Санкт-Петербургский государственный университет
телекоммуникаций им. проф. М.А. Бонч-Бруевича

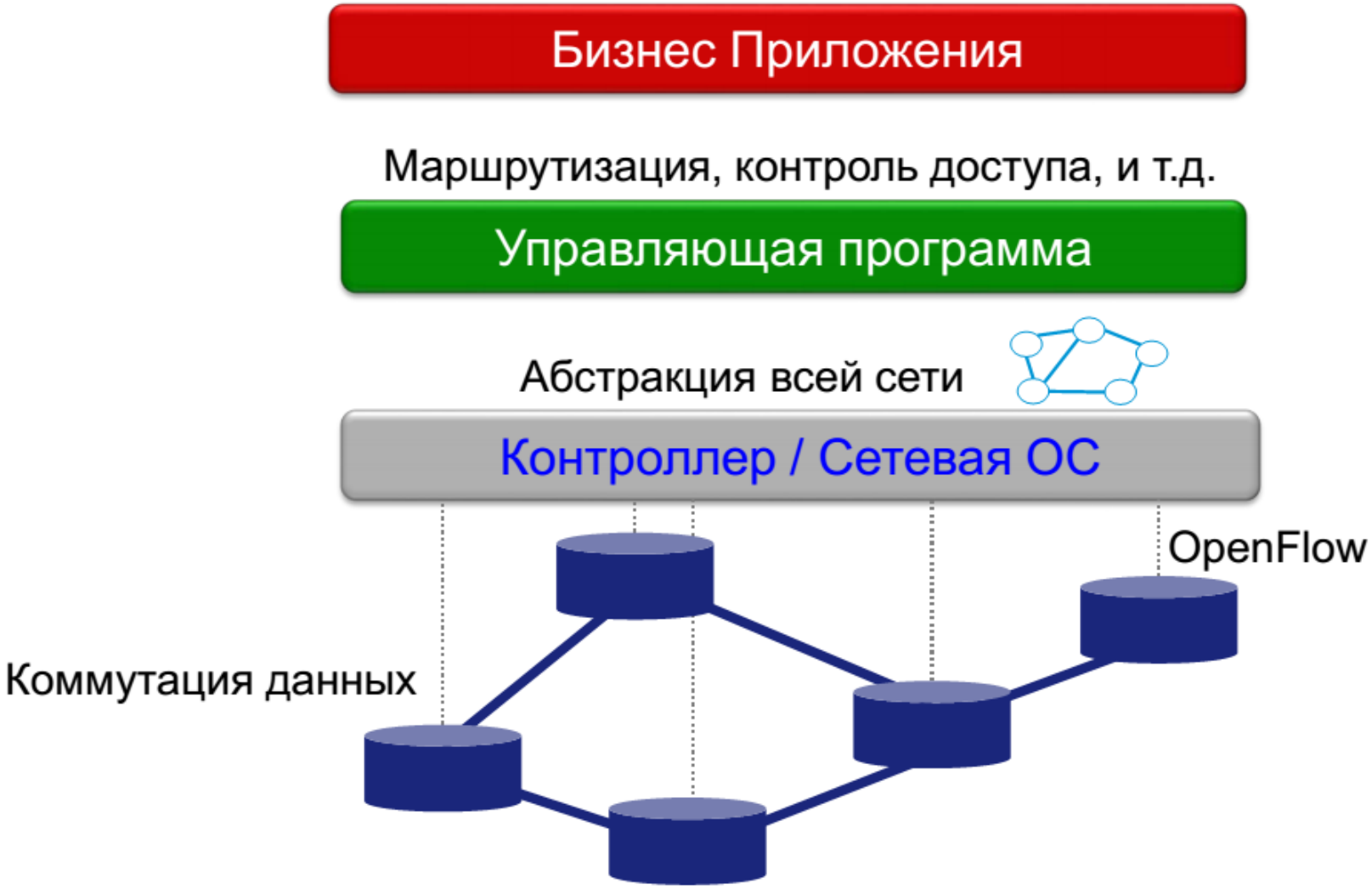
Особенности эксплуатации программно-конфигурируемых сетей

(Лекция 4)

Елагин В.С.
к.т.н. доцент каф. ИКС

СПб ГУТ)))

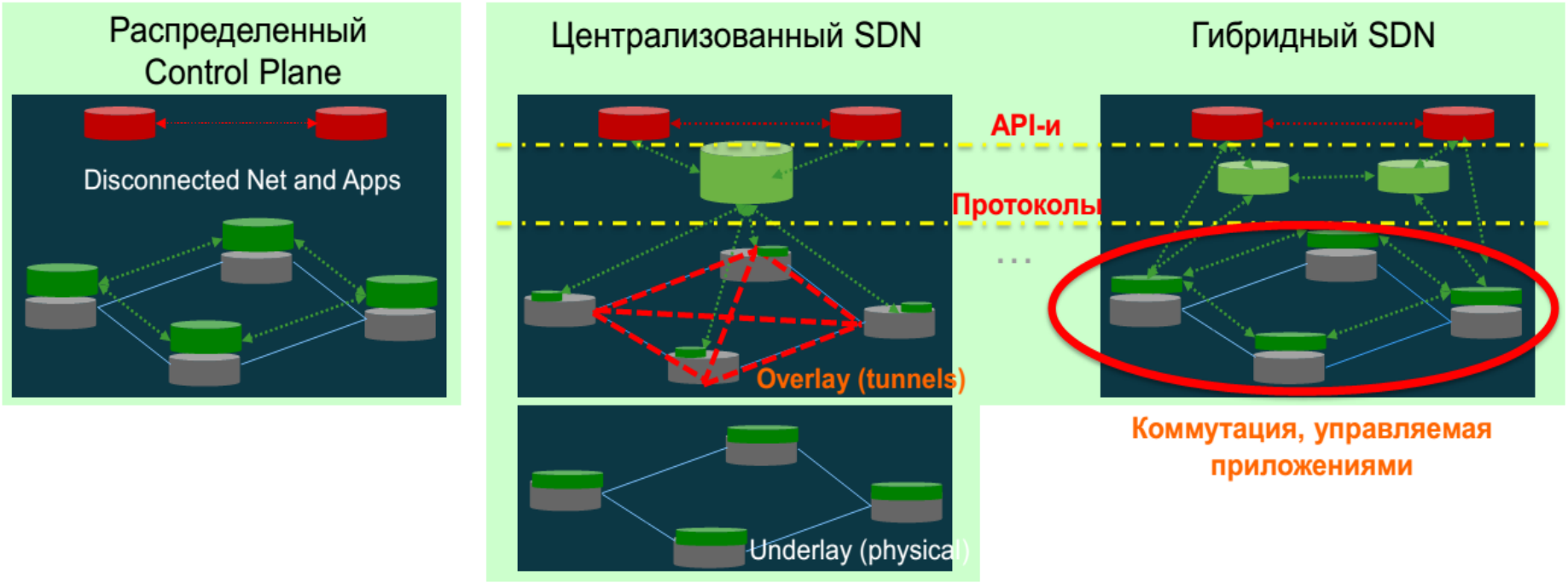
Академический подход



Развитие уровня управления

Традиционный Уровень Управления

Развитие Архитектуры Уровня Управления



Cisco Connect

Контроллеры SDN и Оркестраторы

- Контроллеры SDN, системы централизованного управления, системы оркестрации….
- Одно и тоже или взаимодополняющие компоненты?
- Давайте разберемся.

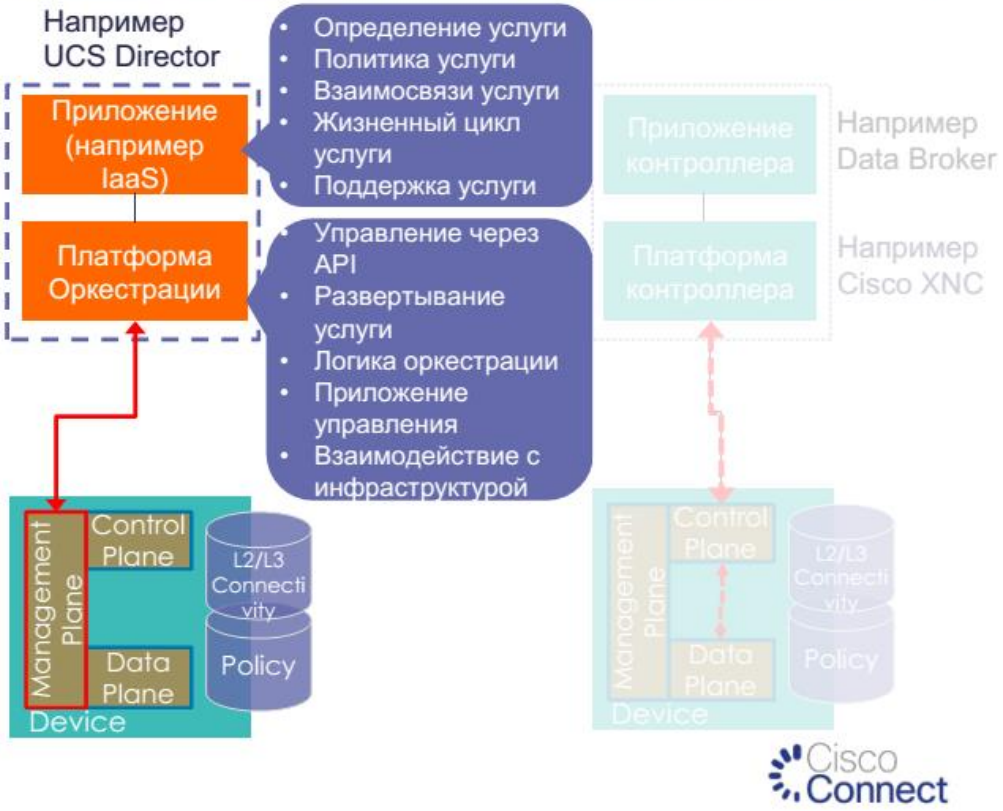
Различие между оркестраторами и контроллерами

Оркестраторы – платформы автоматизации, используются приложениями для конфигурирования инфраструктуры (через *Management Plane* опосредованно определяют/контролируют Control Plane и Data Plane), и выполняют конкретные **Сервисные Запросы**

- Например UCS Director

Контроллеры – платформы для приложений, которые программируют Control plane устройств или получают информацию о состоянии от Control Plane устройств

- Например XNC (платформа) – Data Broker (приложение)



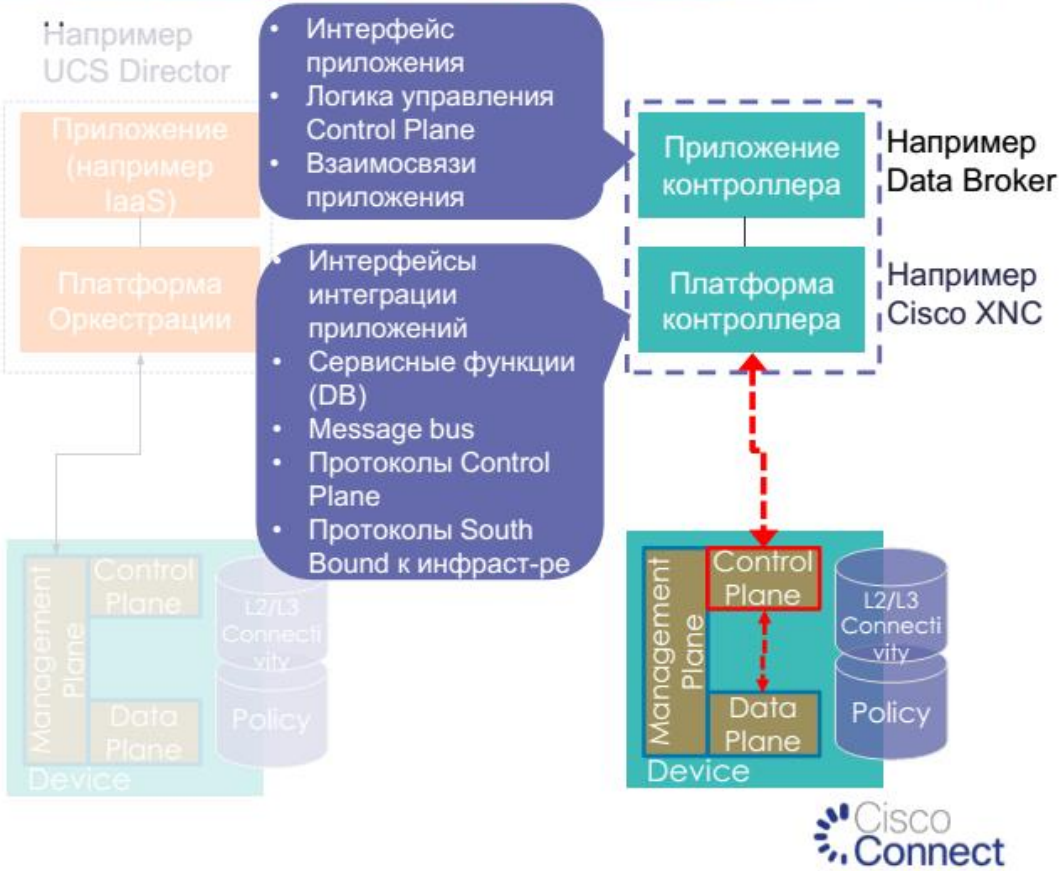
Различие между оркестраторами и контроллерами

Оркестраторы – платформы автоматизации, которые используются приложениями для конфигурирования инфраструктуры (через *Management Plane* опосредованно определяют/контролируют Control Plane и Data Plane), и выполняют конкретные **Сервисные Запросы**

- Например UCS Director

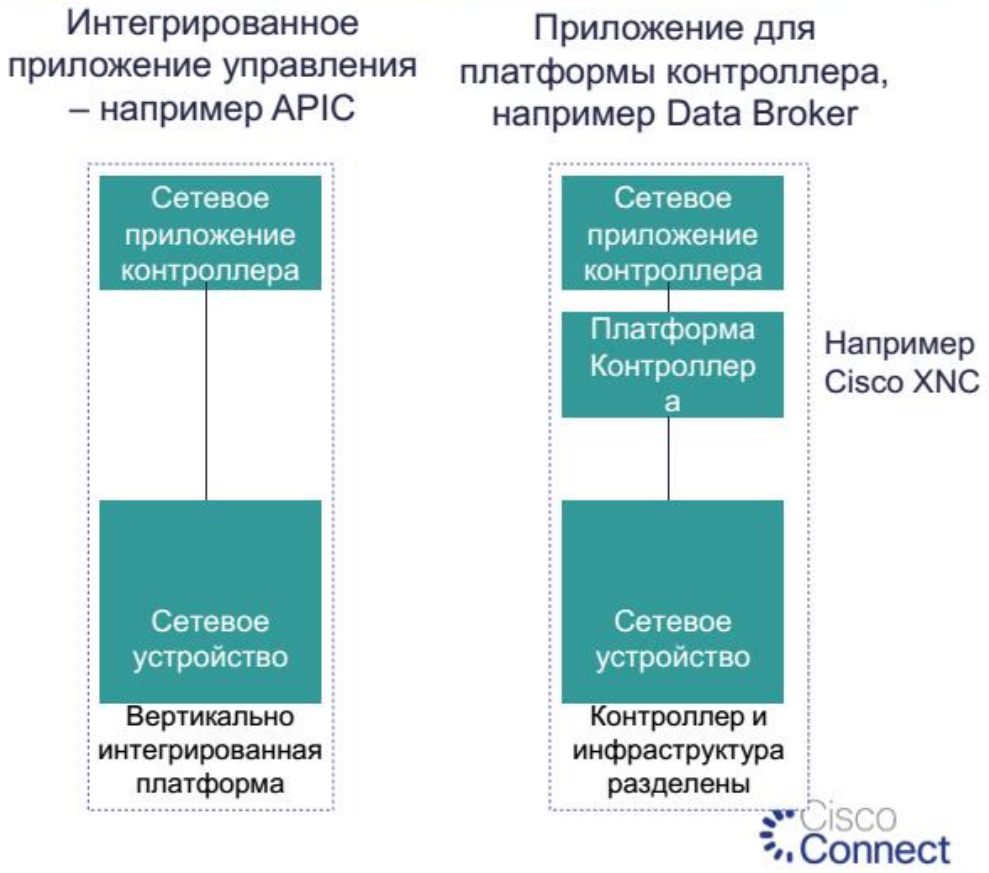
Контроллеры – платформы (для приложений), которые программируют Control plane устройств или получают информацию о состоянии от Control Plane устройств

- Например XNC (платформа) – Data Broker (приложение)



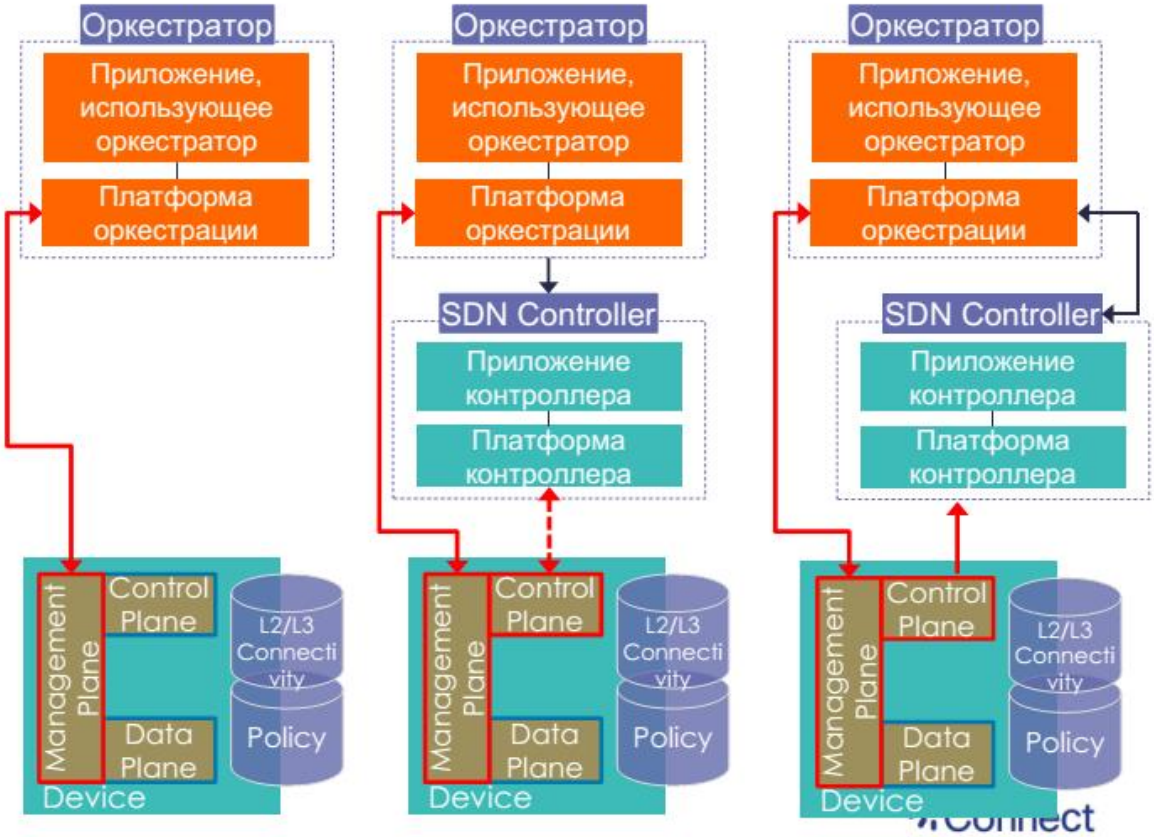
Различие между приложением и платформой контроллера

- Приложение может быть реализовано
  как интегрированное (например APIC)
  или
  на основе открытой платформы контроллера (например Data Broker).
- Множество сетевых приложений управления связано с необходимостью решать различные задачи управления сетевой инфраструктурой.
- Выбор платформы контроллера для приложения определяется возможностями платформы.

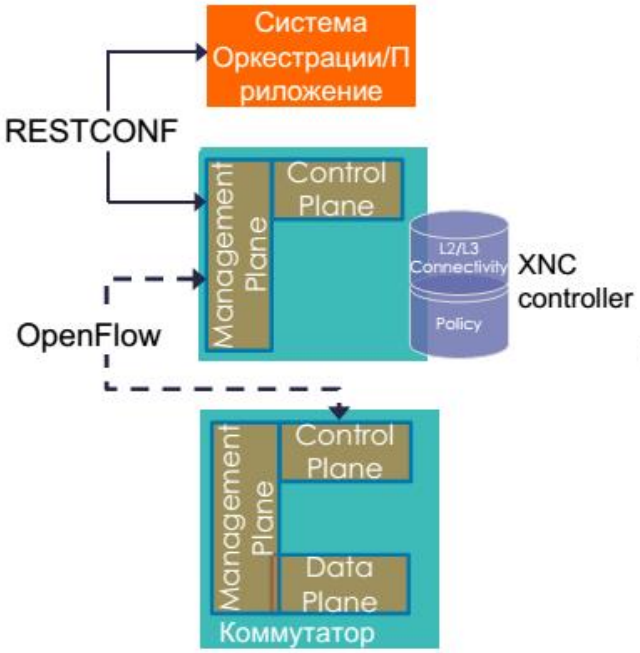


Способы взаимодействия контроллеров и оркестраторов

- Оркестраторы настраивают устройства через Management Plane:
  - CLI
  - RPC (Netconf)
  - Models (Netconf/YANG)
- Контроллеры программируют Control plane устройств:
  - Forwarding tables
  - Policy tables
  - Через протоколы SDN (OpenFlow)/протоколы Control Plane (I2RS, BGP-LS, PCEP)
- Оркестратор может опрашивать контроллеры в процессе принятия решения оркестрации

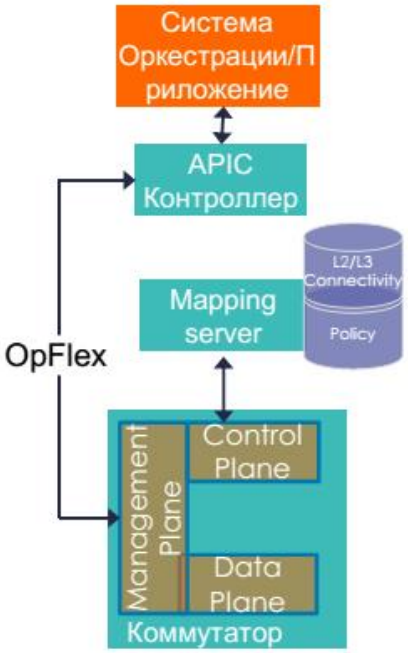


Модели взаимодействия контроллеров/ оркестраторов/ сети



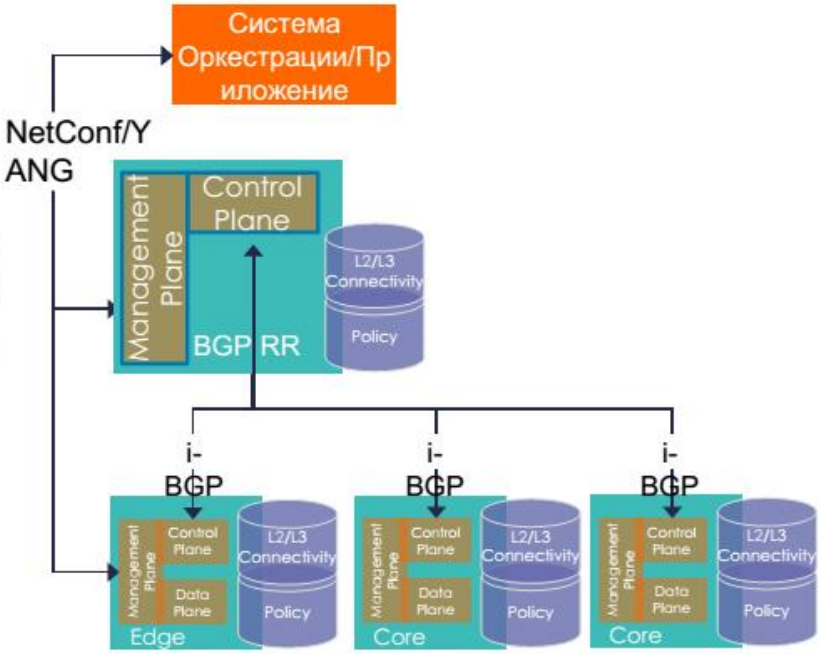
Push – например XNC/Data Broker

- Сетевое состояние проактивно спускается на сетевые устройства при создании новых правил в приложении



Pull – например ACI, LISP

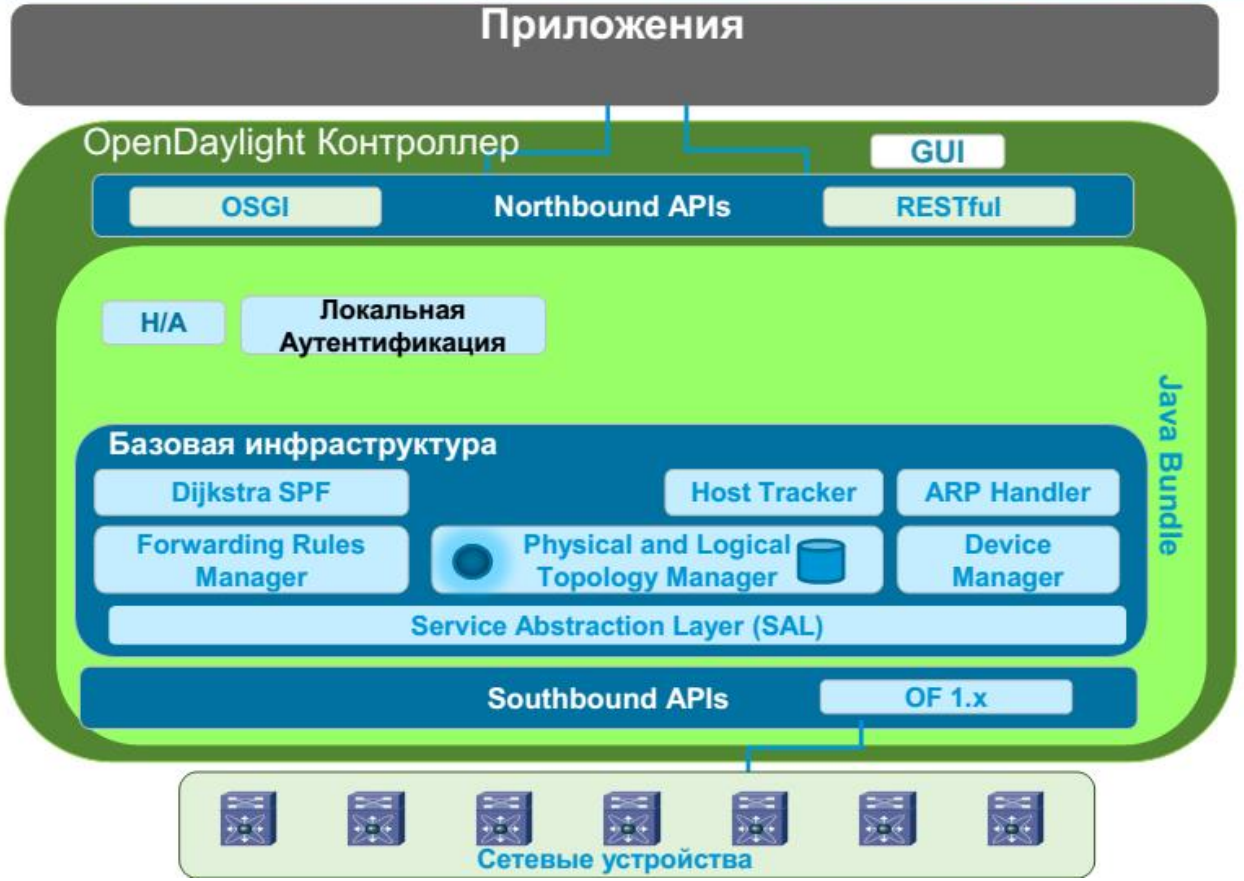
- Сетевое состояние передается на устройства доступа по запросу (реактивно)



Распределенная модель - например BGP, IGP

- Система оркестрации определяет конфигурацию устройств доступа, которые в свою очередь анонсируют доступность через протоколы маршрутизации

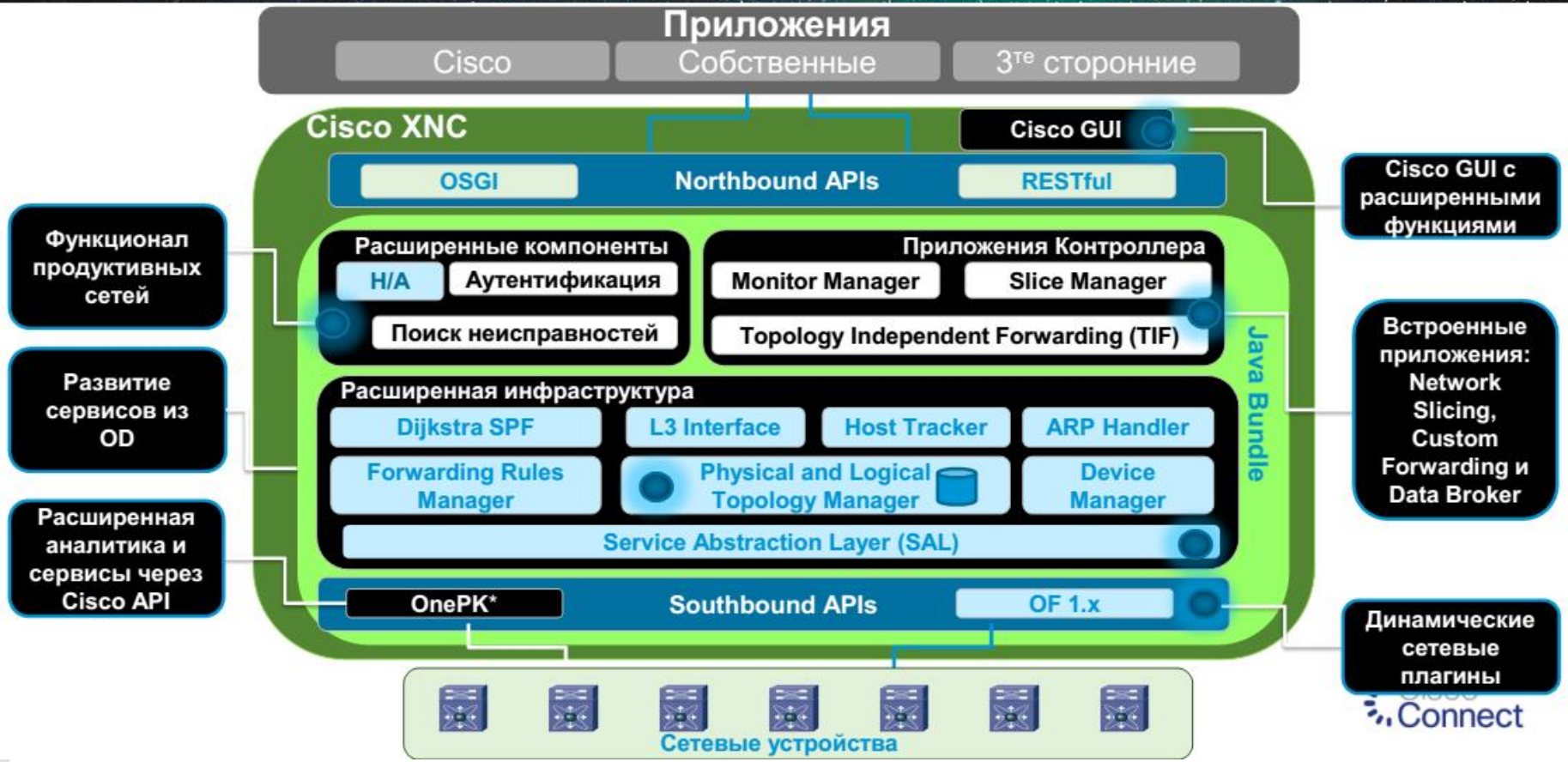
OpenDaylight Контроллер: базовые функции



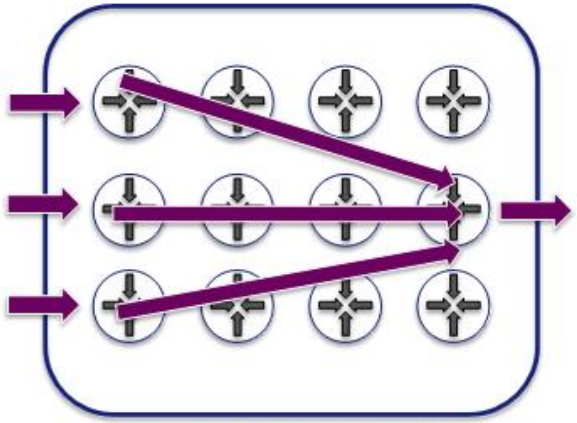
Cisco Connect

Cisco XNC Контроллер

Основан на OpenDaylight + расширенные функции и приложения

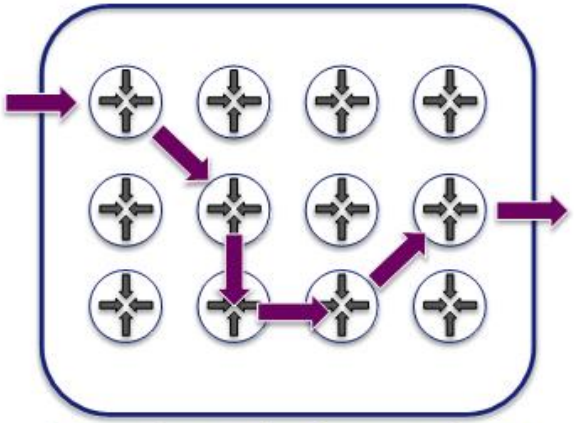


Openflow - Приложения с Контроллером Cisco XNC 1.5



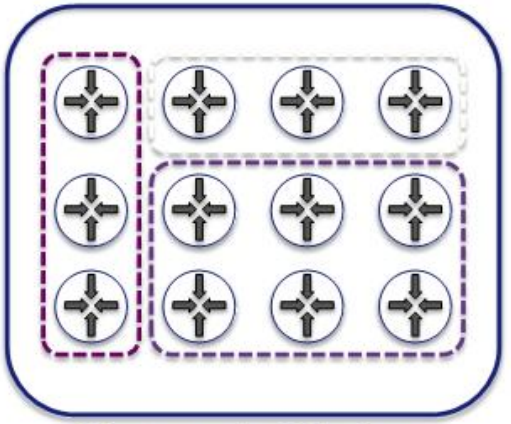
Nexus Data Broker
(Сеть Matrix)

Использование стандартных коммутаторов для передачи на основе политик зеркалированного трафика к Инструментам анализа



Topology Independent Forwarding
(Управление трафиком)

Статическое и динамическое создание правил для каждого потока на основе различных параметров



Network Slicing
(Сегментация сети)

Разделение сети с высокой степенью детализации политик

Cisco Connect

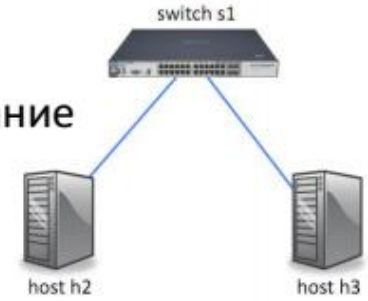
Санкт-Петербургский государственный университет
телекоммуникаций им. проф. М.А. Бонч-Бруевича

MiniNet система моделирования SDN

СПб ГУТ)))

Подходы к тестированию

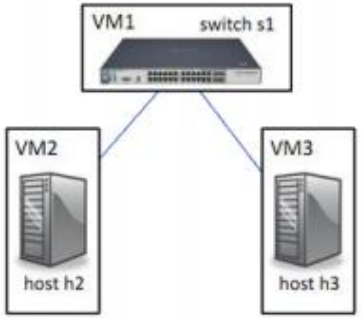
Физическое оборудование



ЗА:
- высокая степень доверия

ПРОТИВ:
- плохая масштабируемость

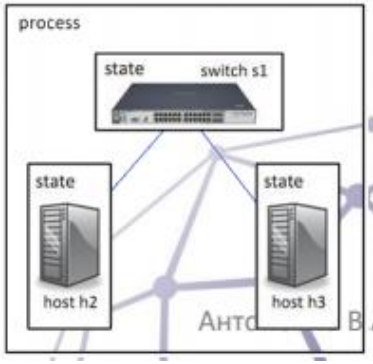
Эмуляция



ЗА:
- высокая степень доверия
- нет необходимости закупки оборудования

ПРОТИВ:
- высокие требования к ресурсам

Моделирование

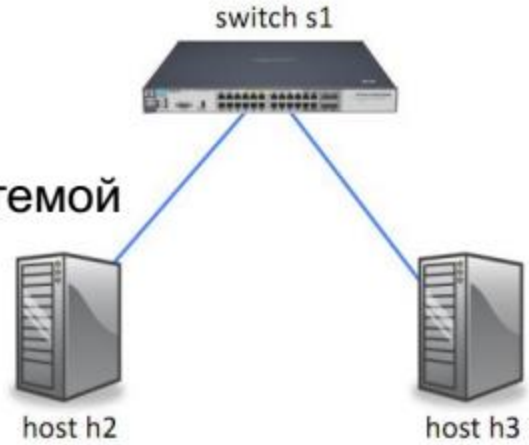


ЗА:
- пониженные требования к ресурсам

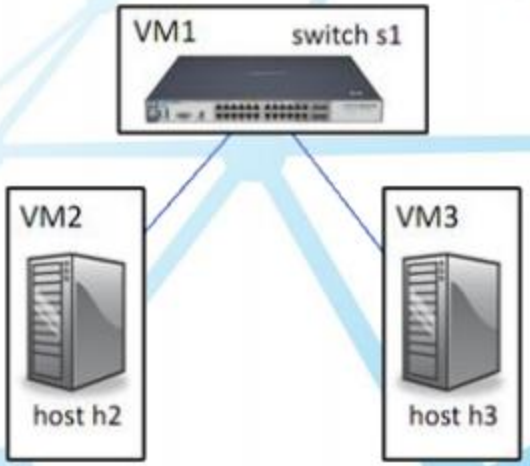
ПРОТИВ:
- необходимость доказывать корректность и адекватность модели

Mininet НЕ является чистой:

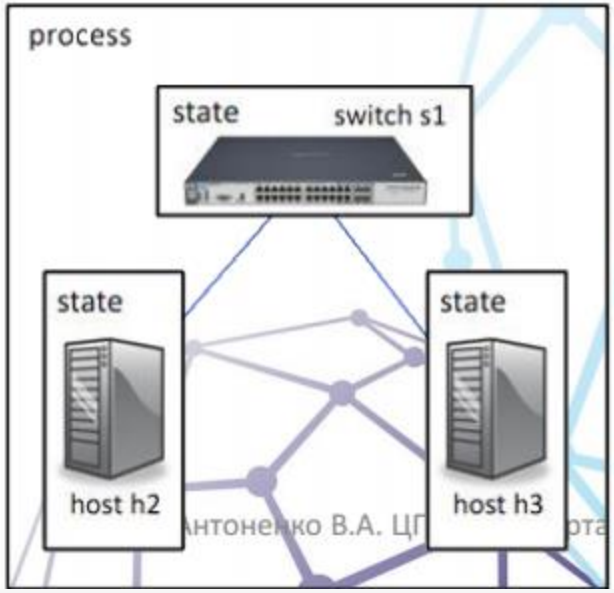
Реальной системой



Эмулятором

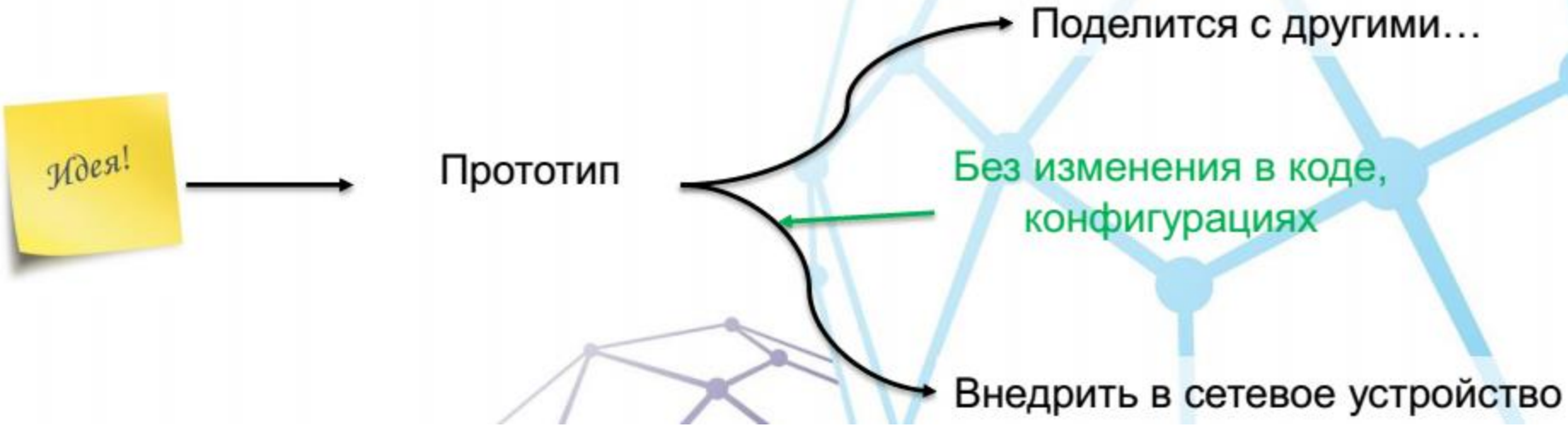


Средой моделирования



Цель MiniNet

- Строить/оценивать/демонстрировать реалистичные сегменты сети
- Воспроизводить ранее проведенные эксперименты



Возможности MiniNet

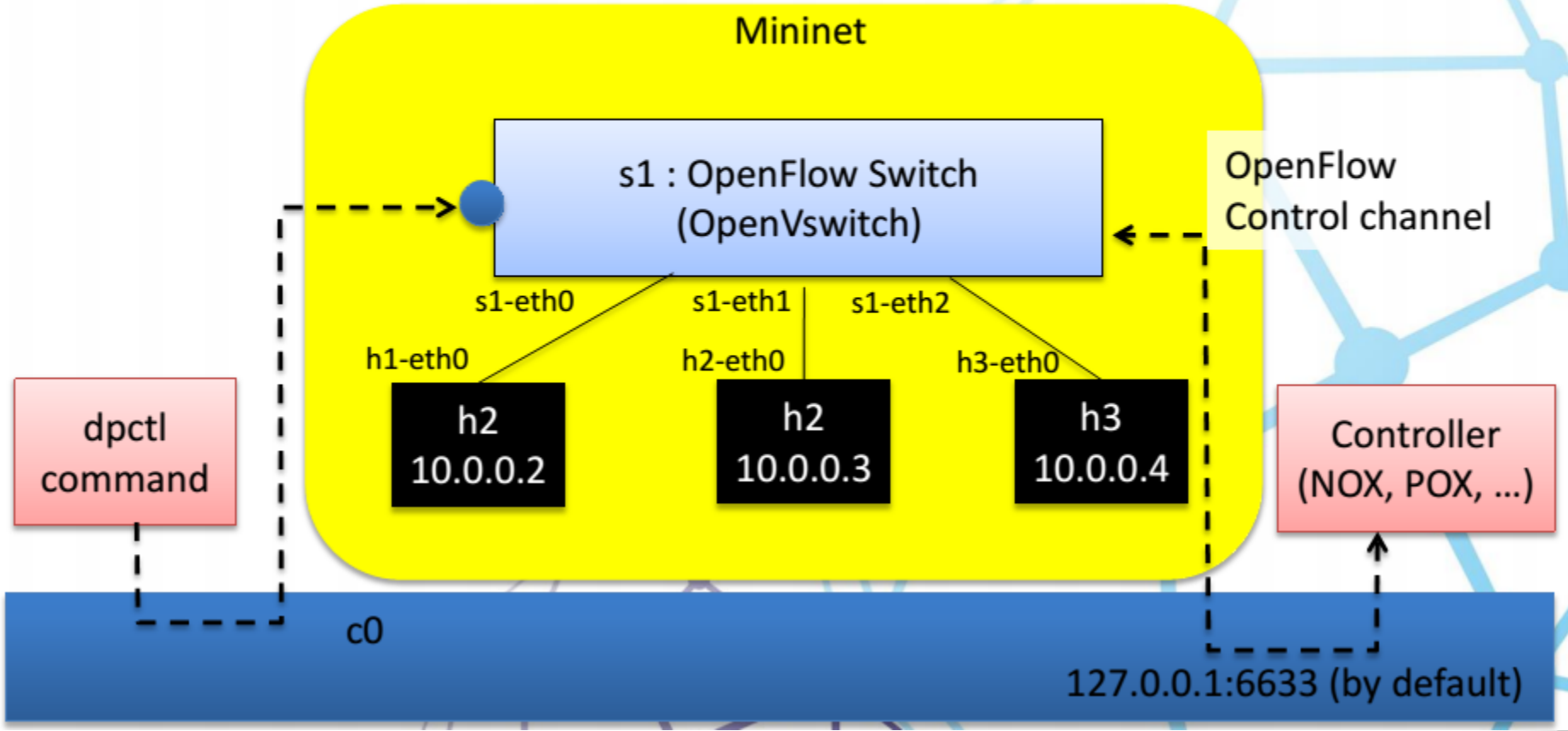
- Виртуальная сеть на локальном ПК
- Гибкое создание топологии сети
- Переносимость кода в реальные контроллеры

Особенности MiniNet

- подхода **легковесной виртуализации** в ОС Linux
- механизма виртуальных сетевых интерфейсов (**vEth pairs**)
- возможности отображать процессы ОС на пространство сетевых имен (**Network Namespaces**)

Пример

```
$ sudo mn --topo single,3 --mac --switch ovsk
--controller remote
```



MiniNet CLI

•Вывести список всех хостов, коммутаторов и контроллеров можно с помощью команды nodes

mininet> nodes
available nodes are:
h1 h2 c0 s1

•посмотреть топологию сети, а именно сопоставление портов коммутатора и хостов можно с помощью команды net:

mininet> net
c0
s1 lo: s1-eth1:h1-eth0 s1-eth2:h2-eth0
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2

MiniNet CLI

вывести конфигурацию сетевого интерфейса конкретного хоста можно с помощью классической команды ifconfig перед которой необходимо указать имя конкретного узла:

mininet> h1 ifconfig

h1-eth0 Link encap:Ethernet HWaddr 96:0d:f2:1a:e3:91
inet addr:10.0.0.1 Bcast:10.255.255.255 Mask:255.0.0.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:11 errors:0 dropped:0 overruns:0 frame:0
TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:846 (846.0 B) TX bytes:468 (468.0 B)

lo Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
UP LOOPBACK RUNNING MTU:16436 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

MiniNet CLI

Любой из портов коммутатора можно выключить и включить по желанию:

mininet> link s1 h1 down
mininet> link s1 h1 up

Посмотреть таблицу маршрутизации конкретного хост можно аналогично с использованием привычной команды route:

mininet> h1 route

Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
10.0.0.0 * 255.0.0.0 U 0 0 0 h1-eth0

MiniNet CLI

В принципе, на каждом из хостов, указывая предварительно его имя, можно выполнять большинство стандартных команд linux. Например посмотреть процессы любого из хостов или коммутаторов поможет все тот же ps:

mininet> s1 ps

PID TTY TIME CMD
1 ? 00:00:00 init
2 ? 00:00:00 kthreadd
3 ? 00:00:00 ksoftirqd/0

Кроме проверки доступности узлов с помощью ping можно еще протестировать пропускную способность между узлами с помощью старого доброго iperf:

mininet> iperf h1 h2
***** Iperf: testing TCP bandwidth between h1 and h2**
waiting for iperf to start up…* Results: ['1.35 Gbits/sec', '1.36 Gbits/sec']**

MiniNet CLI

Ну и в конце концов, можно просто получить терминал к любому из узлов:

mininet> xterm h1

Дополнительные сервисы
На каждом из виртуальных хостов, помимо стандартных процессов есть возможность запускать сторонние сервисы. Например это может быть простой веб-сервер на Python:

mininet> h1 python -m SimpleHTTPServer 80 *&*

При желании, можно ограничить пропускную способность каналов до произвольных значений. А в добавок к этому есть возможность указать задержки в канале(latency).

$ sudo mn --link tc,bw=10,delay=10ms

Команда выше, разворачивает сеть по умолчанию, с пропускной способностью между узлами ограниченной в 10 Mbit/s и минимальными задержками в 10 ms.
Теперь пинг между узлами будет идти с задержкой в 10 ms.

Варианты использования

- **Моделирование процесса заполнения таблицы потоков** на OpenFlow-коммутаторах
- **Моделирование работы** созданного приложения **на различных топологиях**
- **Моделирование работы** созданного приложения **в нештатных ситуациях**
- **Моделирование работы** созданного приложения при **нарушении информационной безопасности** сети

Спасибо за внимание!

СПб ГУТ)))