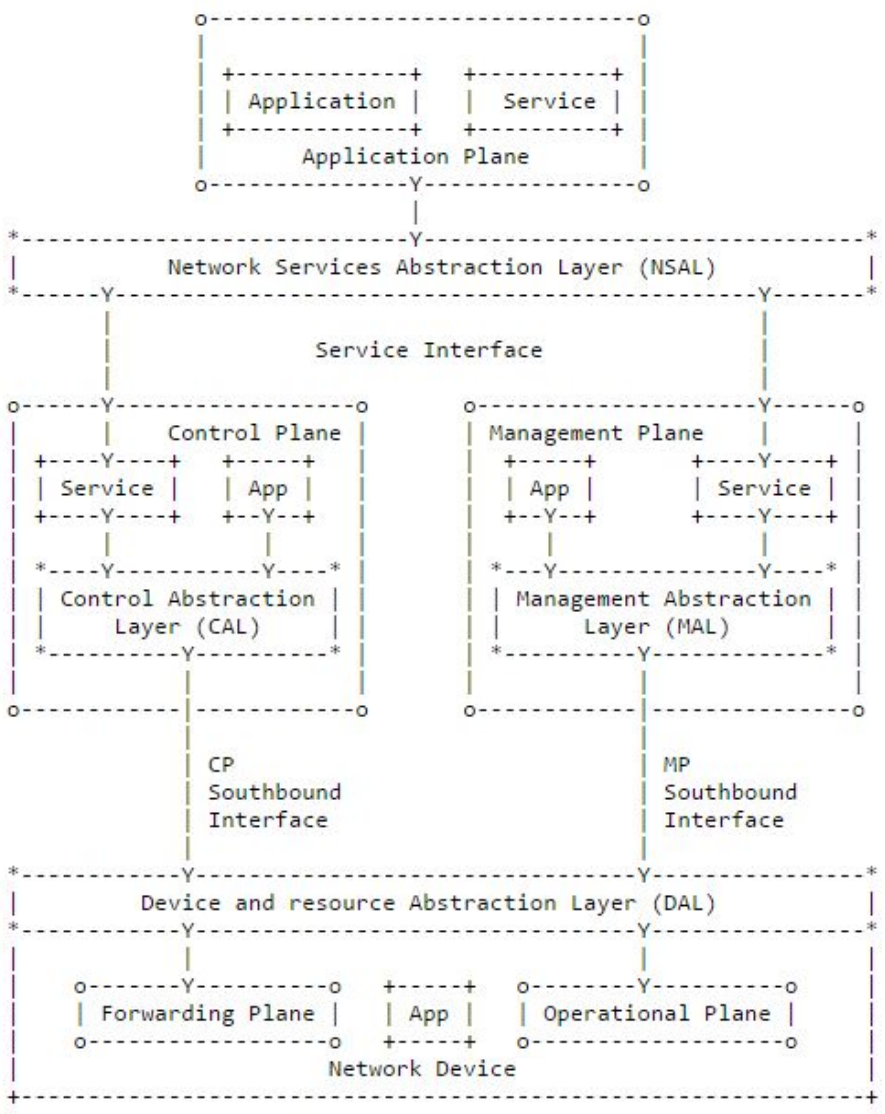


Figure 1: SDN Layer Architecture

Фиг. 1.3. SDN архитектура

SDN работи на принципна, че много на брой SDN комутатори се управляват от един контролер. Това наподобява например вече съществуваща идеология с телефонната централа на Ericsson MD110 PBX. MD 110 е изградена от модулни блокове, 1 line interface modules, LIM (Line Module), които са големи

системи свързани чрез централна група комутатори, GS (Group Switch). Всяка една от телефонните централи може да работи по отделно или като част от голяма MD система.

1.2.1. Предимства на СДМ

- Централизирана мрежа

Софтуерно дефинираните мрежи предоставят централизиран оглед на цялата мрежа, което го прави по-лесно да се управление и инсталация на нови устройства

- Сигурност

SDN контролера осигурява централна точка на контрол, разпространяване на политики за сигурност из цялата мрежа. Евентуален минус би бил, че има централно звено, което може да бъде атакуано, но посредством fault-tolerance структури това може да бъде предотвратено.

- Overhead Reduction

В сегашните физически среди, изолацията на машина изисква конфигурация на VLAN на различни мрежови устройства, в това число маршрутизатори и комутатори. При SDN това се слючва в контролера, което го прави лесно за Доставчиците на услуги да изолират дадена клиентска машина, било то физическа или виртуална от другите клиенти.

- Намален downtime на инфраструктурата

SDN виртуализира повечето от мрежовите процеси, което прави изключително лесно подобряване и тест на машините в инфраструктурта. Цялата конфигурация на SDN контролер може да бъде запазена и да бъде възстановена бързо в случай на проблем.

- Статистики и подобрения

Мрежата може много по-лесно да бъде следена на базата на което да се направят различни видове статистически анализи, било то кога мрежата е по-натоварена, от какви мрежи в какво вреве има повече атаки. Може да се следи колко са използвани дадени ресурси и така да се създадат планове за клиенти на IXP или DC. Което би помогнало за намаляване на CapEx и OpEX.

1.2.2. Openflow

OpenFlow е отворен стандарт менажиран от Open Networking Foundation. Той указва протокол чрез който комутатор и отдалечен контролер могат да модифицират поведението на мрежовите устройства чрез добре дефинирани “forwarding instruction set”.

Разгледана е 3-tier архитектурата на SDN:

- Application Tier
    - Northbound APIs са програмните интерфейси между платформата на контролера и работещите отгоре приложения. Архитектурата на три нива се фокусира върху тези API[11], които експонират универсалните мрежови абстракции и функционалности в контролера и по-точно, Floodlight - за да бъдат използвани от мрежови приложения. Тези програмни интерфейси са публично достъпни за използване и подобряване от общността занимаваща се с отворен код.

- Control Plane Tier
    - Control plane tier се явява самия контролер в SDN архитектурата. Контролера се свързва с всяко едно мрежово устройство. Създава action-и на базата на различни видове характеристики на базата на MAC адреси, IP адреси, портове и други, които се разпращат

[11] API - Application programming interface

на комутаторите. В случай че пристигне пакет за който няма създаден action на някой порт от комутаторите, той се изпраща към контролера, който от своя страна трябва да вземе решение как да процедира с него. След това да създаде action и той да бъде изпратен на комутатора.

Могат да се създават различни видове action-и и flow-ове. Това включва, че едно правило, може да излючва друго или правила да се наслагват подобно на MPLS[12] label.

Open Floodlight контролер поддържа Openflow от 1.0 до 1.4.

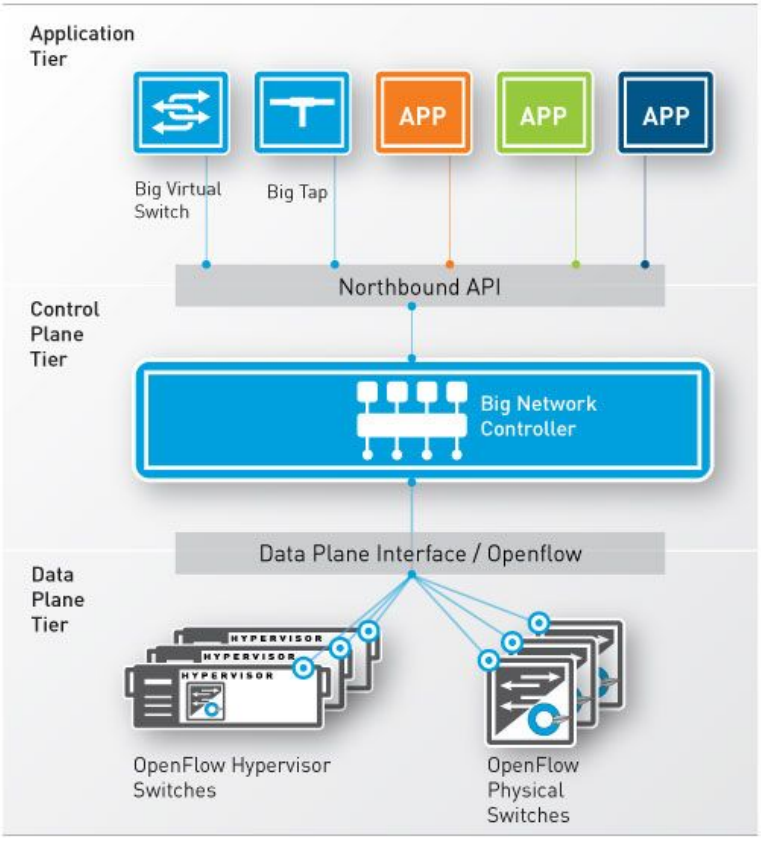
- Data Plane Tier

  В 3-tier SDN архитектурата, контролера е физически разделен от data forwarding plane, примерно: физически или виртуален комутатор може да комутира пакети и отделен сървър да изпълнява network control plane.

  Това разделение позволява на control plane да бъде имплементиран по различен модел на разпределение от data plane. На второ място, позволява разработването на control plane на отделна работна среда от тази на която в даденият момент функционира работещ контролер за разлика от стандартните комутатори и маршрутизатори. За да работи това, трябва да има някакъв начин

[12] MPLS - Multiprotocol Label Switching

контролера все пак да комуникира с data plane. Най-популярният начин за това е OpenFlow - стандартен интерфейс за контол на комутатори.



Фиг. 1.4 OpenFlow архитектура

1.2.3. Хардуерни решения

OpenFlow версия 1.0 излиза през 2009 година, причината за това че чак сега се обръща повече внимание е липсвата на стабилен хардуер, който да го поддържа. Той е протокол с отворен код, който спокойно може да бъде

имплементиран и конфигуриран на стандартен комутатор, но известни проблеми със скоростта биха възникнали.

Комутатора се намира на слой 2 от OSI модела, което означава че взима решения базирани на Layer 2 кадри. Колкото повече портове му бъдат добавени, толкова повече изчисления комутатора трябва да прави. В днешни дни, модерното CPU (central processing unit) x86 може да се справи с тези операции, но bottleneck-а идва при това, че тази информация трябва да стигне до него през периферни устройства и шини. Нормален сървър би използва шина с общо предназначение за да пренесе тази информация, заедно с всички останали изчисления, което би забавило драстично комутирането на пакети. Поради тази причина в комутаторите се използва ASIC (Application-specific integrated circuit) , което е CPU, което не е с общо предназначение, а е направен специално да прави решения за комутирането на пакети, за това и е изключително по-бърз и производителен. Свързан е с писти директно към портовете, което отстранява драстично забавянето. ASIC може да комутира милиони пакети в секунда, заедно с него се използват и TCAM (Ternary Content Addressable Memories) памети, които са далеч по-бързи от RAM паметите.

При RAM паметта операционната система предоставя адреса в паметта където тази информация се съхранявана. Информацията съхранена в CAM (Content Addressable Memories) може да бъде достъпена, чрез изпълняване на заявка за съдържанието и паметта да намери адреса, където информацията е съхранена. TCAM е паралелен, което му позволява да работи много по-бързо от RAM паметите.

Не се използва широко защото е скъп да бъде направен, консумира изключително много електроенергия и генерира високо количество топлина. Използва се често в мрежовото оборудване, при високо скоростни маршрутизатори и комутатори за да се повиши скоростта при route look-up, packet classification и forwarding, както и при access control list-based команди.

Има множество доставчици като: Cisco, Mikrotik и Arista, които имат вече създадени хардуерни openflow съвместими комутатори, но те не са масови и не са обратно съвместими с предишни модели.

SA	DA	Interface
10.0.0.8	172.20.0.1	Fa 1/7

Таблица. 1.1. Таблица в ASIC

1.2.4. Софтуерни решение

Има множество софтуерни решения, които са имплементирали OpenFlow като софтуерен контролер и виртуални комутатори, които са Openflow enabled.

В графиката по-долу е направено сравнение между най-популярните OpenFlow контролери:

Име	Използван програмен език	Информация
Nox/Pox	Python	Оригиналния OpenFlow контролер от Stanford University
Beacon	Java	OpenFlow контролер с отворен код под GPL лиценз от Standford University
Open Floodlight	Java	Разработен от Big Switch, най-стабилна сходимост с OpenvSwitch, базиран на Beacon
Open DayLight	Java (YANG data models)	Универсален контролер, разработен от Open DayLight Consortium - използва се от Cisco (XNC,

		WAE), Ericsson
Ryu	Python	Напълно интегриран с Openstack networking (Neutron)
Maestro	Java	Създадени за научни разработки
SNAC	Python и C++	Базирано на Nox, създаден за enterprise решения

Таблица. 1.2. Openflow софтуерни контролери

По-долу са опоменати виртуалните комутатори, които имат поддръжка на OpenFlow:

Име	Използван програмен език	Информация
OpenvSwitch	C/Python	Използва се и като виртуален комутатор в

		среди с виртуализация и е интегриран в Linux kernel ядрото
OpenFlow Reference	C	Минимална поддръжка на OpenFlow стек
Pica8	C	Switch software platform за hardware switching chips, които включват L2/L3 стек и поддръжка на OpenFlow
Indigo	C	OpenFlow порт за OpenWRT wireless среди
OpenFaucet	Python	Имплеметация на OpenFlow 1.0.0 протокол, може да се използва за имплементация,както и на комутатор, така и на контролер, писан на Python

Таблица. 1.3. Openflow виртуални комутатори

Глава втора

2.1. Технологичен стек

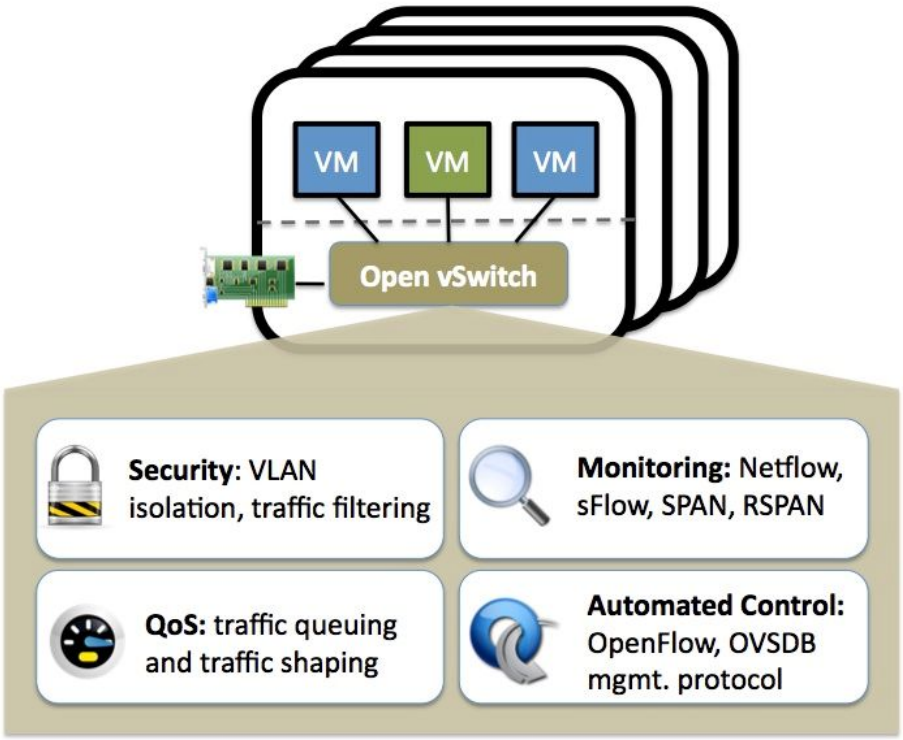
2.1.1. Избор на Openflow комутатор

За целта на дипломната работа е избран софтуерен комутатор - OpenvSwitch (OVS). Към момента софтуерните комутатори са по-добре развити и с повече функционалности. Избран е OpenvSwitch поради факта че е единствения от изброените в глава 1, таблица 1.3 - който е вграден в Linux ядрото от версия 3.3 нагоре. Той е разработен от Linux Foundation, запазена марка на Linus Torvalds.

Виртуализационните hypervisor-и се нуждаят от способността да свържат трафика между виртуалните машини и външният свят. На Linux базираните hypervisor-и, това означаваше вграден L2 комутатор (Linux bridge), който е бърз и надежден. OpenvSwitch обаче е насочен към multi-server virtualization среди, за които предишният стек не е подходящ. Тези среди са в повечето случаи характеризирани с високо динамични end-points, поддръжката на логически мрежови абстракции и интегрирането на нови такива.

OpenvSwitch е изграден като продукт, multilayer virtual switch лицензиран като Open Source технология под Apache 2.0 лиценз. Архитектурата му е предназначена да даде възможност на масова мрежова автоматизация, която

да е програмно осигурена, като все пак поддържа стандартно управление на интерфейси и протоколи (пример:NetFlow, sFlow, IPFIX, RSPAN, CLI, LACP, 802.1ag). Проектиран е да поддържа различни видове измежду различни видове физически или виртуални сървърни решение, подобно на VMware’s vNetwork и Cisco's Nexus 1000V.



Фиг. 2.1. Open vSwitch

Виртуалният комутатор - OpenvSwitch може да оперира и като софтуерен комутатор и като контролен стек за хардуерен чип в комутаторите. Той е

имплементиран в множество виртуализационни платформи и switching chipsets. OVS е комутатора по подразбиране в XenServer 7, Xen Cloud Platform - също така поддържа Xen, KVM, Proxmox VE и VirtualBox. В допълнение на това е интегриран в множество виртуални среди за управление като OpenStack, openQRM, OpenNebula и oVirt. Пакети за OVS са достъпни под Ubuntu, Debian, Fedora и OpenSUSE, има го и за FreeBSD и NetBSD. OpenvSwitch в разработка е внесен и в DPDK (Data Plane Development Kit).

Единична конфигурация на OpenvSwitch би могла да поддържа всичките версии на Openflow протокола. В идеалния случай, дори и по време на работа, трябва да бъде в състояние да поддържа всички версии в един и същ момент на различни Openflow bridge, а дори и на същия bridge.

Основният подход за достигане на толкова висока съвместимост е да се направи абстракция между основният протокол и надгражданията над него. По този начин ще има слой на протокола който преобразува по-високите версии към съвместими към по-ниските.

Използвани са следните подходи:

- Port numbering
- Openflow 1.0 има 16 битов номер за портовете, а по-новите версии използват 32 битов такъв. За сега, OVS поддръжката за по-нови версии

изисква всички номера на портове да са от 16 битов диапазон, превеждайки reservedOFPP_* портови номера.

- Различни видове action-и.
- Openflow 1.0 и по-късни версии имат много различна идеология за action-ите. OVS превежда action-ите и инструкциите на общо вътрешно представяне.

2.1.2. Избор на OpenFlow контролер

За целта на дипломната работа е избран софтуерен контролер OpenFloodlight, той е с добре изградена поддръжка на OpenFlow, документиран е по достъпен начин и има удобен за използване Web GUI[13] и REST API[14].

Floodlight е OpenFlow SDN контролер с отворен код предназначен за enterprise решения под Apache лиценз, базиран на Java. Поддържа се от общност на разработчици, включително редица инженери от Big Switch Networks. Floodlight контролерът е създаден да работи и поддържа разрастващия се брой комутатори, било то физически или виртуални, маршрутизатори и точки за достъп, които поддържат OpenFlow.

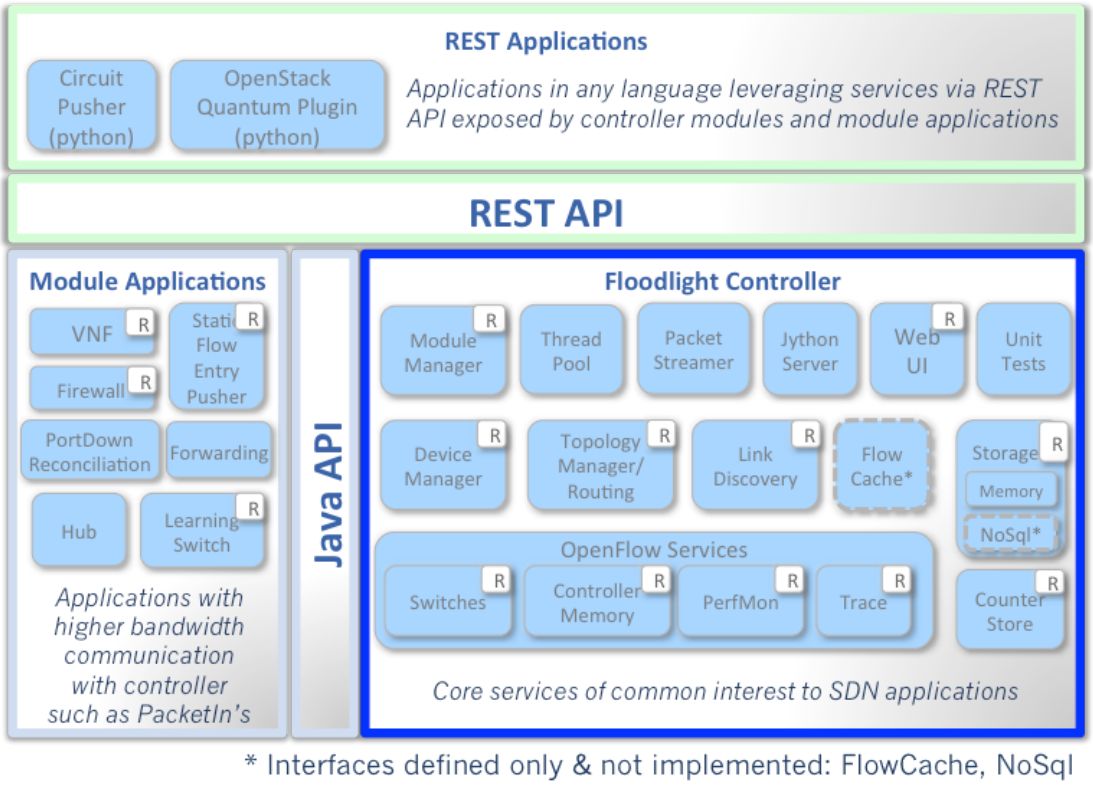
Предимства на Floodlight контролера:

[13] WEB GUI - WEB Graphical user interface
[14] REST API - Representational State Transfer API

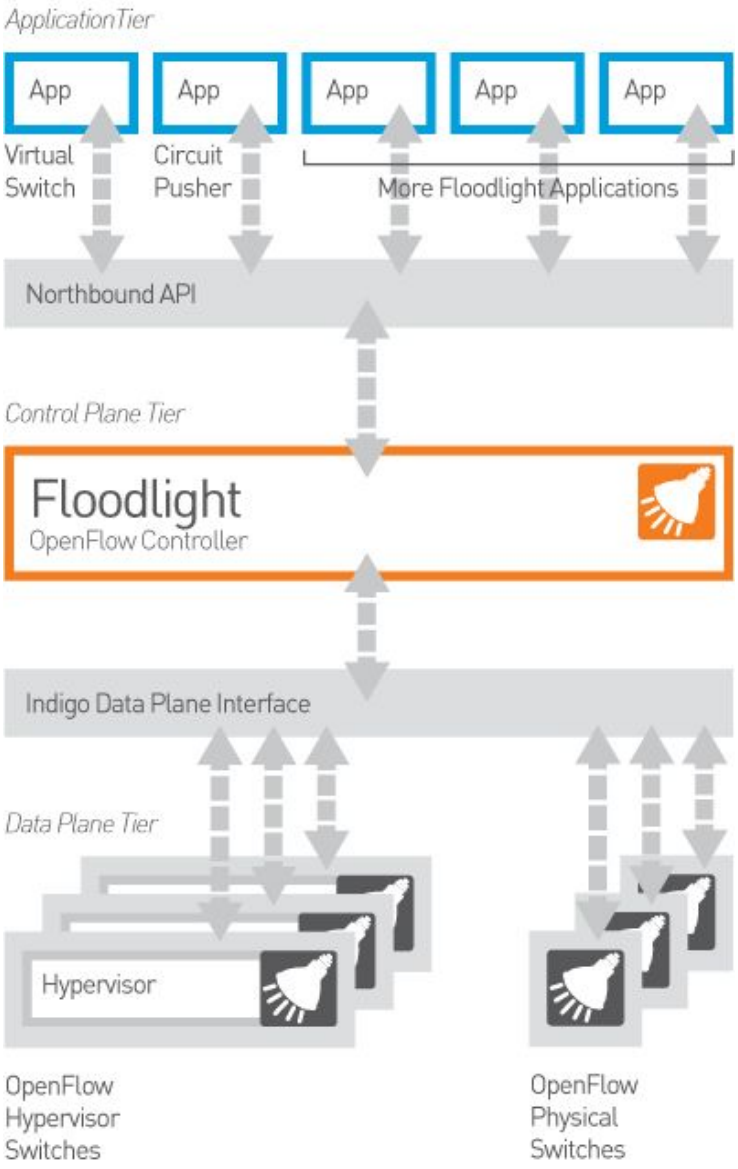
- Поддържа голям брой виртуални и физически OpenFlow комутатори
    - Виртуални: OpenvSwitch, ofsoftswitch
    - Хардуерни: Arista 7050, Brocade MLXe, Dell S4810, Extreme Summit x440 +, HP 3500+ , Huawei openflow-capable router platforms, IBM 8264, Juniper (MX, EX), NEC IP8800

- Може да работи в смесена среда от OpenFlow и non-OpenFlow мрежи - може да управлява от множество отделни OpenFlow комутатори
- Проектиран да бъде високо ефективен - многонишков
- Поддържа Openstack cloud orchestration platform

Flooflight не е просто Openflow controller, а и сбор от различни видове приложения изградени върху самия контролер. Реализира набор от общи функционалности за контрол на OpenFlow мрежата. Множеството приложения са конструирани върху Java модули компилирани в началото заедно с Floodlight и такива конструирани върху REST API.



Фиг. 2.2 OpenFlow Floodlight RestAPI



Фиг. 2.3 OpenFlow Floodlight архитектура

2.2. Ползите от СДМ в IXP

Целта на дипломната работа е да даде примерна имплеменация за решение на най-общите проблеми и трудности срещани в един Internet Exchange Point. Голямото предимство на SDN е че дава възможността за администрация на много нива от една мрежа.

Софтуерните приложения, които са драстично по-динамични и позволяват много по-голяма оптимизация, не са налични в локално менажираните комутатори и маршрутизатори.

Скалируемостта е лесно постижима при SDN, защото самата мрежа се скалира софтуерно. Добавянето на повече хардуерни устройства резултира в добавянето на нови инструкции, които са лесни за създаване, управление и контрол. По този начин се създава много по-ефикасно менажиране на трафика. С използването на СДМ мрежата може да бъде изградена според специфичните нужди на клиента.

SDN приспособява мрежата за повече гъвкавост и усъвършенстване.

Тази гъвкавост би допринесла за много по-добри услуги в един IXP. Преимуществото на Софтуерно дефинираните мрежи е че позволяват лесно включване и изключване на клиенти от дадени групи. Това може да са

публичен или частен обмен на мрежи, DDOS митигации и намаляване на разходите.

2.3. Демонстрационна среда

2.3.1. Комутация на пакетите

Демонстрационната среда която ще бъде представена се състои от виртуална машина с CentOS 7 и OpenvSwitch с 8 на брой, 10Gbit Ethernet порта с общ капацитет 80Gbit. Към един от тези портове са свързани вътрешните виртуални машини (на Ubuntu 16.04), които са създадени върху XenServer и са както следва:

- BG peering (bg_peering)и IXP peering (ixp_peering) - eth0 от XenServer хоста, като са на отделни виртуални интерфейси

Клиентските машини:

- Internet Service Providers (ISPs) - isp1 и isp2
- Client 1 - свързаност към BG peering и директна свързаност към ISP2
- Client 2 - свързаност към IXP peering
- Client 3 - свързаност към BG peering и директна свързаност към ISP1
- ISP 1 и ISP 2 - свързаност към IXP peering

2.3.2. Реализиране на вътрешна инфраструктура за SDN IXP

В Internet Exchange Point-а главното е обмена на BGP трафик между отделните клиенти. За реализирането на вътрешната инфраструктура се използва Citrix Xen Server 7.0, който е базирана на Xen hypervisor и за управлението може да бъде използван XenCenter. Поддържа до 1,5TB RAM на виртуална машина и 5TB на хост. В него са създадени три виртуални машини, две от които (BG peering и IXP peering) са BGP маршрутизатори посредством приложението - Quagga.

Quagga е маршрутизиращ софтуерен стек, чрез който могат да се конфигурират и имплементират OSPFv2, OSPFv3, RIPv1, RIPv2, RIPng и BGP-4 на UNIX платформи, по-конкретно FreeBSD, Linux, Solaris и NetBSD. Quagga е част от GNU Zebra - multi-server маршрутизиращ софтуер, който предоставя TCP/IP базирани маршрутизиращи протоколи. Zebra превръща виртуална или физическа машина в пълноценен рутер. Архитектурата на Quagga се състои от централен демон на процеса, Zebra, която създава ниво на абстракция на Unix ядрото отдолу и представя Zserv API над TCP потока на Quagga клиент. Zserv клиентите прилагат маршрутизиращи протоколи и комуникират routing updates на Zebra демон процеса.

Софтуерният контролер Floodlight е отговорен за взимането на решения за действия относно даден пакет, създаването на action за него и изпращането на този action на комутаторите. За целите на демонстрационната среда всички BGP рутери ще се намират в обща мрежа 10.0.0.0/8, като

ограниченията кой до кого да има достъп ще бъдат имплементирани от контролера. Той се намира на отделна машина от XenServer, като е инсталиран върху Ubuntu 16.04. Клиентските машини нямат достъп до контролера, той е с публичен адрес:

- Controller - 78.130.176.33:6653

2.3.3. Връзка с клиентски машини

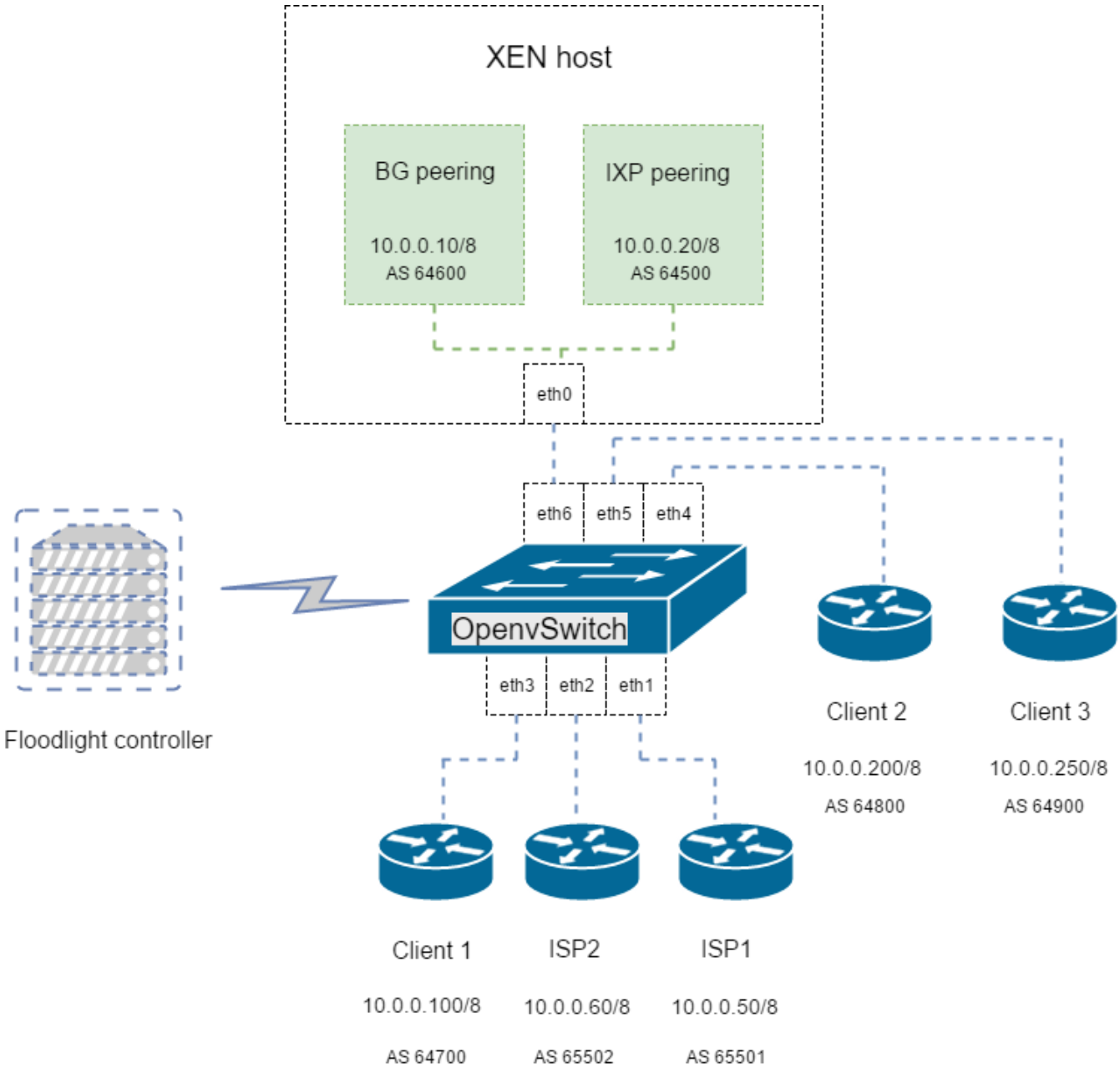
Всяка една от клиентските машини се свързва към openflow enabled маршрутизатор, за целите на демонстрационната среда това е OpenvSwitch. За Client 1 е конфигурирана Mikrotik виртуална машина свързана към eth3 на OVS. ISP 1, ISP 2, Client 2 и Client 3 са Quagga виртуални машини свързани респективно към eth1, eth2, eth4 и eth5. В реална инфраструктура, не е нужно клиентски машини да са виртуални или под Unix операционна система, единственото изискване е физическа свързаност към маршрутизатор в мрежата на IXP.

Всяка една от клиентските машини получава частен адрес от 10.0.0.0/8.

Адреси са зададени както следва:

- bg_peering - 10.0.0.10/8
- ixp_peering - 10.0.0.20/8
- isp1_ovs - 10.0.0.50/8
- isp2_ovs - 10.0.0.60/8
- mikrotik _ovs - 10.0.0.100/8

- client2_ovs - 10.0.0.200/8
- client3_ovs - 10.0.0.250/8



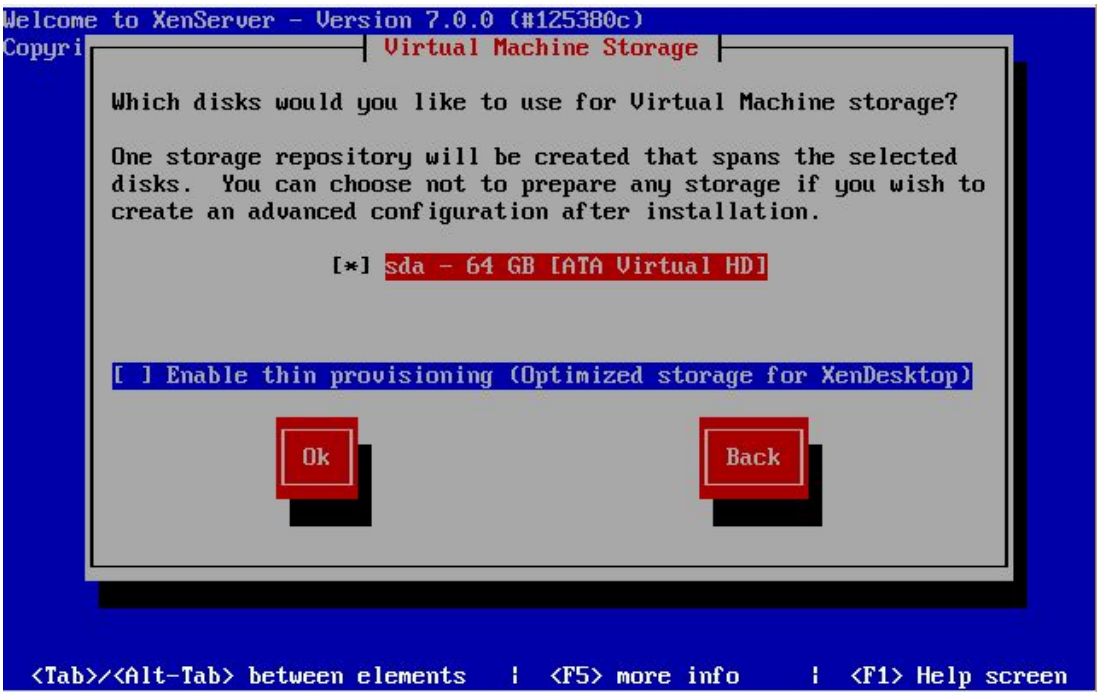
Фиг. 2.4. Архитектура на дипломната работа

Глава трета

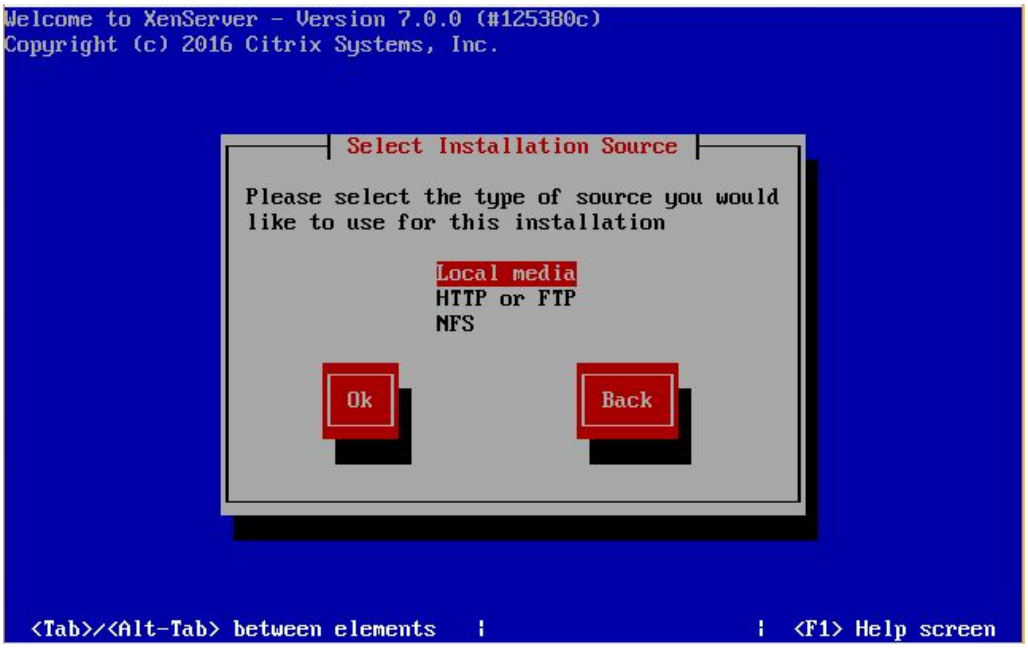
3.1. Инсталация и конфигурация на хипервайзор

За вътрешната инфраструктура се инсталира Xen hypervisor на който ще бъдат създадени виртуални сървъри за BG peering и IXP peering.

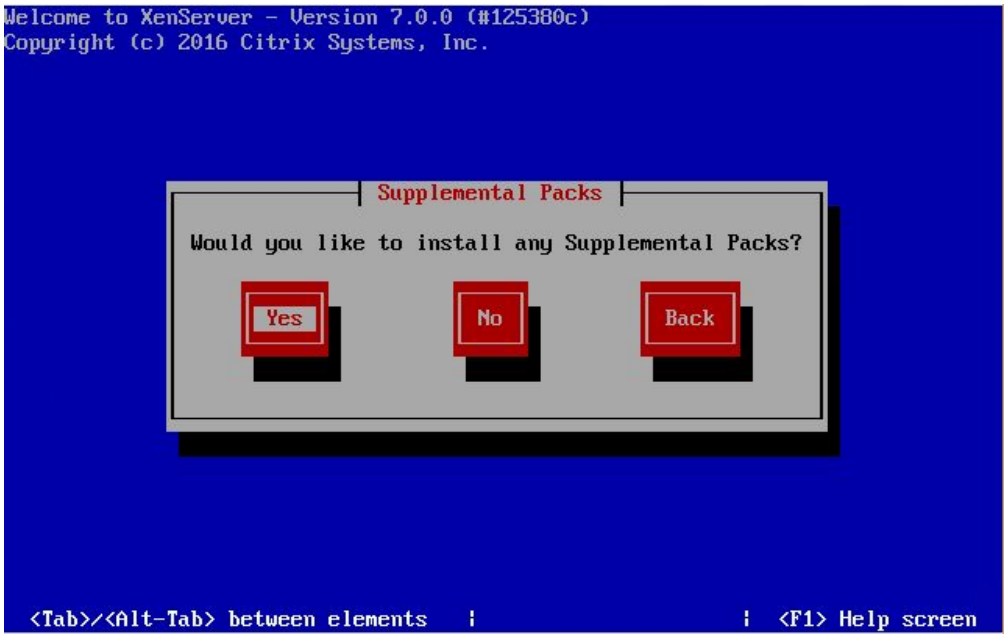
Инсталация на Citrix XenServer 7.0:



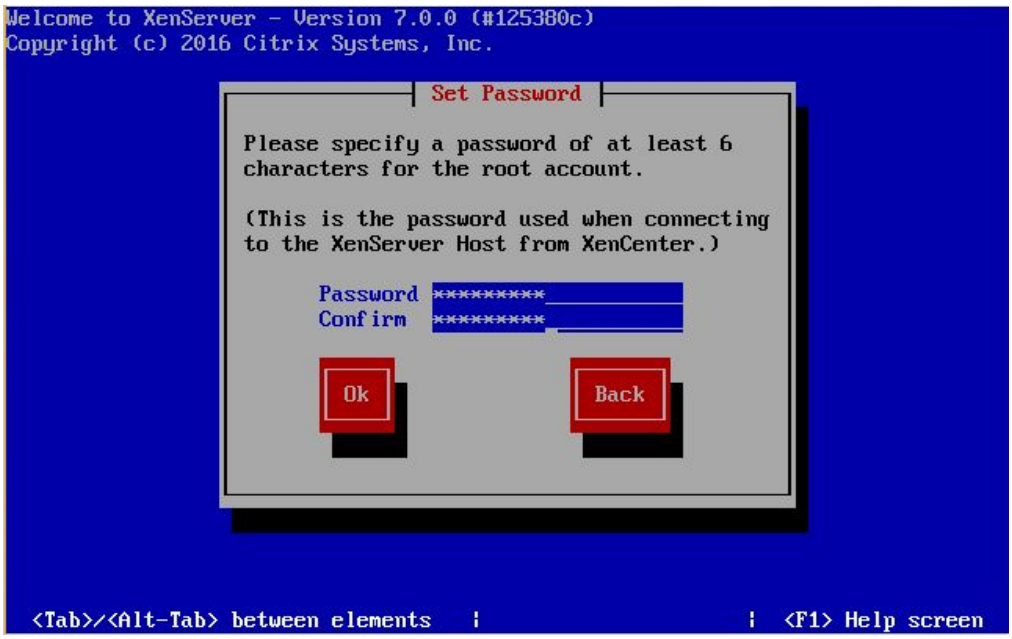
Фиг. 3.1. Избор на твърд диска върху който да се осъществи инсталацията



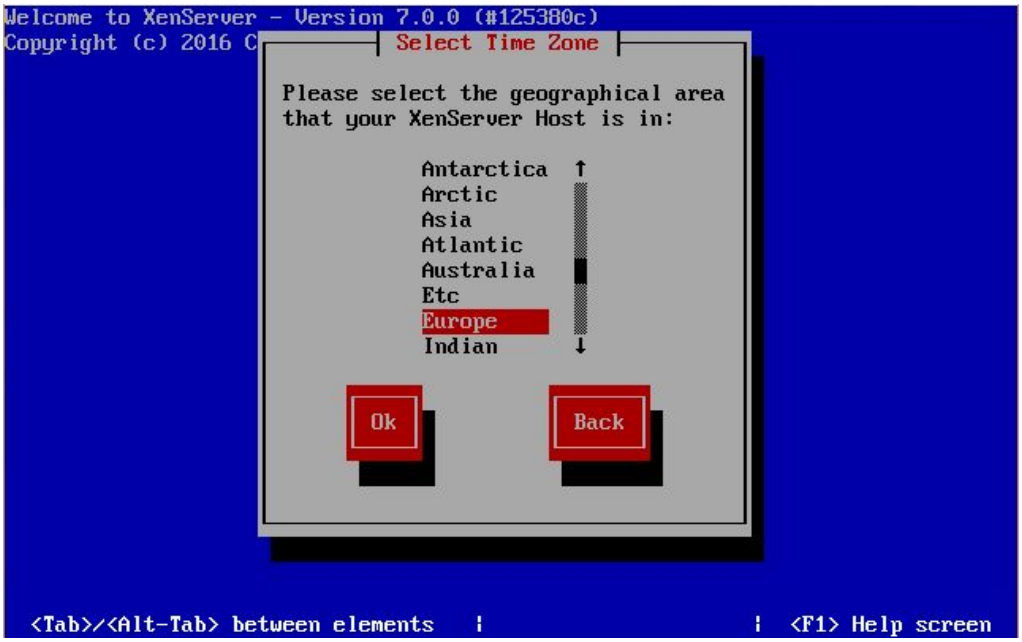
Фиг. 3.2 Избор на медия, от която да се извърши инсталацията на операционната система



Фиг. 3.3 Избор дали да се инсталират допълните пакети, които са с XenServer ОС

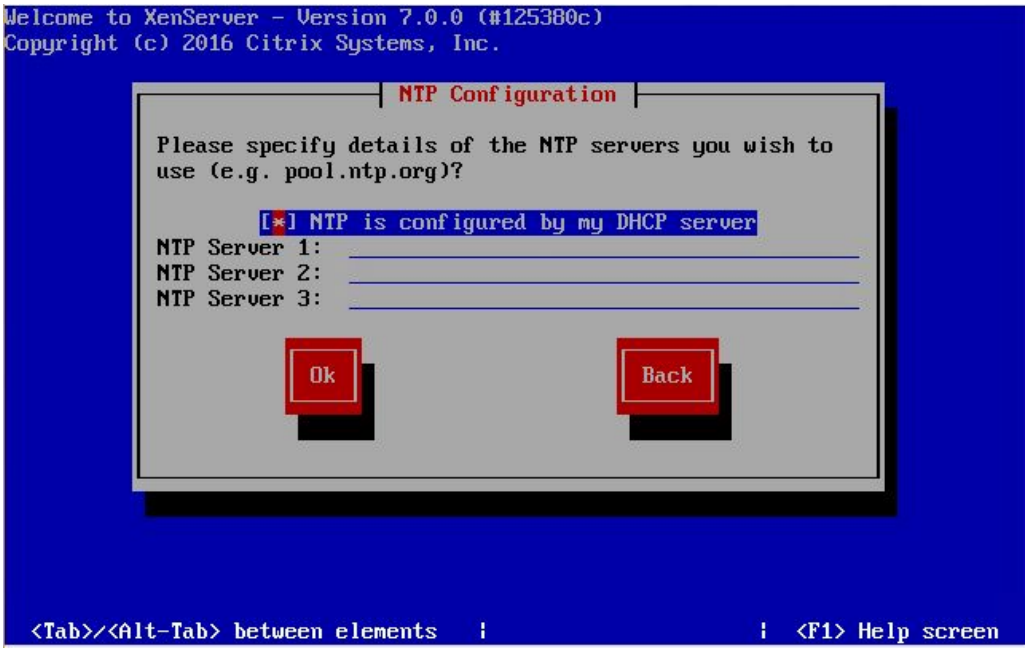


Фиг. 3.4 Задаване на парола за root акаунта

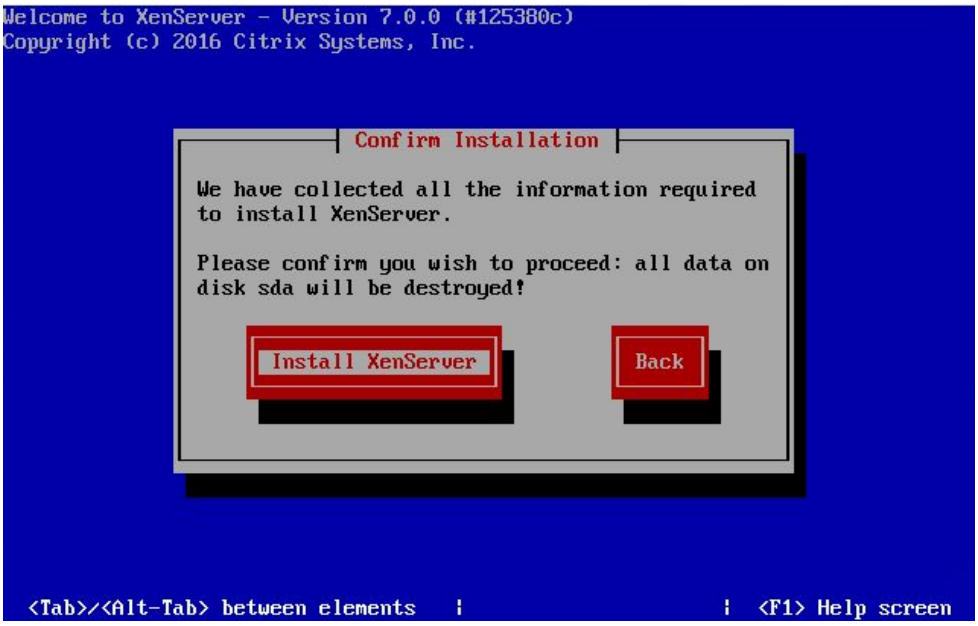


Фиг. 3.5 Избор на географска зона

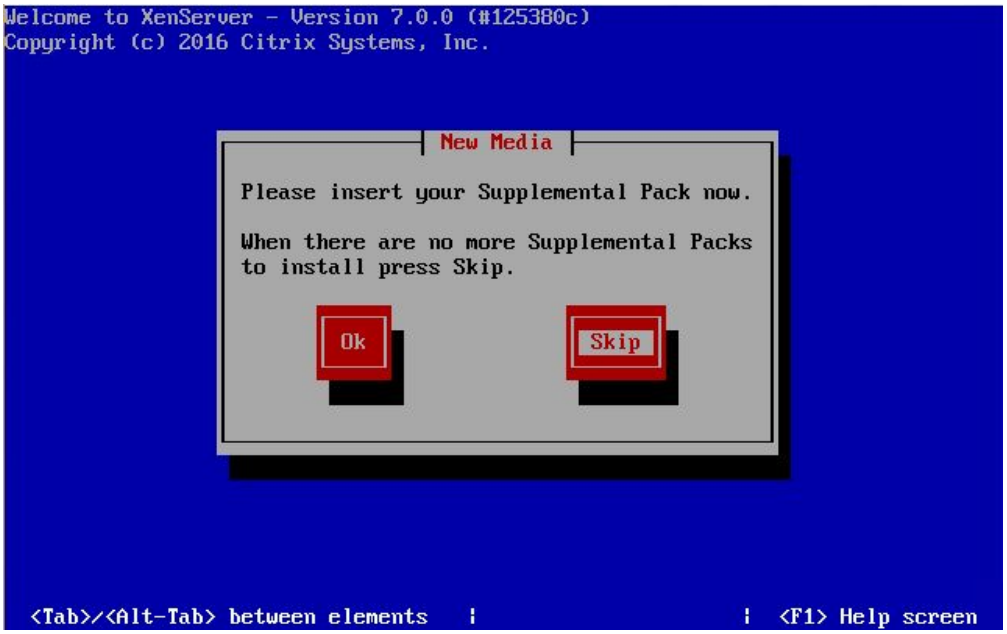
Фиг. 3.6 Избор на град



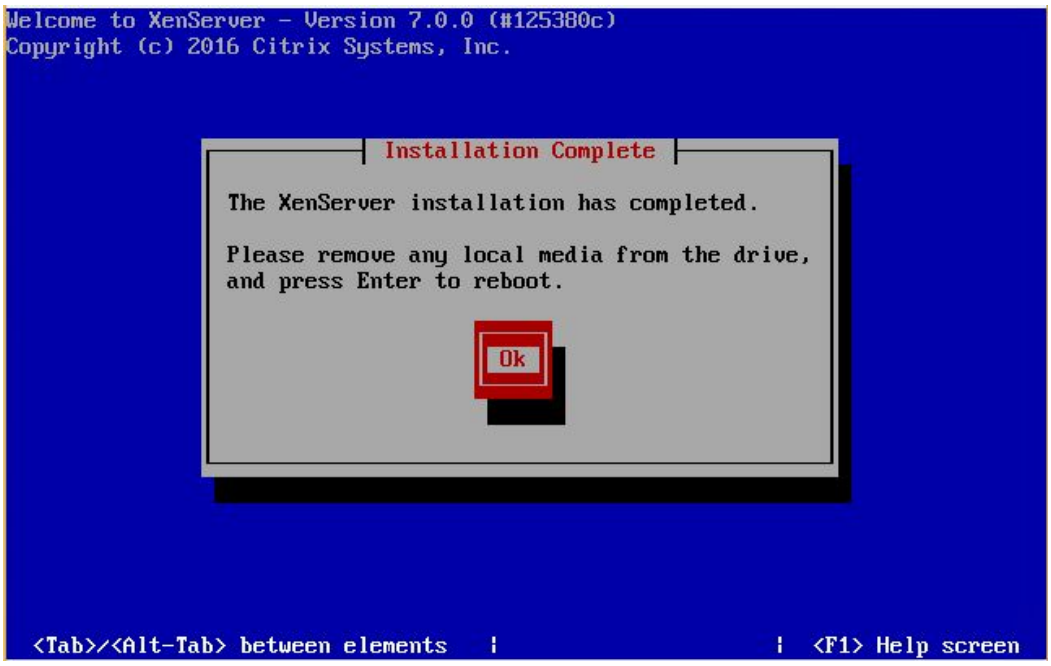
Фиг. 3.7 Конфигуриране на часова зона, която ще бъде взета по DHCP заедно с реалния адрес на машината



Фиг. 3.8 След като всичко е конфигурирано се продължава към самата инсталация



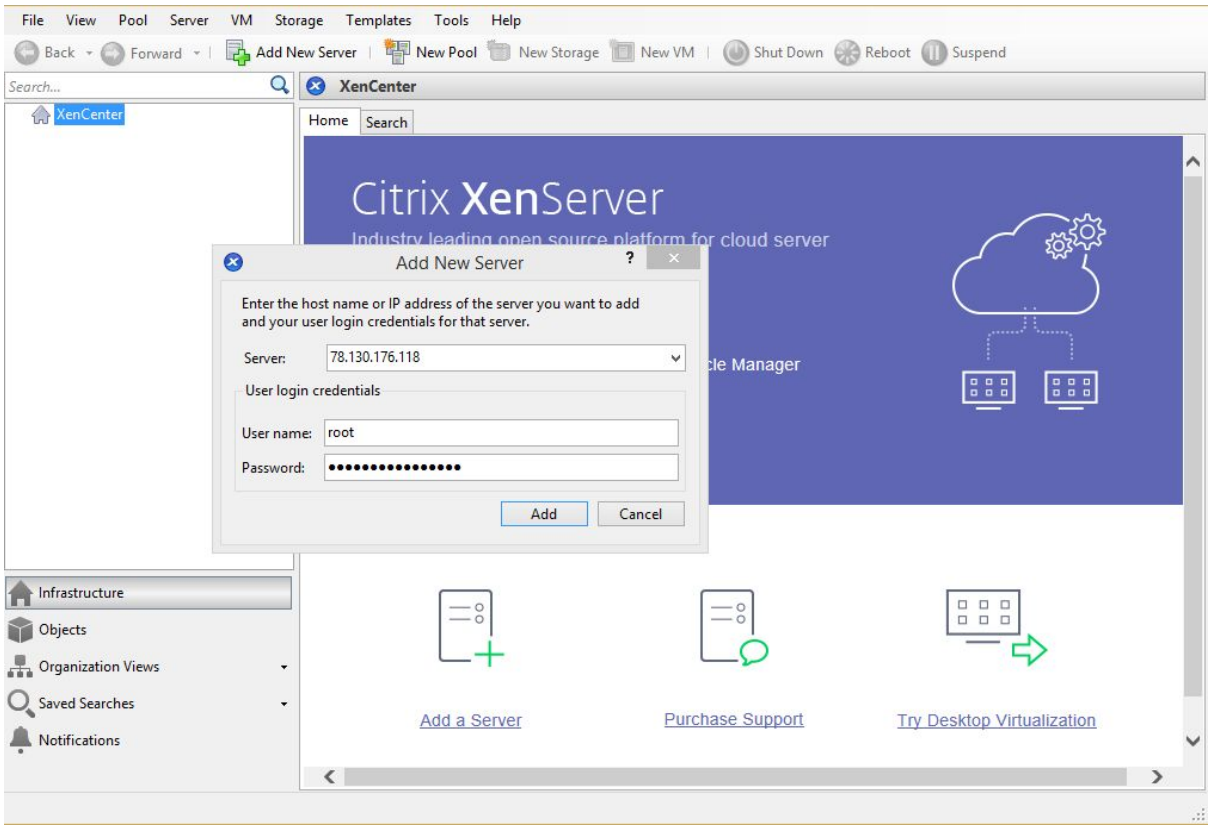
Фиг. 3.9 Избор на допълнителни пакети



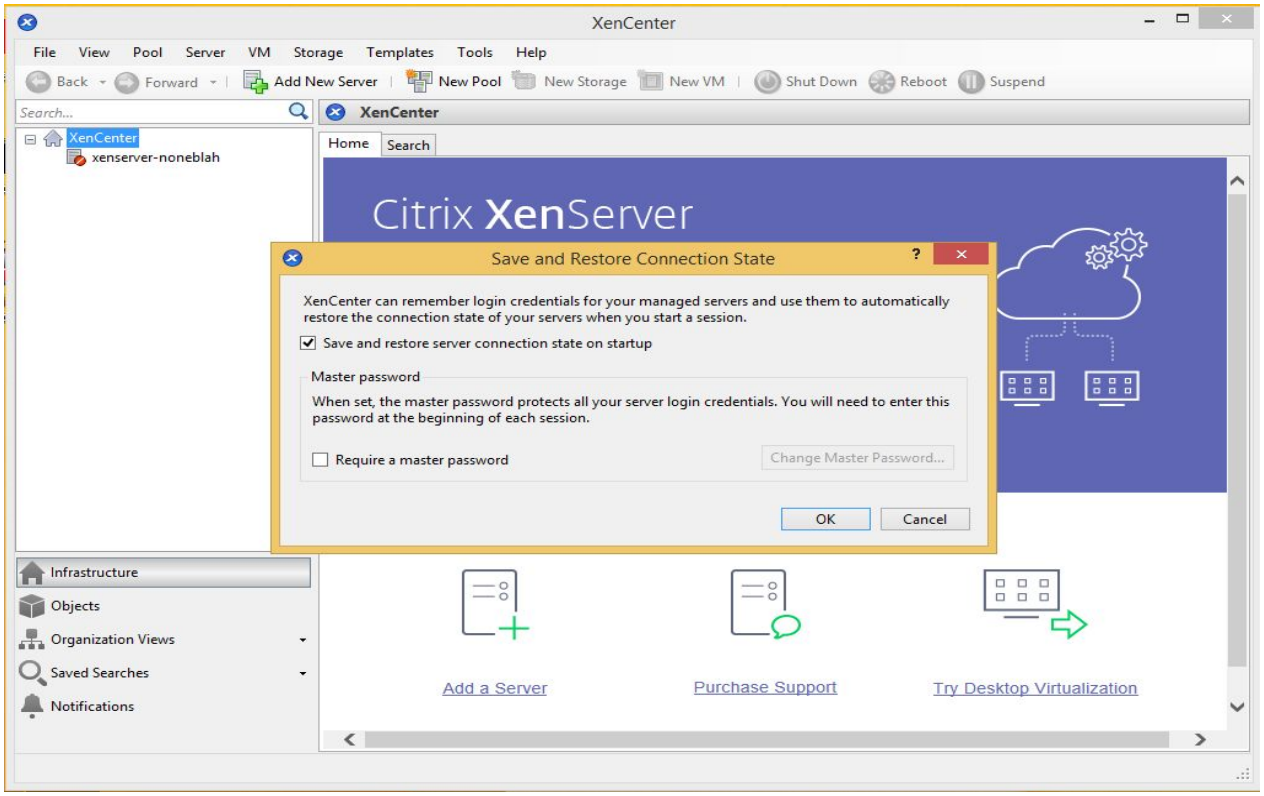
Фиг. 3.10 Съобщение, че инсталацията на XenServer 7.0 е готова

Фиг. 3.11 Процеса на зареждане на Citrix XenServer

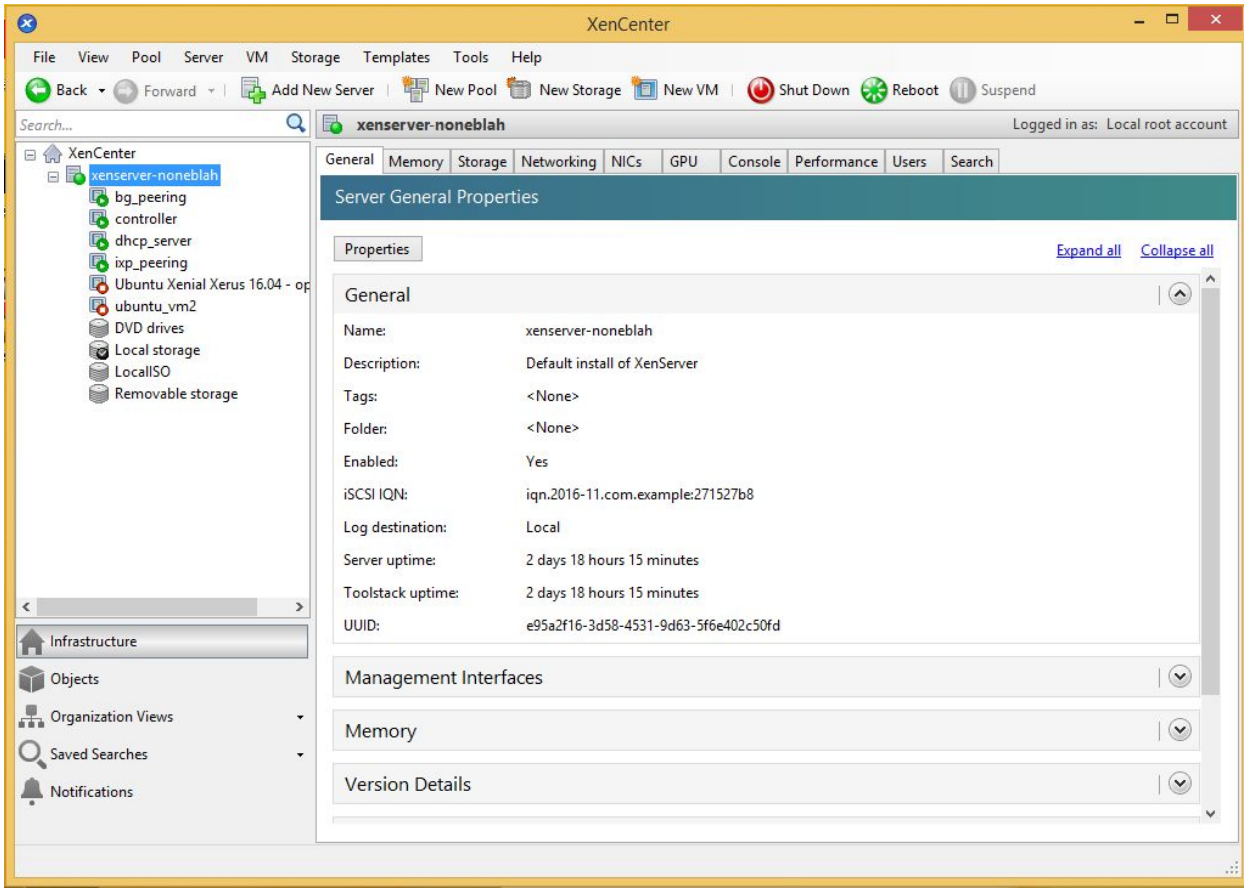
За управление на Xen хипервайзор се използва XenCenter под Windows, под Linux може да се използва OpenXenCenter, но той не работи стабилно. За целите на дипломната работа, XenCenter ще бъде инсталиран.



Фиг 3.12 Конфигурация на Citrix XenCenter



Фиг. 3.13 Запазване на потребител и парола



Фиг. 3.14 Среда за менажиране на виртуални машини в XenCenter

3.2. Конфигурация на виртуален комутатор OpenvSwitch (OVS)

3.2.1. Вътрешен OVS за XEN host

Това е вътрешен комутатор за самия Xen host, той не е основния свързващ виртуален комутатор на IXP и неговите клиенти. Xen version 7, по

подразбиране има инсталиран OpenVSwitch, но трябва да го зададем като комутатор по подразбиране. За целта се използват следните команди:

```
vif.default.script="vif-openvswitch"
```

се добавя в xl.conf файла, което прави OVS комутатора по подразбиране. След това се изпълнява следната команда, която ще стартира OVS:

```
xe-switch-network-backend openvswitch
```

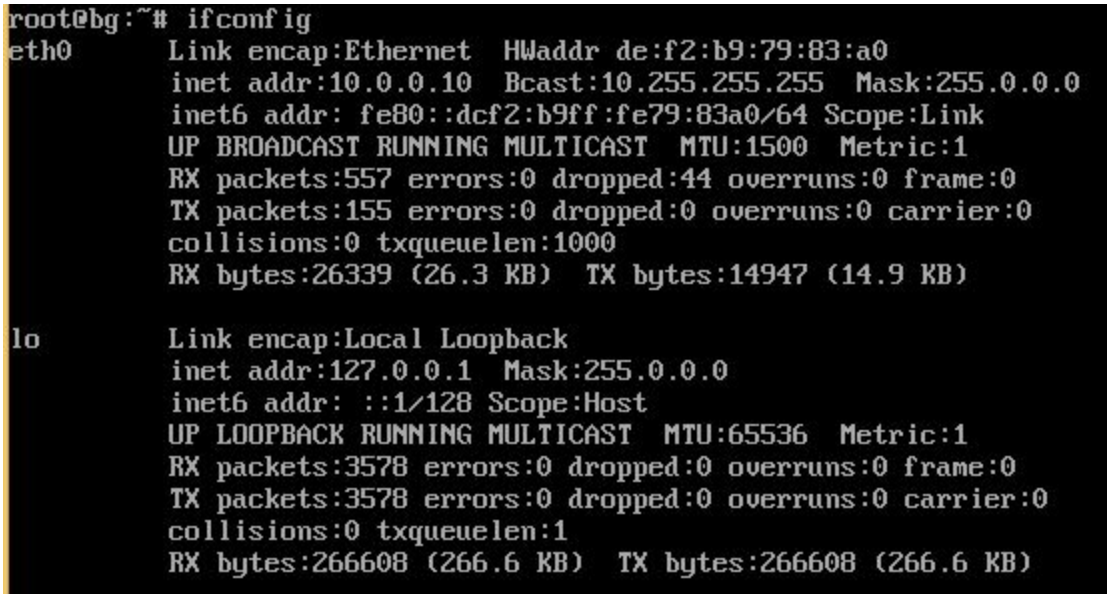
Порт eht0 е конфигуриран по подразбиране от Xen в OVS - xenbr0. На него са свързани и другите и двете виртуални машини - bg-peering и ixp-peering.
Веднъж създадени виртуалните машина и се създава виртуален интерфейс на хоста. Резултат от командата, ovs-vsctl show изпълнена на хоста:

Фиг. 3.15. OVS интерфейси на Xen host

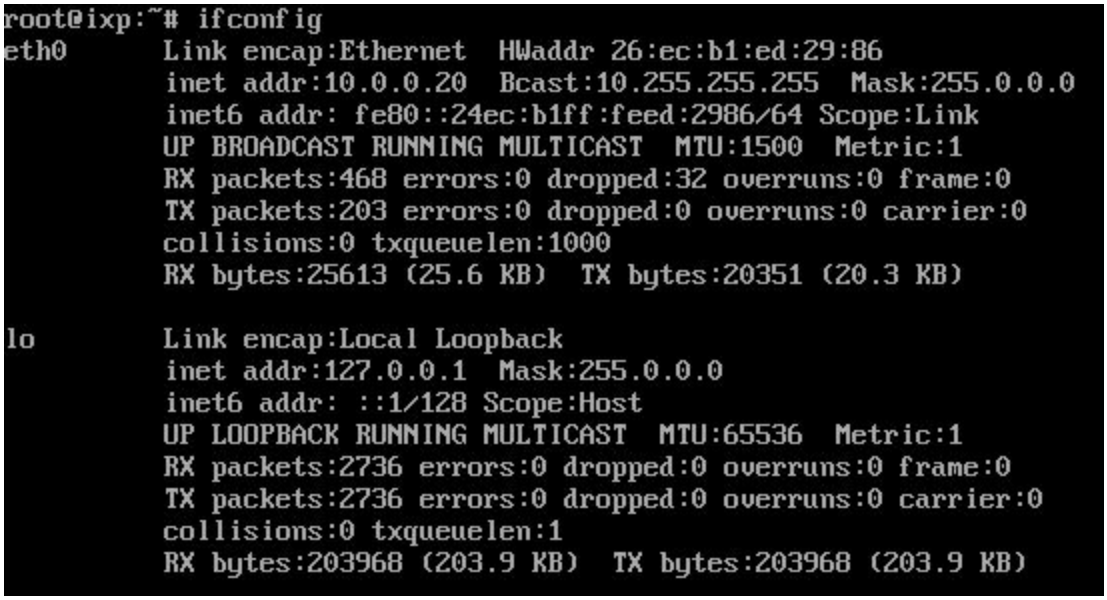
За BG peering и Exchange peering задаваме статични IP адреси, съответно:

bg_peering: 10.0.0.10/8 на eth0

Ixp_peering: 10.0.0.20/8 на eth0

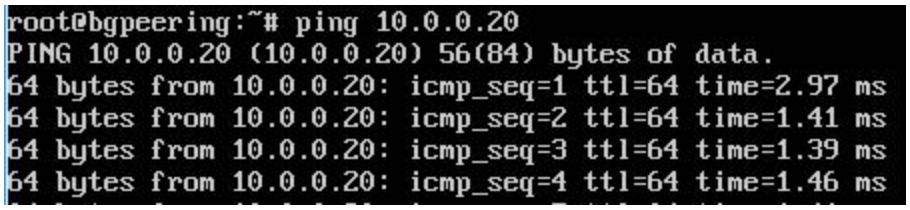


Фиг. 3.16. Резултат от ifconfig, командата на bg_peering.

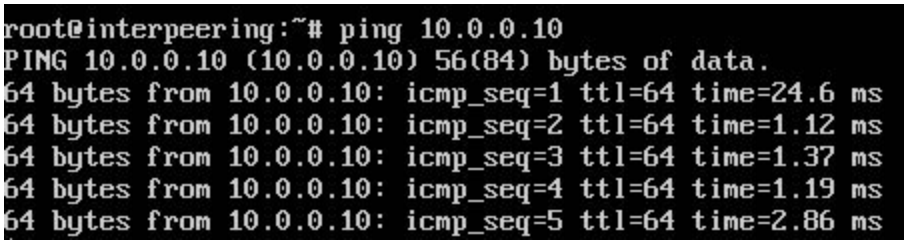


Фиг. 3.17. Резултат от командата ifconfig на ixp_peering.

Демонстрация на свързаност между устройствата в моментната конфигурация.



Фиг. 3.18. Ping свързаност от bg_peering до ixp_peering.



Фиг. 3.19 Ping свързаност от ixp_peering до bg_peering.

3.2.2. IXP OVS

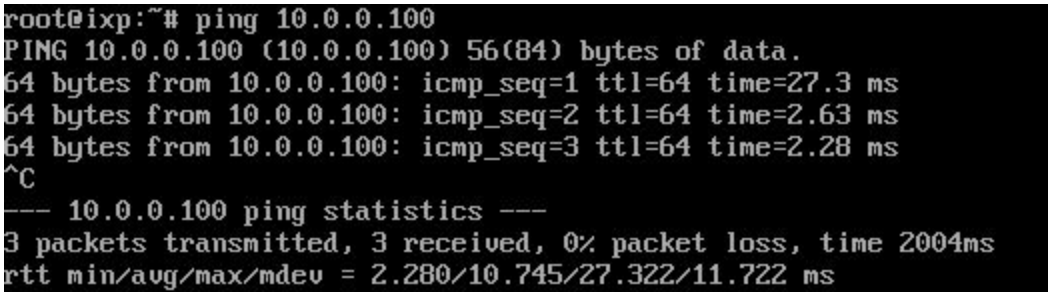
Това е основният свързващ комутатор между вътрешната мрежа на IXP и клиентите, с общ капацитет 80Gbps. Създадена е на виртуална CentOS 7 машина:

```
ovs-vsctl add-br ovs_br
```

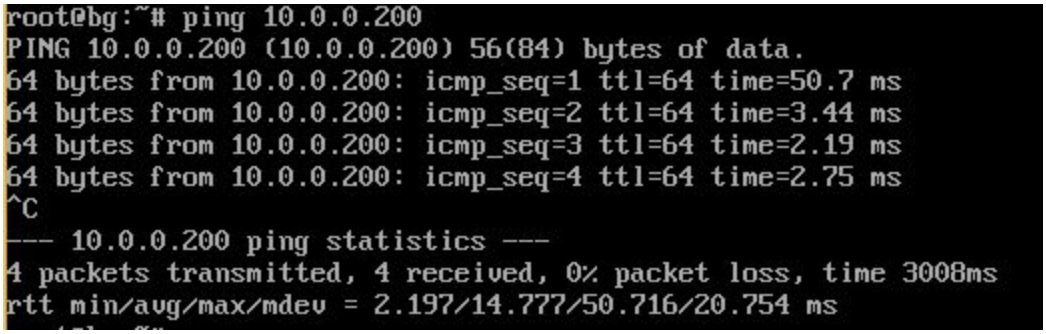
Добавят се портове към които са свързани клиентите и IXP в OVS:

```
ovs-vsctl add-port ovs_br eth1
ovs-vsctl add-port ovs_br eth2
ovs-vsctl add-port ovs_br eth3
ovs-vsctl add-port ovs_br eth4
ovs-vsctl add-port ovs_br eth5
ovs-vsctl add-port ovs_br eth6
```

Свързва се eth0 от Xen хост, се свързва директно към eth6 на OVS от IXP. Чрез тази свързност имат връзка към всички останали машини в мрежата 10.0.0.0/8 свързани в OVS.



Фиг. 3.20. Свързаност между IXP и Client 1



Фиг. 3.21. Свързаност между BG peering и Client 2

Щом контролера е инсталиран се добавя:

```
ovs-vsctl set-controller ovs_br tcp:ipНаКонтролера:6653
```

3.3. Инсталация и конфигурация на OpenFlow софтуерен контролер

3.3.1. Инсталация на контролера

Openflow контролера, Floodlight ще бъде инсталиран на машина с Ubuntu 16.04.

Обновяват се пакетите на операционната система:

```
apt-get update
```

Floodlight е разработен на Java, за това трябва да се инсталират нужните среди за разработка - JDK 8:

```
apt-get install build-essential ant maven python-dev
```

За поддръжката на по-ниски версии от Floodlight 1.2 надолу ни е нужен JDK 7:

```
apt-get install build-essential openjdk-7-jdk ant maven
python-dev eclipse
```

Floodlight трябва да бъде клониран от публичното хранилище в github на Floodlight project:

```
git clone git://github.com/floodlight/floodlight.git
```

Влиза се във вече създадената директория за Floodlight:

```
cd floodlight
```

Клонират се и нужните модули за web интерфейса:

```
git submodule init
git submodule update
```

Компилира се Java приложението:

```
ant
```

Създава се нова директория за Floodlight в `/var/lib`:

```
mkdir /var/lib/floodlight
chmod 777 /var/lib/floodlight
```

Стартира се контролера, като се пуска под екран и ще записва информация за работата си във файл floodlight.log:

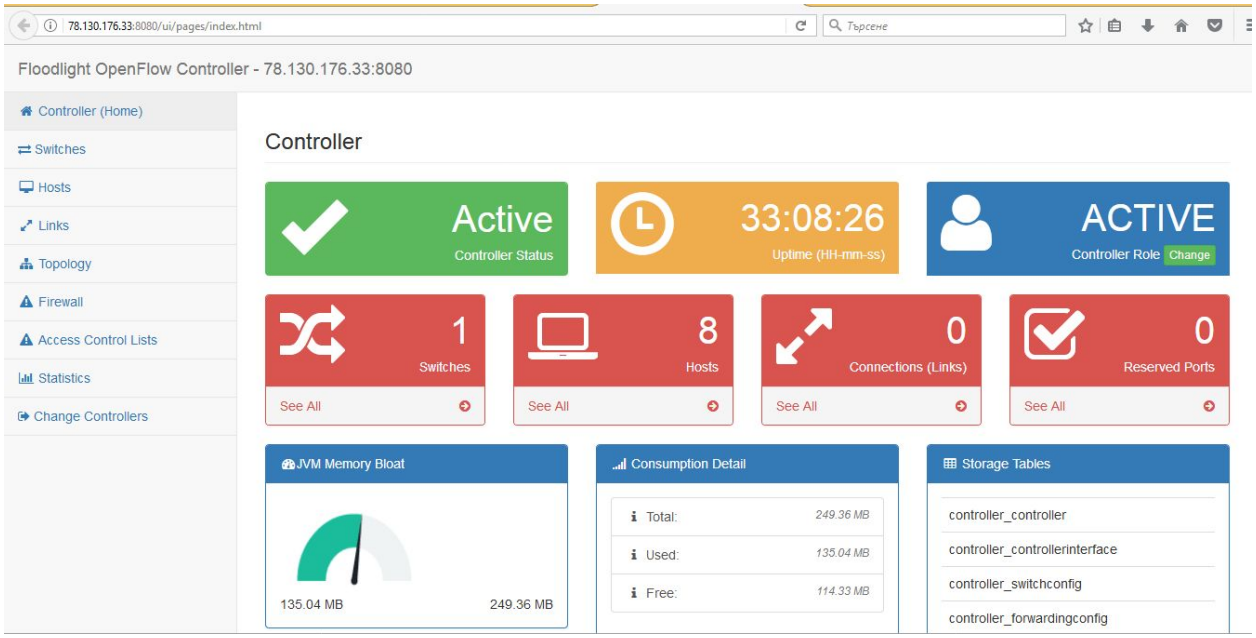
```
java -jar target/floodlight.jar > floodlight.log &&
```

WEB интерфейса му е достъпен на публичния адрес:

```
78.130.176.33:8080/ui/pages/index.html
```

Добавя се следната конфигурация на IXP OVS:

```
ovs-vsctl set-controller ovs_br tcp:78.130.176.53:6653
```



Фиг. 3.22. Началната страница на контролера

3.3.2. Създаване на Firewall правила

Създадени са правила за следните връзки:

- BG exchange - ще усъществява мрежовия обмен на български мрежи
- IXP exchange - ще усъществява мрежови обмен на чуждестрани мрежи
- Client 1 (Mikrotik) - връзка към BG exchange и директна свързаност към ISP 2
- Client 2 - връзка към IXP exchange
- Client 3 - връзка към BG exchange и директна свързаност към ISP 1
- ISP 1 - връзка към IXP exchange и Client 3
- ISP 2 - връзка към IXP exchange и Client 1

Проверява се състоянието на firewall:

```
curl http://localhost:8080/wm/firewall/module/status/json
```

Разрешава се работата на firewall:

```
curl http://localhost:8080/wm/firewall/module/enable/json
```

Нужно е да се създадат по четири двойки правила за всяка желана връзка:

- Разрешаващо правило за IPv4 адреси по TCP - ограниченията ще бъдат въведени по начален MAC адрес и дистинтационен такъв, за избягване на MAC spoofing, и начален порт адрес
- Разрешаващо правило за IPv4 адреси по UDP - ограниченията ще бъдат въведени по начален MAC адрес и дистинтационен такъв, за избягване на MAC spoofing, и начален порт адрес
- Разрешаващо правило за ARP протокол за откриване на MAC адрес за даден IP адрес - ограниченията ще бъдат въведени по начален MAC адрес и начален порт адрес, дистанционен MAC адрес няма да бъде сложен поради факта, че ARP протокола е broadcast.
- Разрешаващо правило за ICMP протокол - ограниченията ще бъдат въведени по начален MAC адрес и дистинтационен такъв, за избягване на MAC spoofing, и начален порт адрес

Формата на едно Firewall правило е както следва:

- Switchid - уникалното ID на комутатора
- src-inport - начален порт адрес
- src-mac - начален MAC адрес
- dst-mac - дистанционен MAC адрес
- dst-inport - дистанционен порт адрес
- dl-type - вид на протокола
- nw-proto - транспортния прокол по който IPv4/IPv6/ICMP ще минава
- priority - приоритетност на правилото, по-ниският е с по-високата приоритетност
- dst-ip - дистанционен IP адрес
- src-ip - начален IP адрес
- action - по какъв начин се процедира с дадения пакет

Адрес на контролера към който да бъде изпратен в JSON формат през командния ред на контролера:

```
http://localhost:8080/wm/firewall/rules/json
```

За да може да се въведат нужните правила в правилния комутатор се проверя неговото ID в страница switches, което ще бъде изпозлвано в създаване на Firewall правила и Openflow потоци:

```
00:00:00:15:5d:58:03:30
```

За целите на дипломната работа е въведено първо забраняващо връзките правило, като след това ще бъдат разрешени само нужните връзки:

```
curl -X POST -d '{"switchid":
"00:00:00:15:5d:58:03:30","action":"DENY"}'
http://localhost:8080/wm/firewall/rules/json
```

Разрешаващо правило от Client1 до BG peering - Client 1 MAC:00:15:5d:58:03:15; src-port: 3 -> BG peering MAC:de:f2:b9:79:83:a0; src-port: 2 :

```
>>>>TCP+IPv4<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"3","src-mac" :"00:15:5d:58:03:15","dst-mac"
:"de:f2:b9:79:83:a0","dl-type":"IPV4","nw-proto": "TCP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"de:f2:b9:79:83:a0","dst-mac"
:"00:15:5d:58:03:15","dl-type":"IPV4","nw-proto": "TCP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
```

```
>>>>UDP+IPv4<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"3","src-mac" :"00:15:5d:58:03:15","dst-mac"
:"de:f2:b9:79:83:a0","dl-type":"IPV4","nw-proto": "UDP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"de:f2:b9:79:83:a0","dst-mac"
:"00:15:5d:58:03:15","dl-type":"IPV4","nw-proto": "UDP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json


>>>>ARP<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"3","src-mac" :"00:15:5d:58:03:15","dl-type":"ARP","tp-src" :
"","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
```

```
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"de:f2:b9:79:83:a0","dl-type":"ARP","tp-src" :
"","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json

>>>>ICMP<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"3","src-mac" :"00:15:5d:58:03:15","dst-mac"
:"de:f2:b9:79:83:a0","nw-proto": "ICMP","tp-src" : "","tp-dst":
"","priority": "-2","dst-ip": "","src-ip": "","action":
"ALLOW"}' http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"de:f2:b9:79:83:a0","dst-mac"
:"00:15:5d:58:03:15","nw-proto": "ICMP","tp-src" : "","tp-dst":
"","priority": "-2","dst-ip": "","src-ip": "","action":
"ALLOW"}' http://localhost:8080/wm/firewall/rules/json
```

Разрешаващо правило от Client3 до BG peering - Client 3 MAC:00:15:5d:58:03:1e; src-port: 5 -> BG peering MAC:de:f2:b9:79:83:a0; src-port: 2 :

```
>>>>TCP+IPv4<<<<
```

```
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"5","src-mac" :"00:15:5d:58:03:1e","dst-mac"
:"de:f2:b9:79:83:a0","dl-type":"IPV4","nw-proto": "TCP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"de:f2:b9:79:83:a0","dst-mac"
:"00:15:5d:58:03:1e","dl-type":"IPV4","nw-proto": "TCP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
```

```
>>>>UDP+IPv4<<<<
```

```
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"5","src-mac" :"00:15:5d:58:03:1e","dst-mac"
:"de:f2:b9:79:83:a0","dl-type":"IPV4","nw-proto": "UDP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"de:f2:b9:79:83:a0","dst-mac"
:"00:15:5d:58:03:1e","dl-type":"IPV4","nw-proto": "UDP","tp-src"
```

```
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json


>>>>ARP<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"5","src-mac" :"00:15:5d:58:03:1e","dl-type":"ARP","tp-src" :
"","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"de:f2:b9:79:83:a0","dl-type":"ARP","tp-src" :
"","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json

>>>>ICMP<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"5","src-mac" :"00:15:5d:58:03:1e","dst-mac"
:"de:f2:b9:79:83:a0","nw-proto": "ICMP","tp-src" : "","tp-dst":
"","priority": "-2","dst-ip": "","src-ip": "","action":
"ALLOW"}' http://localhost:8080/wm/firewall/rules/json
```

```
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"de:f2:b9:79:83:a0","dst-mac"
:"00:15:5d:58:03:1e","nw-proto": "ICMP","tp-src" : "","tp-dst":
"","priority": "-2","dst-ip": "","src-ip": "","action":
"ALLOW"}' http://localhost:8080/wm/firewall/rules/json
```

Разрешаващо правило от Client 2 до IXP - client3 MAC:00:15:5d:58:03:1d; src-port: 4 -> ixp MAC:26:ec:b1:ed:29:86; src-port: 2 :

```
>>>>TCP+IPv4<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"4","src-mac" :"00:15:5d:58:03:1d","dst-mac"
:"26:ec:b1:ed:29:86","dl-type":"IPV4","nw-proto": "TCP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"26:ec:b1:ed:29:86","dst-mac"
:"00:15:5d:58:03:1d","dl-type":"IPV4","nw-proto": "TCP","tp-src"
```

```
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json


>>>>UDP+IPv4<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"4","src-mac" :"00:15:5d:58:03:1d","dst-mac"
:"26:ec:b1:ed:29:86","dl-type":"IPV4","nw-proto": "UDP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"26:ec:b1:ed:29:86","dst-mac"
:"00:15:5d:58:03:1d","dl-type":"IPV4","nw-proto": "UDP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json


>>>>ARP<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"4","src-mac" :"00:15:5d:58:03:1d","dl-type":"ARP","tp-src" :
"","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
```

```
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"26:ec:b1:ed:29:86","dl-type":"ARP","tp-src" :
"","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
```

```
>>>>ICMP<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"4","src-mac" :"00:15:5d:58:03:1d","dst-mac"
:"26:ec:b1:ed:29:86","nw-proto": "ICMP","tp-src" : "","tp-dst":
"","priority": "-2","dst-ip": "","src-ip": "","action":
"ALLOW"}' http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"26:ec:b1:ed:29:86","dst-mac"
:"00:15:5d:58:03:1d","nw-proto": "ICMP","tp-src" : "","tp-dst":
"","priority": "-2","dst-ip": "","src-ip": "","action":
"ALLOW"}' http://localhost:8080/wm/firewall/rules/json
```

Разрешаващо правило от ISP1 до IXP - ISP1 MAC:00:15:5d:58:03:1f; src-port: 1 -> IXP MAC:26:ec:b1:ed:29:86; src-port: 2 :

```
>>>>TCP+IPv4<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"1","src-mac" :"00:15:5d:58:03:1f","dst-mac"
:"26:ec:b1:ed:29:86","dl-type":"IPV4","nw-proto": "TCP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"26:ec:b1:ed:29:86","dst-mac"
:"00:15:5d:58:03:1f","dl-type":"IPV4","nw-proto": "TCP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json


>>>>UDP+IPv4<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"1","src-mac" :"00:15:5d:58:03:1f","dst-mac"
:"26:ec:b1:ed:29:86","dl-type":"IPV4","nw-proto": "UDP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
```

```
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"26:ec:b1:ed:29:86","dst-mac"
:"00:15:5d:58:03:1f","dl-type":"IPV4","nw-proto": "UDP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
```

```
>>>>ARP<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"1","src-mac" :"00:15:5d:58:03:1f","dl-type":"ARP","tp-src" :
"","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"26:ec:b1:ed:29:86","dl-type":"ARP","tp-src" :
"","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json

>>>>ICMP<<<<
```

```
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"1","src-mac" :"00:15:5d:58:03:1f","dst-mac"
:"26:ec:b1:ed:29:86","nw-proto": "ICMP","tp-src" : "","tp-dst":
"","priority": "-2","dst-ip": "","src-ip": "","action":
"ALLOW"}' http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"26:ec:b1:ed:29:86","dst-mac"
:"00:15:5d:58:03:1f","nw-proto": "ICMP","tp-src" : "","tp-dst":
"","priority": "-2","dst-ip": "","src-ip": "","action":
"ALLOW"}' http://localhost:8080/wm/firewall/rules/json
```

Разрешаващо правило от ISP2 до IXP - ISP2 MAC:00:15:5d:58:03:20; src-port: 6 -> IXP MAC:26:ec:b1:ed:29:86; src-port: 2 :

```
>>>>TCP+IPv4<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"6","src-mac" :"00:15:5d:58:03:20","dst-mac"
:"26:ec:b1:ed:29:86","dl-type":"IPV4","nw-proto": "TCP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
```

```
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"26:ec:b1:ed:29:86","dst-mac"
:"00:15:5d:58:03:20","dl-type":"IPV4","nw-proto": "TCP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json


>>>>UDP+IPv4<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"6","src-mac" :"00:15:5d:58:03:20","dst-mac"
:"26:ec:b1:ed:29:86","dl-type":"IPV4","nw-proto": "UDP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"26:ec:b1:ed:29:86","dst-mac"
:"00:15:5d:58:03:20","dl-type":"IPV4","nw-proto": "UDP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json


>>>>ARP<<<<
```

```
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"6","src-mac" :"00:15:5d:58:03:20","dl-type":"ARP","tp-src" :
"","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"26:ec:b1:ed:29:86","dl-type":"ARP","tp-src" :
"","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json

>>>>ICMP<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"6","src-mac" :"00:15:5d:58:03:20","dst-mac"
:"26:ec:b1:ed:29:86","nw-proto": "ICMP","tp-src" : "","tp-dst":
"","priority": "-2","dst-ip": "","src-ip": "","action":
"ALLOW"}' http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"2","src-mac" :"26:ec:b1:ed:29:86","dst-mac"
:"00:15:5d:58:03:20","nw-proto": "ICMP","tp-src" : "","tp-dst":
"","priority": "-2","dst-ip": "","src-ip": "","action":
"ALLOW"}' http://localhost:8080/wm/firewall/rules/json
```

Разрешаващо правило от Client 3 до ISP1 - client3 MAC:00:15:5d:58:03:1e; src-port: 5 -> isp1 MAC:00:15:5d:58:03:1f; src-port: 1 :

```
>>>>TCP+IPv4<<<<
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"5","src-mac" :"00:15:5d:58:03:1e","dst-mac"
:"00:15:5d:58:03:1f","dl-type":"IPV4","nw-proto": "TCP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"1","src-mac" :"00:15:5d:58:03:1f","dst-mac"
:"00:15:5d:58:03:1e","dl-type":"IPV4","nw-proto": "TCP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json


>>>>UDP+IPv4<<<<
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"5","src-mac" :"00:15:5d:58:03:1e","dst-mac"
:"00:15:5d:58:03:1f","dl-type":"IPV4","nw-proto": "UDP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
```

```
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"1","src-mac" :"00:15:5d:58:03:1f","dst-mac"
:"00:15:5d:58:03:1e","dl-type":"IPV4","nw-proto": "UDP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json


>>>>ARP<<<<
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"5","src-mac" :"00:15:5d:58:03:1e","dl-type":"ARP","tp-src" :
"","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"1","src-mac" :"00:15:5d:58:03:1f","dl-type":"ARP","tp-src" :
"","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json

>>>>ICMP<<<<
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"5","src-mac" :"00:15:5d:58:03:1e","dst-mac"
:"00:15:5d:58:03:1f","nw-proto": "ICMP","tp-src" : "","tp-dst":
```

```
"","priority": "-2","dst-ip": "","src-ip": "","action":
"ALLOW"}' http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"1","src-mac" :"00:15:5d:58:03:1f","dst-mac"
:"00:15:5d:58:03:1e","nw-proto": "ICMP","tp-src" : "","tp-dst":
"","priority": "-2","dst-ip": "","src-ip": "","action":
"ALLOW"}' http://localhost:8080/wm/firewall/rules/json
```

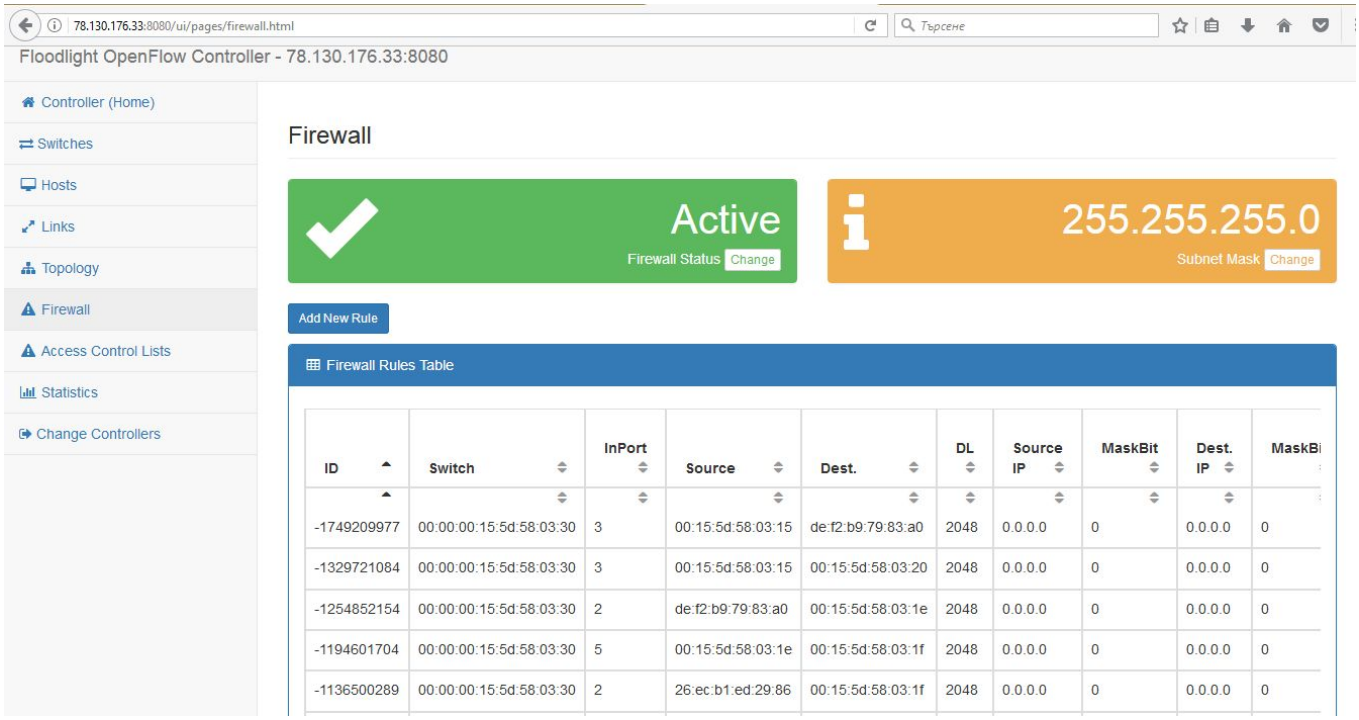
Разрешаващо правило от Client 1 до ISP 2 - client1 MAC:00:15:5d:58:03:15; src-port: 3 -> ISP 2 MAC:00:15:5d:58:03:20; src-port: 6 :

```
>>>>TCP+IPv4<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"3","src-mac" :"00:15:5d:58:03:15","dst-mac"
:"00:15:5d:58:03:20","dl-type":"IPV4","nw-proto": "TCP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"6","src-mac" :"00:15:5d:58:03:20","dst-mac"
```

```
:"00:15:5d:58:03:15","dl-type":"IPV4","nw-proto": "TCP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json


>>>>UDP+IPv4-w<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"3","src-mac" :"00:15:5d:58:03:15","dst-mac"
:"00:15:5d:58:03:20","dl-type":"IPV4","nw-proto": "UDP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"6","src-mac" :"00:15:5d:58:03:20","dst-mac"
:"00:15:5d:58:03:15","dl-type":"IPV4","nw-proto": "UDP","tp-src"
: "","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json


>>>>ARP<<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"3","src-mac" :"00:15:5d:58:03:15","dl-type":"ARP","tp-src" :
```

```
"","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"6","src-mac" :"00:15:5d:58:03:20","dl-type":"ARP","tp-src" :
"","tp-dst": "","priority": "-2","dst-ip": "","src-ip":
"","action": "ALLOW"}'
http://localhost:8080/wm/firewall/rules/json
```

```
>>>>ICMP<<<<

curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"3","src-mac" :"00:15:5d:58:03:15","dst-mac"
:"00:15:5d:58:03:20","nw-proto": "ICMP","tp-src" : "","tp-dst":
"","priority": "-2","dst-ip": "","src-ip": "","action":
"ALLOW"}' http://localhost:8080/wm/firewall/rules/json
curl -X POST -d
'{"switchid":"00:00:00:15:5d:58:03:30","src-inport"
:"6","src-mac" :"00:15:5d:58:03:20","dst-mac"
:"00:15:5d:58:03:15","nw-proto": "ICMP","tp-src" : "","tp-dst":
"","priority": "-2","dst-ip": "","src-ip": "","action":
"ALLOW"}' http://localhost:8080/wm/firewall/rules/json
```



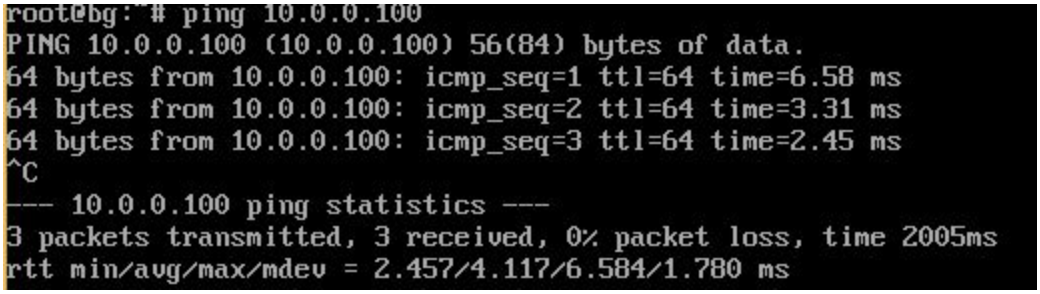
Фиг. 3.23. Firewall правила през WEB интерфейс

Проверява се работата на firewall правилата:

В сравнение с фиг. 3.21:

```
root@bg:~# ping 10.0.0.200
PING 10.0.0.200 (10.0.0.200) 56(84) bytes of data.
^C
--- 10.0.0.200 ping statistics ---
6 packets transmitted, 0 received, 100% packet loss, time 5010ms
```

Фиг. 3.24 Връзката на свързаност между BG peering и Client 2



Фиг.3.25 Свързаност между BG peering и Client 1

3.4. Конфигурация на BGP маршрутизатор

За реализарането на мрежови обмен ще бъде инсталиран и конфигуриран Quagga пакет, чрез който ще се конфигурира BGP рутер на следните устройства:

- BG peering, IXP peering, ISP1, ISP2,Client2 и Client3

За конфигурирането на BGP рутер се използват виртуални машини:

- Client 1 - Mikrotik Cloud Router
- Client 2, Client 3, IXP 1 и IXP 2 - Ubuntu 16.04 OS + Quagga

Конфигурирането на Quagga сървиз се усъществява по следния начин:

Актуализарат се пакетите на операционната система:

```
apt-get update
```

Инсталира се Quagga сървиз:

```
apt-get install quagga
```

Трябва да се разреши със сигурност IPv4 и IPv6 рутиране на пакети:

```
echo "net.ipv4.conf.all.forwarding=1" | sudo tee -a
/etc/sysctl.conf
```

резултат: `net.ipv4.conf.all.forwarding=1`

```
echo "net.ipv4.conf.default.forwarding=1" | sudo tee -a
/etc/sysctl.conf
```

резултат: `net.ipv4.conf.default.forwarding=1`

```
sed's/#net.ipv6.conf.all.forwarding=1/net.ipv6.conf.all.forwardi
ng=1/g' /etc/sysctl.conf | sudo tee /etc/sysctl.conf

echo "net.ipv6.conf.default.forwarding=1" | sudo tee -a
/etc/sysctl.conf

sysctl -p
```

Създават се нужните конфигурационни файлове, които служат за различните видове протокли, които quagga може да рутира:

```
nano /etc/quagga/babeld.conf
nano /etc/quagga/bgpd.conf
nano /etc/quagga/isisd.conf
nano /etc/quagga/ospf6d.conf
nano /etc/quagga/ospfd.conf
nano /etc/quagga/ripd.conf
nano /etc/quagga/ripngd.conf
nano /etc/quagga/vtysh.conf
nano /etc/quagga/zebra.conf
```

За целта на дипломната работа ще бъде разгледан /etc/quagga/bgpd.conf и по-конкретно частта която засяга маршрутизацията на 10.0.0.0/8 мрежи, която под подразбиране е спряна :

```
access-list Internet deny 127.0.0.0/8
#       Loopback
access-list Internet permit 192.0.0.0/29
#       DS-Lite
access-list Internet permit 192.0.0.0/24
#       IETF Protocol Assignments
access-list Internet permit 192.0.2.0/24
#       TEST-NET-1
access-list Internet permit 198.51.100.0/24
#       TEST-NET-2
access-list Internet permit 203.0.113.0/24
```

```
#       TEST-NET-3
access-list Internet permit 240.0.0.0/4
#       Reserved for Future Use
access-list Internet permit any
#       RFC 6890
ipv6 access-list Internet permit 2001::/23
#       IETF Protocol Assignments
ipv6 access-list Internet permit 2001:db8::/32
#       Documentation
ipv6 access-list Internet permit 2001:10::/28
#       Overlay Routable Cryptographic Hash IDentifiers
ipv6 access-list Internet permit 2000::/3
#       RFC 6890
```

Това се променя в конфигурационните файлове за всички BGP рутери.

За всеки отделен файл опоменат по-горе трябва да се проми собствеността и пермисиите, както следва:

```
chown quagga:quagga /etc/quagga/babeld.conf && chmod 640
/etc/quagga/babeld.conf
chown quagga:quagga /etc/quagga/bgpd.conf && chmod 640
/etc/quagga/bgpd.conf
chown quagga:quagga /etc/quagga/isisd.conf && chmod 640
/etc/quagga/isisd.conf
chown quagga:quagga /etc/quagga/ospf6d.conf && chmod 640
/etc/quagga/ospf6d.conf
```

```
chown quagga:quagga /etc/quagga/ospfd.conf && chmod 640
/etc/quagga/ospfd.conf
chown quagga:quagga /etc/quagga/ripd.conf && chmod 640
/etc/quagga/ripd.conf
chown quagga:quagga /etc/quagga/ripngd.conf && chmod 640
/etc/quagga/ripngd.conf
chown quagga:quaggavty /etc/quagga/vtysh.conf && chmod 660
/etc/quagga/vtysh.conf
chown quagga:quagga /etc/quagga/zebra.conf && chmod 640
/etc/quagga/zebra.conf
```

По този начин потребител:quagga може да чете и да пише във файла, а група: quagga може да чете. Следващото което трябва да се направи е да се упоменат на quagga демона кои протоколи може да маршрутизира:

```
nano /etc/quagga/daemons
zebra=yes
bgpd=yes
ospfd=no
ospf6d=no
ripd=no
ripngd=no
isisd=no
babeld=no
```

Направени са промени в telnet конфигурационния файл на операционната система:

```
nano /etc/quagga/debian.conf
vtysh_enable=yes
zebra_options=" --daemon -A 127.0.0.1 -P 2601 -u quagga -g
quagga"
bgpd_options=" --daemon -A 127.0.0.1 -P 2605 -u quagga -g quagga
--retain -p 179"
ospfd_options=" --daemon -A 127.0.0.1 -P 2604 -u quagga -g
quagga"
ospf6d_options=" --daemon -A ::1 -P 2606 -u quagga -g quagga"
ripd_options=" --daemon -A 127.0.0.1 -P 2602 -u quagga -g
quagga"
ripngd_options=" --daemon -A ::1 -P 2603 -u quagga -g quagga"
isisd_options=" --daemon -A 127.0.0.1 -P 2608 -u quagga -g
quagga"
babeld_options=" --daemon -A 127.0.0.1 -P 2609 -u quagga -g
quagga"
```

За да влязат по-горните промени в сила, quagga приложението трябва да се рестартира:

```
/etc/init.d/quagga restart
```

За да се влезе в режим на конфигурация на BGP рутера се пише в командния ред в root режим:

```
vtysh
```

Конфигурацията на устройствата е както следва:

BG exchange:

router bgp 64600

bgp router-id 10.0.0.10

neighbor 10.0.0.100 remote-as 64700 (Client 1)

neighbor 10.0.0.250 remote-as 64900 (Client 3)

```
router bgp 64600
 bgp router-id 10.0.0.10
 neighbor 10.0.0.100 remote-as 64700
 neighbor 10.0.0.100 distribute-list 2 in
 neighbor 10.0.0.250 remote-as 64900
 neighbor 10.0.0.250 distribute-list 1 in
```

Фиг. 3.26 Конфигурацията на BG peering

IXP exchange:

router bgp 64500

bgp router-id 10.0.0.20

neighbor 10.0.0.200 remote-as 64800 (Client 2)

neighbor 10.0.0.50 remote-as 65501(ISP 1)

neighbor 10.0.0.60 remote-as 65502 (ISP2)

```
router bgp 64500
 bgp router-id 10.0.0.20
 neighbor 10.0.0.50 remote-as 65501
 neighbor 10.0.0.60 remote-as 65502
 neighbor 10.0.0.200 remote-as 64800
```

Фиг. 3.27 Конфигурация на IXP

ISP 1:

Router bgp 65501

Bgp router-id 10.0.0.50

neighbor 10.0.0.20 remote-as 64500 (IXP)

neighbor 10.0.0.250 remote-as 64900 (Client 3)

ISP 2:

Router bgp 65502

BGP router-id 10.0.0.60

neighbor 10.0.0.20 remote-as 64500 (IXP)

neighbor 10.0.0.100 remote-as 64700 (Client1)

Client 1:

router bgp 64700

bgp router-id 10.0.0.100

neighbor 10.0.0.10 remote-as 64600 (BG)

neighbor 10.0.0.60 remote-as 65502 (ISP2)

Client 2:

router bgp 64800

Bgp router-id 10.0.0.200

neighbor 10.0.0.20 remote-as 64500 (IXP)

Client 3:

Router bgp 64900

bgp router-id 10.0.0.250

neighbor 10.0.0.10 remote-as 64600(BG peering)

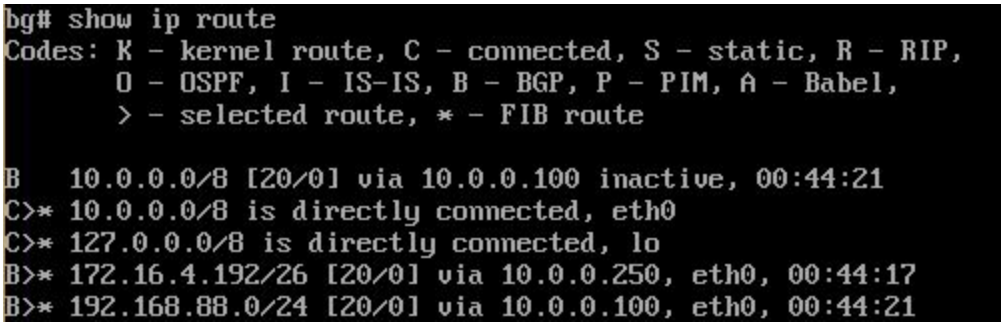
neighbor 10.0.0.50 remote-as 65501(ISP 1)

Въведени са следните distribute-list на клиентските машини, като това не е пряк предмет на дипломната работа:

ISP 2
neighbor 10.0.0.20 distribute-list 2 out
access-list 2 deny 192.168.88.0 0.0.0.255
access-list 2 permit any
C1
neighbor 10.0.0.10 distribute-list 2 out
access-list 2 deny 172.16.0.0 0.0.254.255
access-list 2 deny 172.16.2.0 0.0.254.255
access-list 2 permit any
C3
neighbor 10.0.0.10 distribute-list 2 out

access-list 2 deny 192.168.0.0 0.0.254.255
access-list 2 deny 192.168.2.0 0.0.254.255
access-list 2 permit any

Таблица 3.1 Access-list за BGP



Фиг. 3.28 Резултат от show ip route на BG peering

4. Заключение

Дипломната работа е изградена като работоспособен прототип за решение за Internet Exchange Point базирано на Софтуерно Дефинирана Мрежа. Ключовото в нея, е че чрез софтуер е постигнато управление на мрежа и доказва приложимостта на OpenFlow протокола и свързаните технологии.

По време на изграждането на демонстранционния прототип са срещнати известни затруднения с това че все още повечето софтуерни продукти не са стабилни. Това би попречило на имплементирането им в реална инфраструктура.

Въпреки това, технологичните възможности на протокола, контролера и виртуалния комутатор са обещаващи и биха довели до значително подобрение и улеснение в компютърните мрежи.

Бъдещи цели на дипломната работа включват да се подобри софтуерния контролер или да се работи по създаването на нов такъв. Идея би било да се напише на високо-производителен и високо-конкурентен език, например Erlang (често използван за комуникационни приложения).

Използвана литература:

https://tools.ietf.org/html/rfc7426

http://therandomsecurityguy.com/openvswitch-cheat-sheet/

http://www.projectfloodlight.org/documentation/

https://wiki.ubuntu.com/JonathanFerguson/Quagga

http://xmodulo.com/centos-bgp-router-quagga.html

https://www.de-cix.net/

http://openvswitch.org/

http://ericssonhistory.com/global/Ericsson%20review/Ericsson%20Review.%201988.%20V.65/PDF/Ericsson_Review_Vol_65_1988_4.pdf

https://www.opendaylight.org/

https://osrg.github.io/ryu/