



Linux в качестве ОС для коммутаторов ЦОД

Александр Петровский, Mellanox System Engineer

Конференция OS DAY, Май 2017



Требования



- Масштабируемость
- Гибкость и скорость инноваций
- Унифицированное управление и автоматизация
- Эффективность и сокращение ТСО

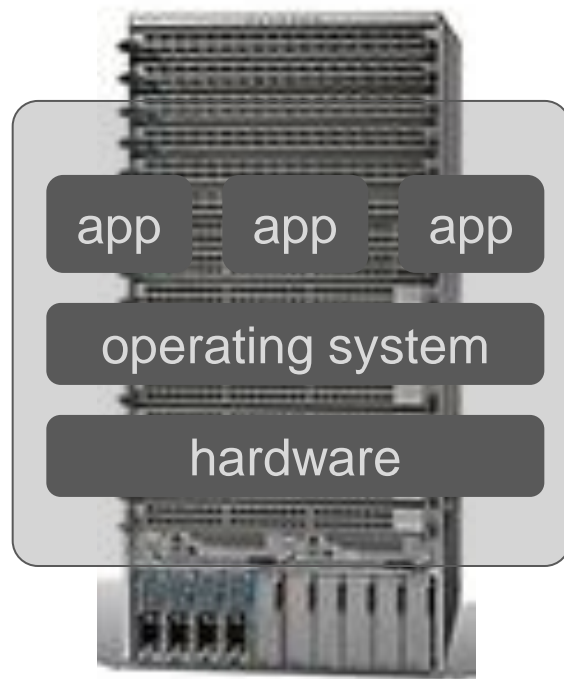
Решения



- Открытая архитектура
- Гибкая конфигурация и кастомизация
- Стандартные инструменты и API для всех элементов инфраструктуры
- Дезагрегация «железа» и ПО

Linux в качестве основной ОС лучше всего подходит под современные требования к ИТ-инфраструктуре ЦОД

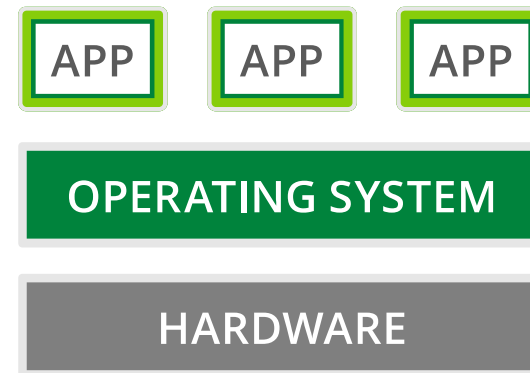
Проприетарные коммутаторы:



Закрытые платформы:

- Привязка к одному вендору
- Дорого!
- Медленный цикл разработки

Дезагрегация и Open Ethernet:

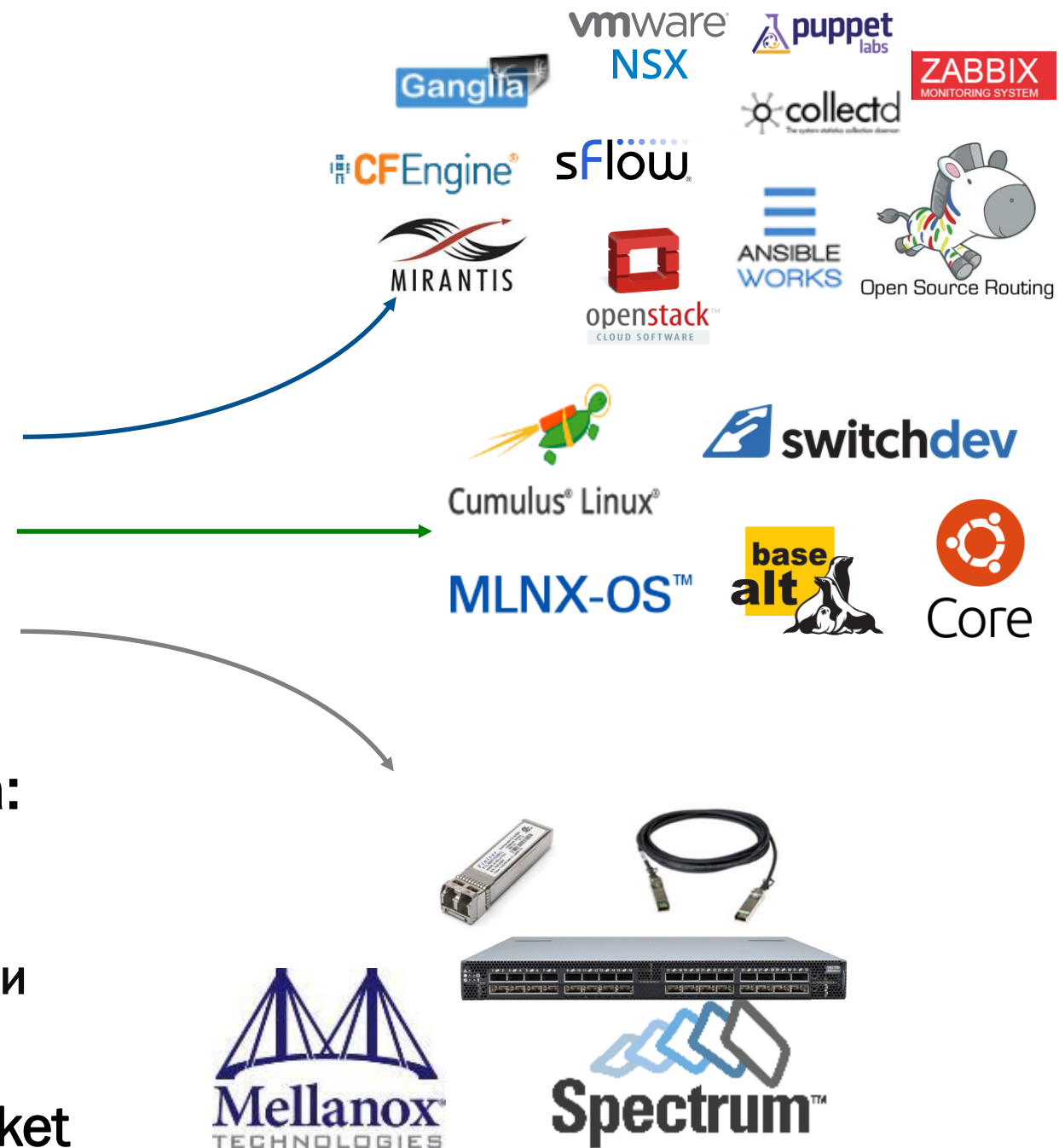


Возможность выбора:

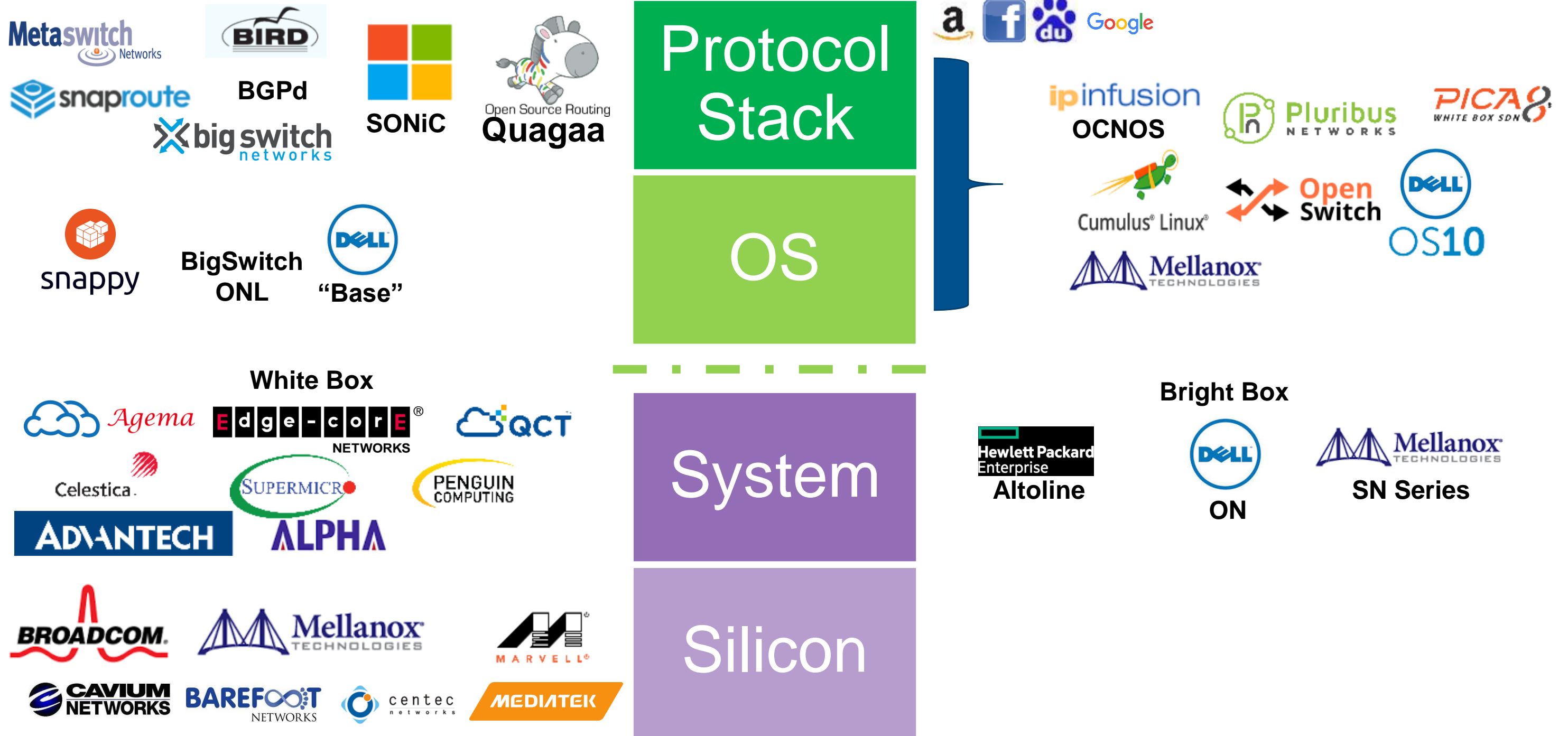
- Лучшего железа
- Подходящей ОС
- Лучшего ПО под задачи

Снижение CAPEX

Быстрый Time-to-market



Экосистема Open Ethernet - большая картинка



■ Традиционные закрытые NOS

- NOS неотделима от железа
- У каждого производителя HW своя NOS
- Полностью закрытая платформа
- Закрытое проприетарное окружение
- Часто основывается на Linux
- Примеры: Cisco IOS-*/NXOS, JunOS, Arista EOS, HPE Comware и др.

Closed Apps

Closed OS

Closed Hardware

■ Коммерческие NOS

- Для bare-metal/white-box железа
- Возможность выбора HW
- Возможность выбора приложений
- Стандартный Linux в основе NOS
- Открытое Linux окружение (Cumulus) или оболочка CLI
- Data Plane через закрытый SDK
- Примеры: Cumulus, Pica8, Metaswitch и др.

Open Apps

Closed OS

Open Hardware

Device manufacturers



■ Открытые NOS

- Развивается сообществом/крупными web-компаниями
- Возможность выбора HW, ОС, сетевого стека и приложений
- Стандартный Linux + открытое Linux окружение
- Data Plane через открытые или «открытые» абстракции API
- Примеры: ONL, SONiC, FBOSS, OpenSwitch

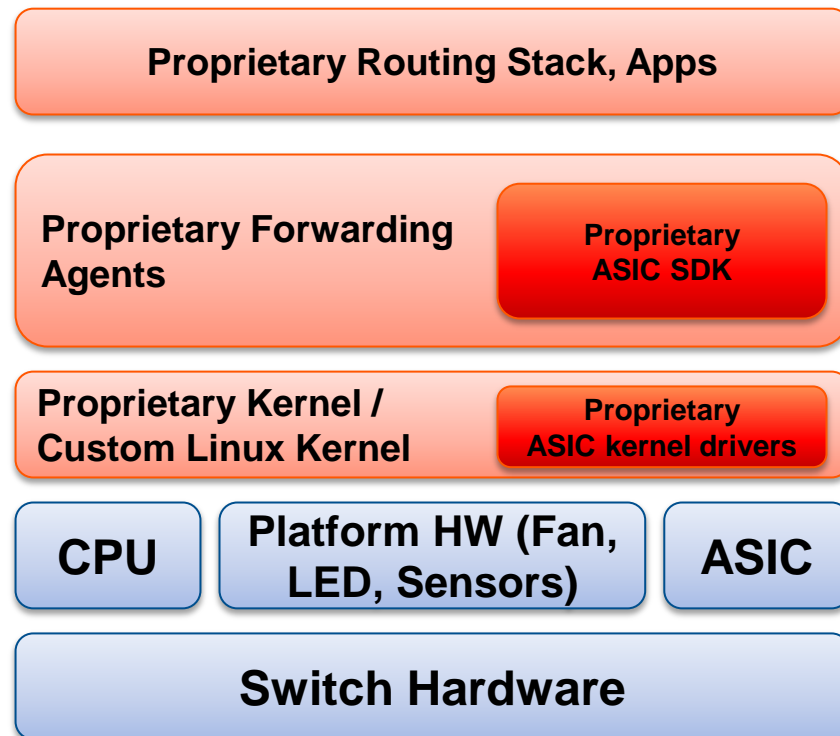
Open Apps

Open OS

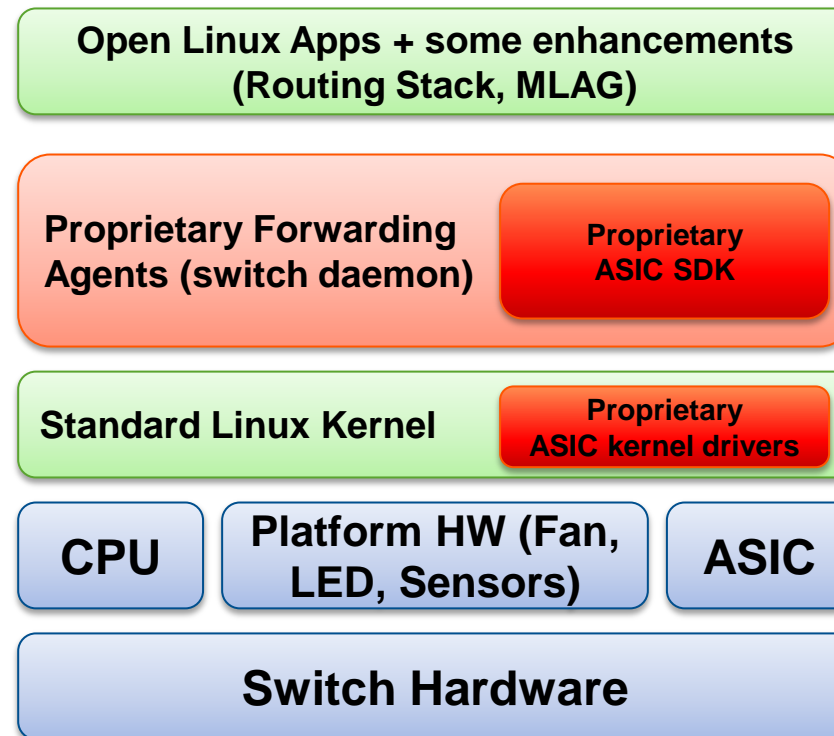
Open Hardware

Chip manufacturers

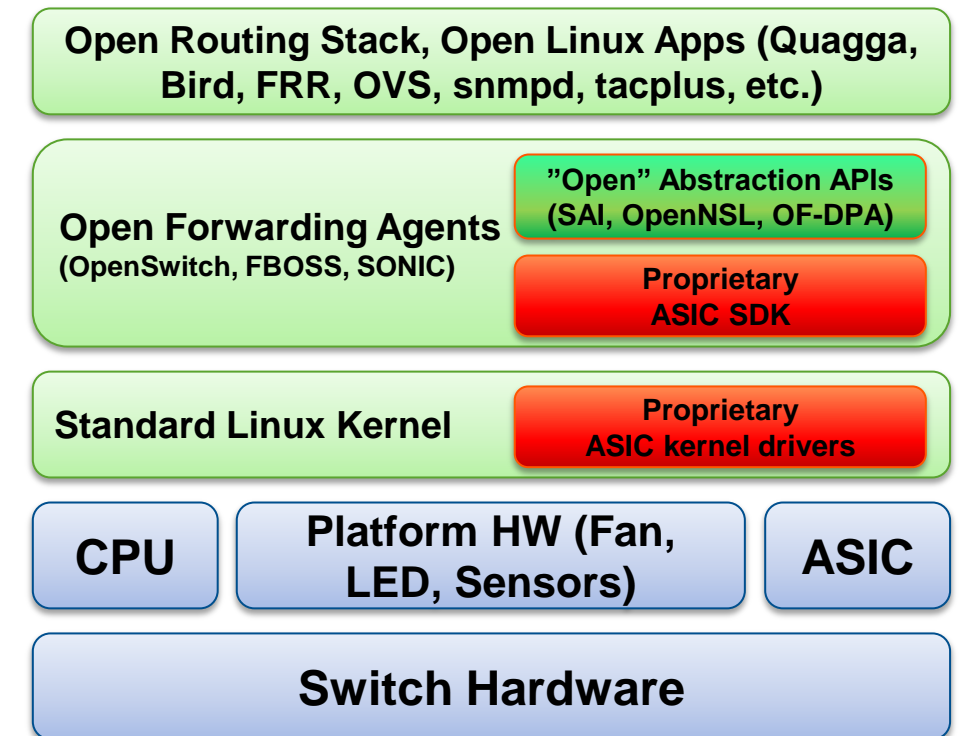
Закрытые Legacy NOS



Коммерческие NOS на базе Linux



Открытые NOS на базе Linux



Open/Free SW

Proprietary SW

Binary blob

Что такое Forwarding Agent и Routing Stack в открытых NOS?

■ Forwarding Agent (FA)

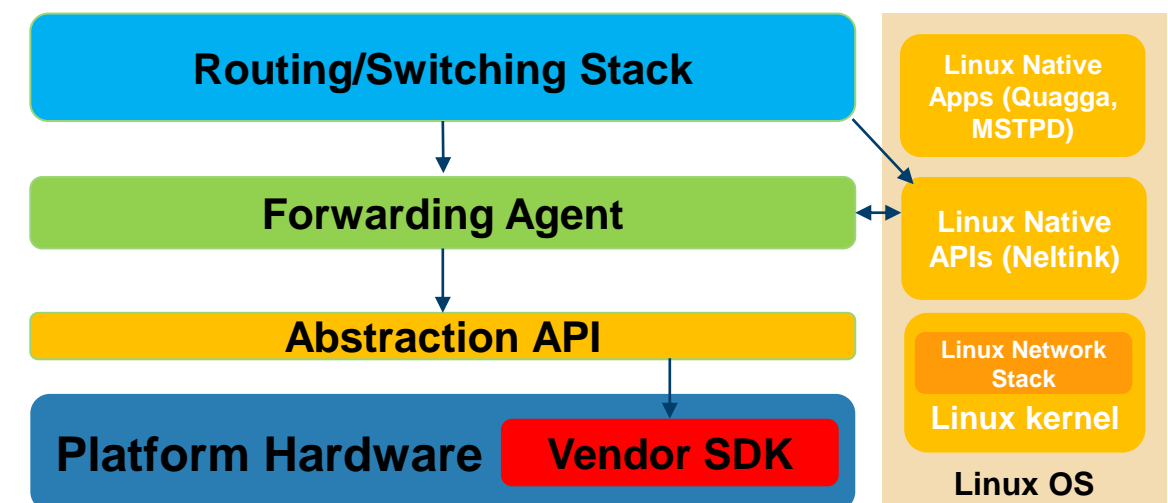
- Реализует Data Plane функции коммутатора через API
 - управление портами, VLAN, bridging, routing
- Формирует промежуточные структуры данных для выгрузки в ASIC (состояние портов, L2/L3 кэш, ACL таблицы и т.п.)
- Получает и обрабатывает статистику от ASIC
- Использует API для доступа к функциям ASIC
- Интегрируется с Routing Stack и/или с сетевым стэком Linux (Netlink Listener)
- FlexSwitch, SONiC, OpenSwitch, FBOSS

■ Виды API для связи с ASIC

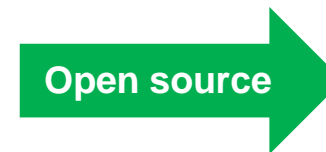
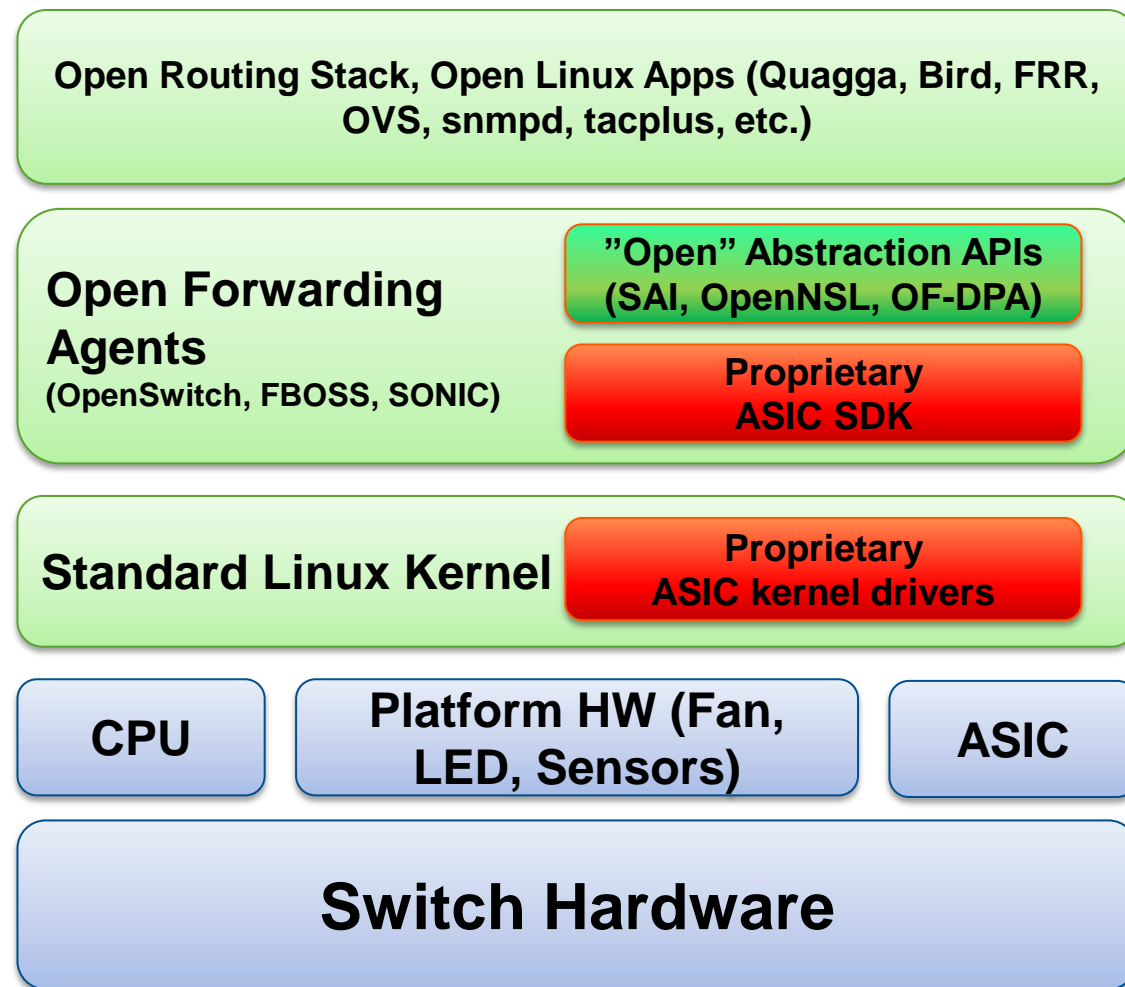
- SDK вендора (Mellanox, BRCM, Marvell и т.д.)
- Switch Abstraction Interface (SAI)
- «Открытые» прослойки над SDK (OpenNSL, OF-DPA)

■ Routing/Switching Stack

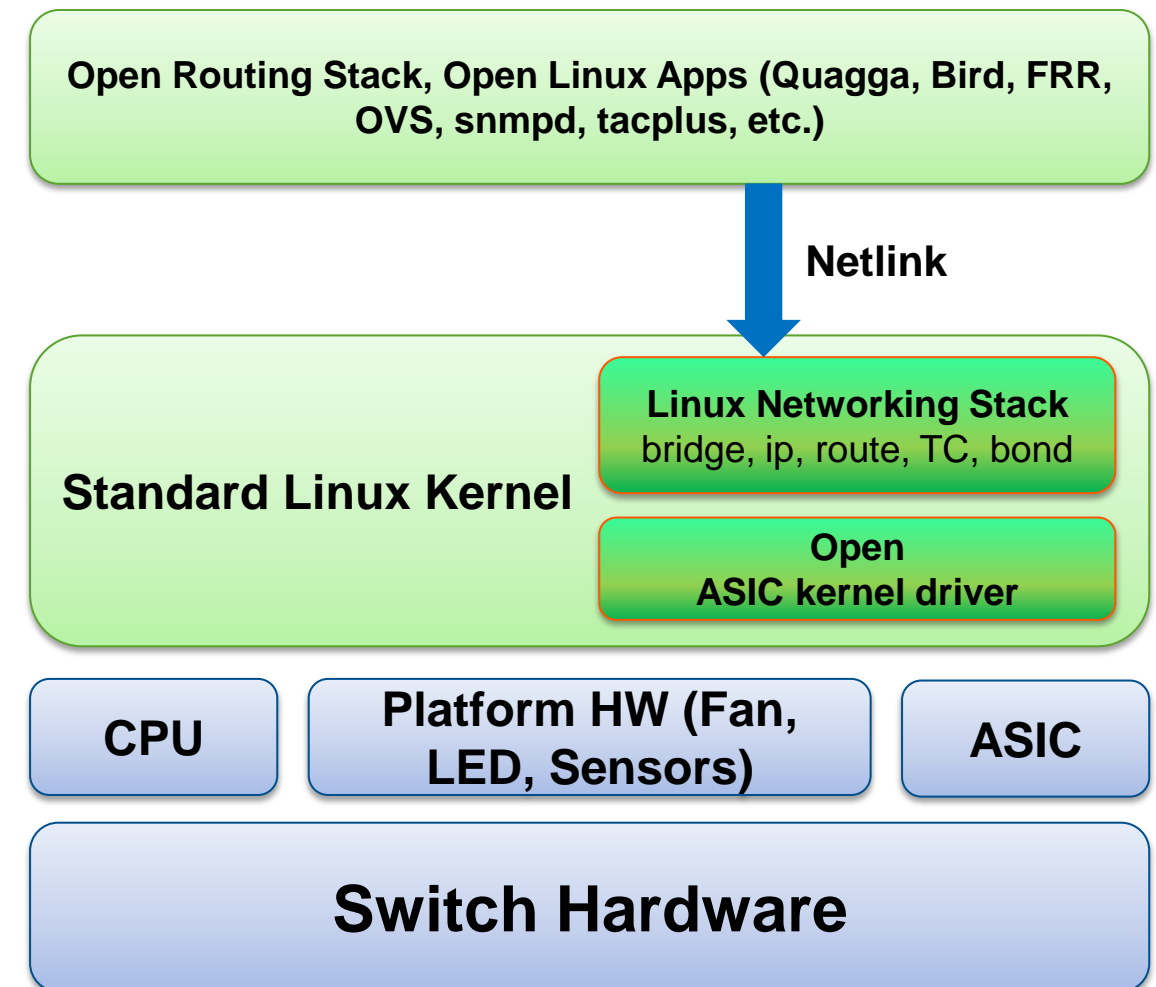
- Открытые реализации протоколов коммутации и маршрутизации
- Бывают как отдельные проекты по каждой функции, так и интегрированные стеки
 - Quagga, Bird, FRR, XORP, GoBGP (routing)
 - Linux bridge, mstpd (switching)
 - Open vSwitch (switching + OpenFlow)
 - FlexSwitch (SnapRoute), SONiC (MS Azure), FBOSS (Facebook)
 - Коммерческие: Metaswitch, IPinFusion, Open EOS



Открытые NOS на базе Linux



Открытый Linux в качестве NOS



**Нет закрытых/проприетарных компонентов
Драйвер ASIC в upstream ядра Linux**

■ OpenWRT swconfig

- Драйвер ядра + инструменты конфигурации в User-space
- Поддержка различных SOHO-чипов (Atheros, Broadcom, Realtek) для домашних роутеров
- Нестандартный способ настройки интерфейсов
 - Интерфейсы в системе не создаются
 - Для настройки используются User-space утилиты
- Не принят в upstream, существует только в OpenWRT

■ DSA

- Стандартная замена swconfig в upstream
- Изначально для SOHO чипов Marvell
 - Сейчас поддерживает Marvell, Broadcom, Qualcomm/Atheros
- Создает netdev представления для портов коммутатора
- Поддерживает offload состояния портов, bridge offload, MAC learning/ageing и т.п.
- Только MDIO, нет host интерфейса

■ Switchdev

- Реализует **абстрактную** модель драйверов для Switch ASIC
 - DSA - один из драйверов switchdev
- netdev представления для портов коммутатора
- Стандартные операции FDB, MDB, VLAN, FIB, TC
- Поддержка Host интерфейса
- Поддерживает любой тип шины подключения к ASIC: PCI/PCIe, MDIO, SPI
- Драйверы High-end чипов коммутатора: **Mellanox SwitchX-2/SwitchIB/Spectrum** (до 6.4 Тбит/с)

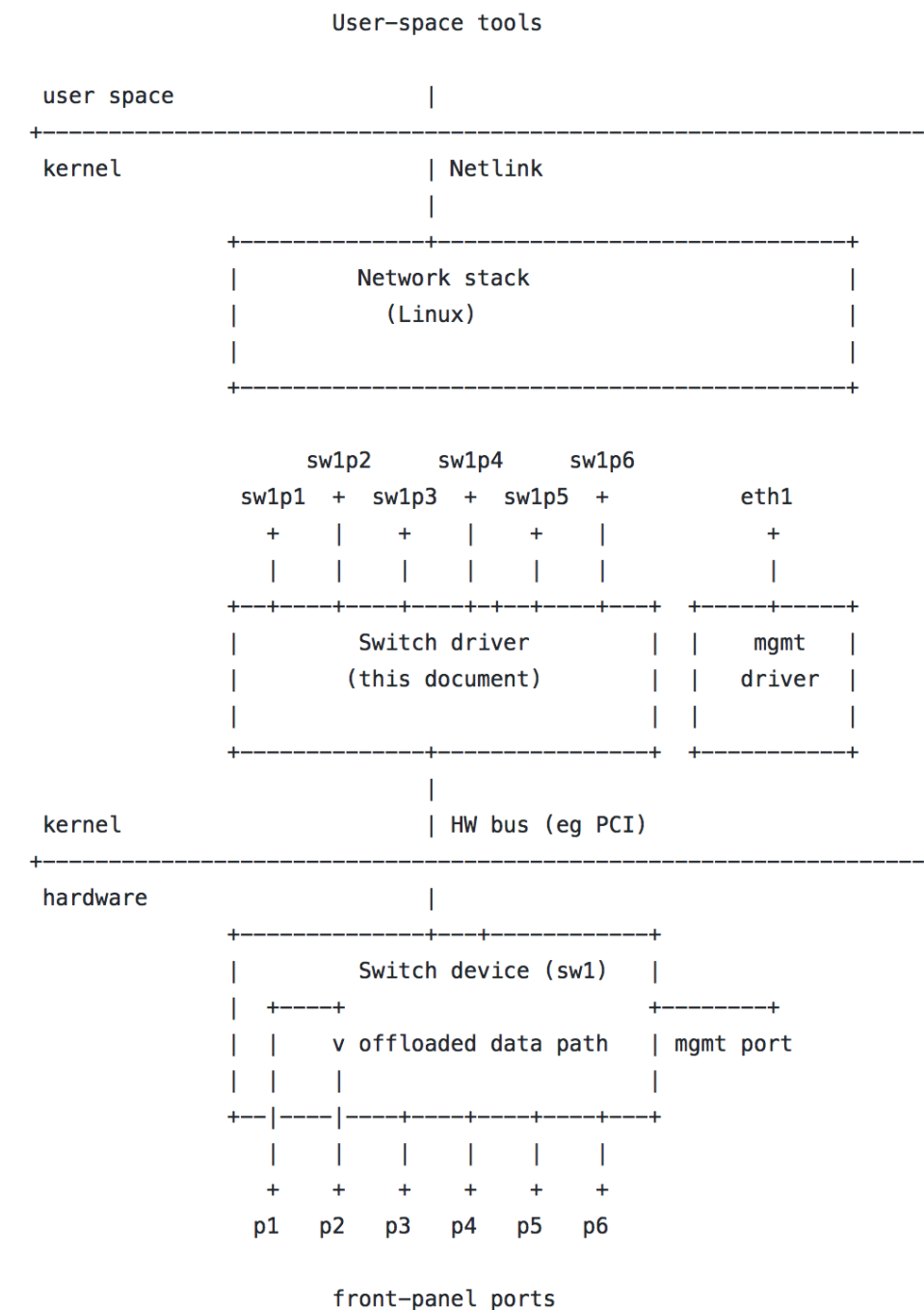
Основной фокус разработки



Switchdev - подсистема ядра Linux для управления Switch ASIC

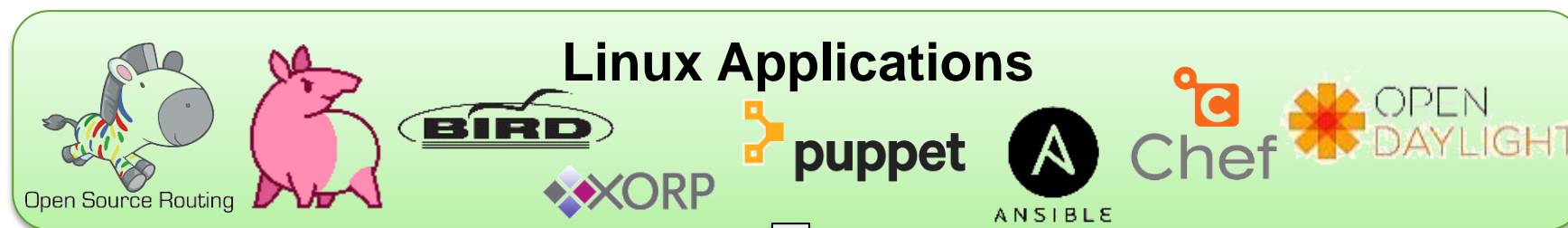


- Реализует абстрактную модель драйверов для аппаратных чипов коммутатора в ядре Linux
 - Реализует инфраструктуру для Data Plane offload из ядра Linux в ASIC
 - Описывает базовые примитивы и механизмы offload:
 - Netdev представление физических портов ASIC
 - Port state management, Port topology
 - L2 forwarding offload (FDB, MDB, STP)
 - L3 routing offload (FIB, ARP, ND)
 - TC offload
- Драйверы switchdev
 - mlxsw - драйвер Mellanox Spectrum ASIC
 - DSA драйвер (BRCM, Marvell, etc.) - для low-end ASIC с поддержкой DSA (домашние роутеры)
 - SR-IOV драйвер - для коммутации SR-IOV VF в NIC
 - Rocker драйвер - программный эмулятор ASIC
- Материалы
 - <https://github.com/torvalds/linux/blob/master/Documentation/networking/switchdev.txt>
 - <https://lwn.net/Articles/675826/>



Архитектура Linux-коммутатора на базе Switchdev

User Space



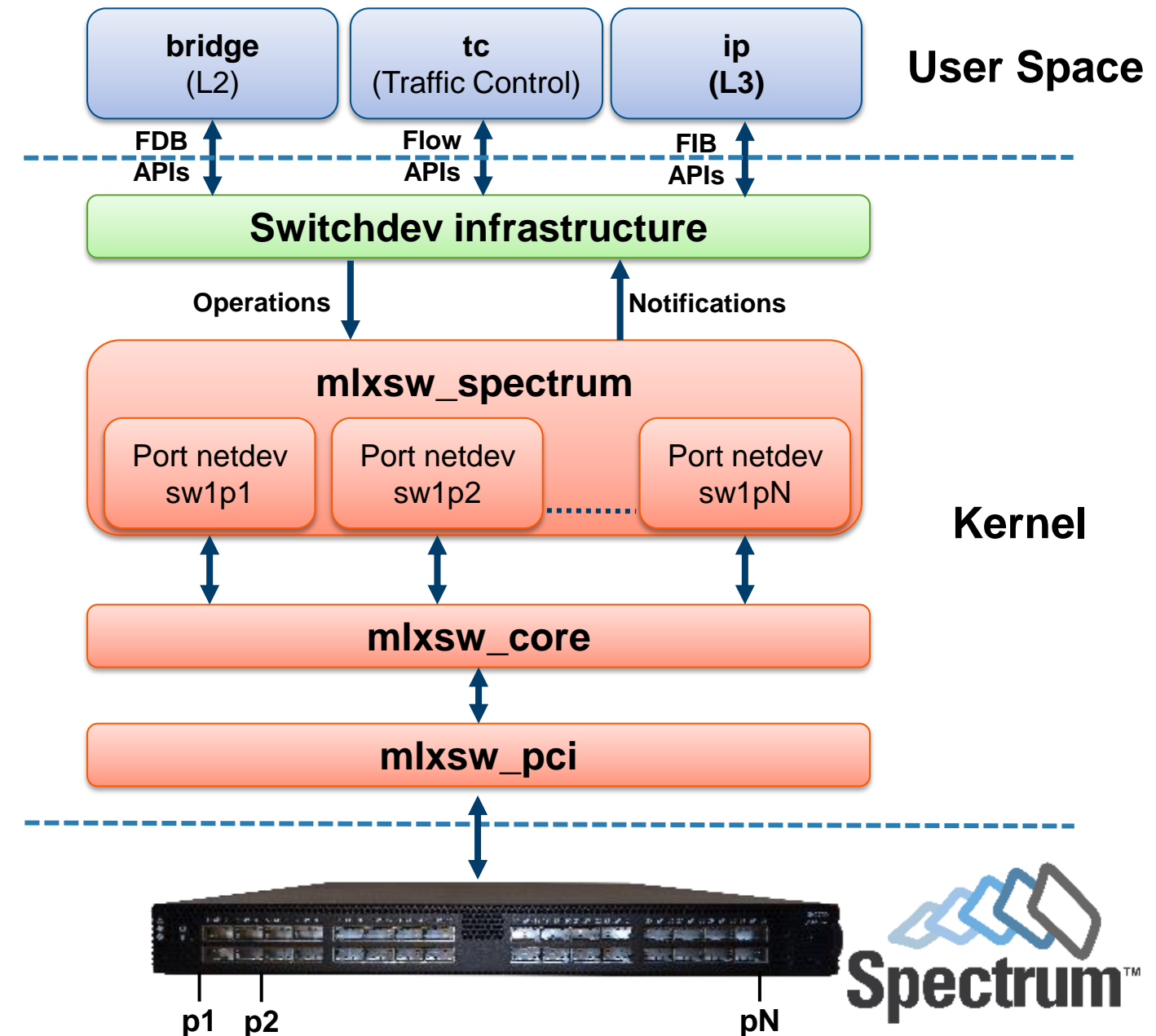
Kernel



Hardware



- **Открытый** драйвер для Mellanox Spectrum ASIC в ядре Linux
- Каждый порт коммутатора представлен как сетевой адаптер в ядре Linux
 - Доступ к статусу линии и статистика I/O
- Используя стандартные утилиты Linux, порты могут:
 - Коммутировать (L2)
 - Объединяться в Bond (LACP)
 - Разделены на под-сети (VLANs)
 - Маршрутизировать (L3 routing)
 - Фильтровать (ACL)
 - Приоритизировать (QoS)
 - Запаковать трафик в туннель
- Драйвер отражает сетевую конфигурацию Linux в железо для полноценной L2/L3 коммутации в матрице (ASIC)
- Доступен в upstream начиная с ядра 4.4

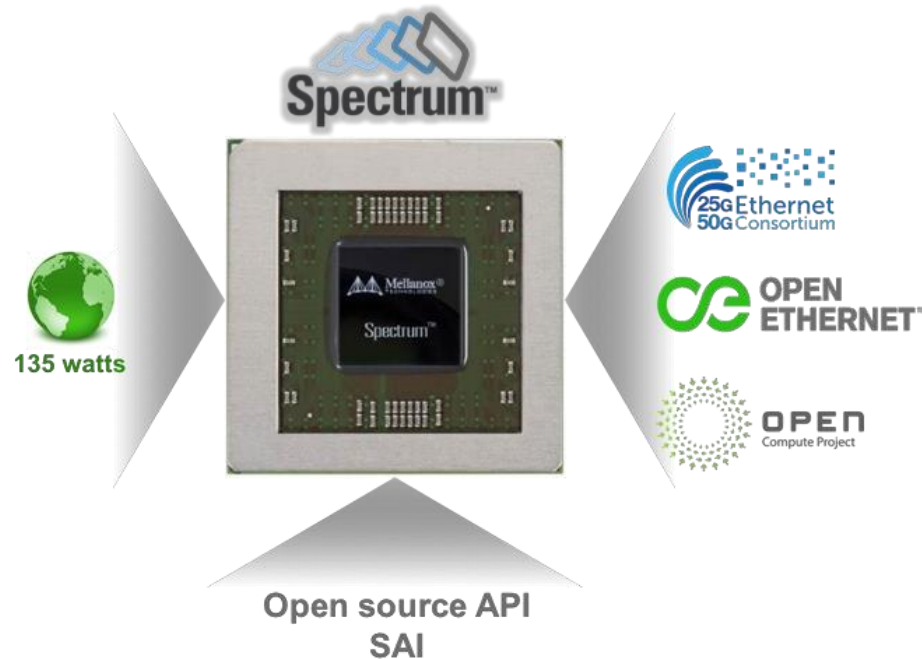


L2 Switching	<ul style="list-style-type: none"> • Ports Bridging/Bonding, VLANs, STP, LLDP, FDB learning/aging, IGMP, SPAN
L3 Routing	<ul style="list-style-type: none"> • Unicast IPv4 router, ECMP, VRFs • Integrated with any L3 protocol stack (Quagga, BIRD, XORP, GoBGP)
Policy Based Forwarding	<ul style="list-style-type: none"> • ACLs (based on TC Flower) • OpenFlow (with OVS and TC)
Chassis Management	<ul style="list-style-type: none"> • LEDs, fans, power supplies and temperature gauges access
Advanced Configuration	<ul style="list-style-type: none"> • Port splitter • Packet buffer configuration, QoS, DCB • Port/traffic mirroring (TC-based, sFlow)
Roadmap	<ul style="list-style-type: none"> • Unicast IPv6 • Tunneling (GRE/VxLAN) • Multicast Router • MLAG • Buffer Monitoring

Scale	
Item	Max/Current
Max # MAC addresses	256k/224k
Max # IPv4 routes / LPMs	256k/118k
Max # VLANs	4k/4k
Max # IGMP groups	8k/8k
Max # ECMP routes	256k/1500
Max ECMP group size	4k/4k
Max # ports in LAG	64/64



100GbE коммутаторы Mellanox



Mellanox Spectrum ASIC - высокопроизводительный data plane
OS BaseALT - гибкий и удобный control plane коммутатора

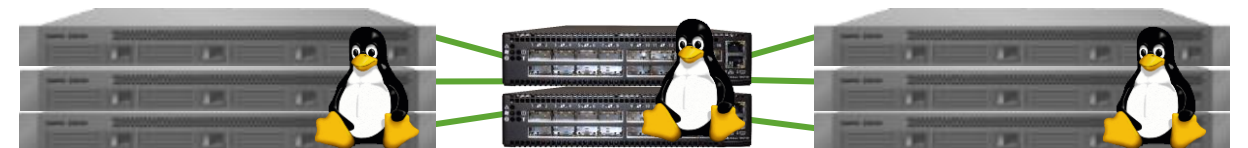
Отечественная ОС BaseALT

- **Отечественный дистрибутив Linux**
- Сизиф - Один из пяти крупнейших в мире репозиториях программ
- Содержит около 18000 исходных пакетов (март 2017).
- Поддержка аппаратных платформ: x86, x86_64, ARMv7, AArch64; отечественные "Эльбрус", "Байкал-М".
- Полный открытый технологический цикл подготовки, выпуска и поддержки дистрибутивов Альт.
- Воспроизводимость сборки.
- Быстрая реакция на появляющиеся угрозы.

Open Source приложения и утилиты

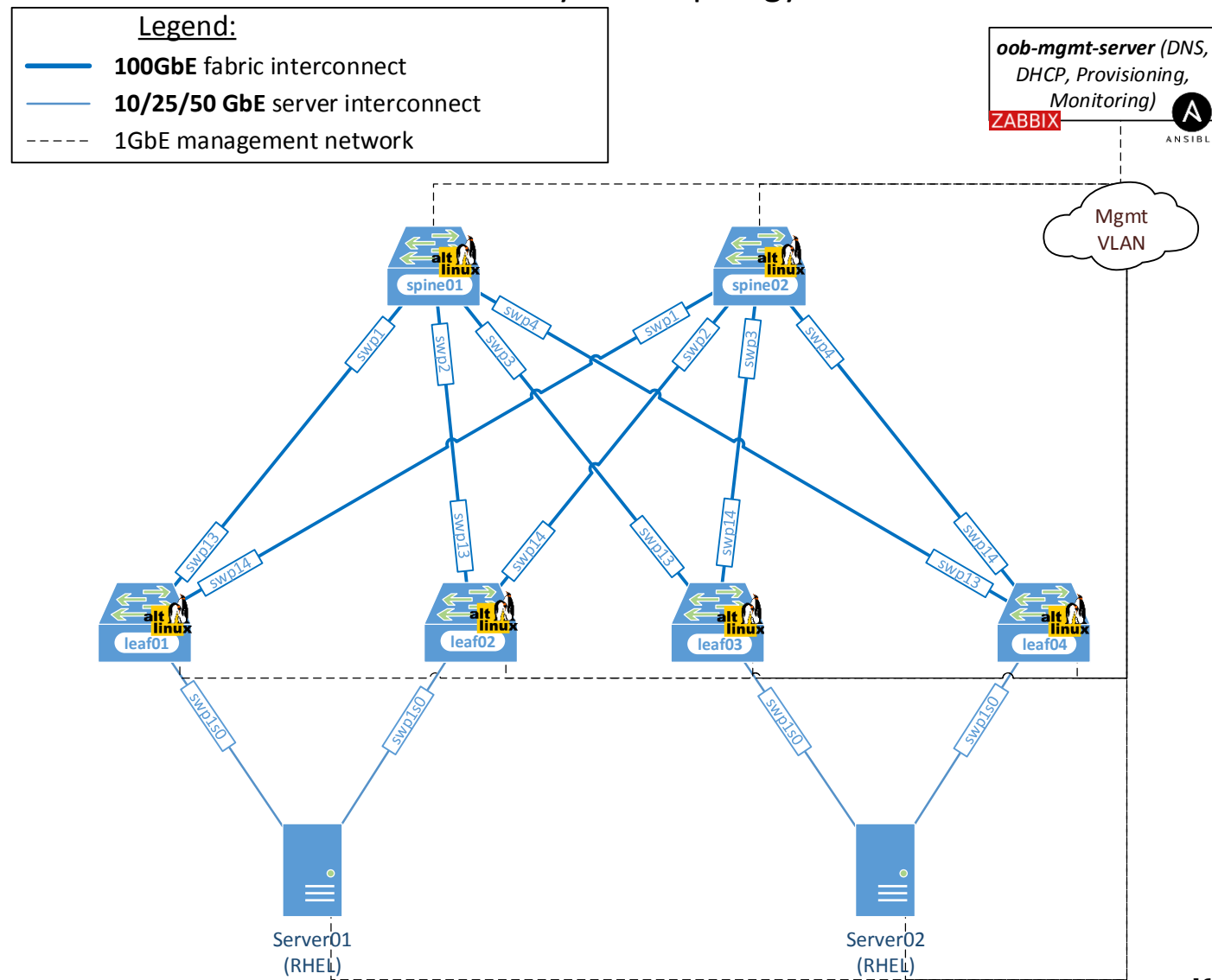
iproute2 **Bird** Ildpd Ansible
Quagga **GoBGP** teamd hsFlow
Zabbix VRRPd

Linux end-to-end

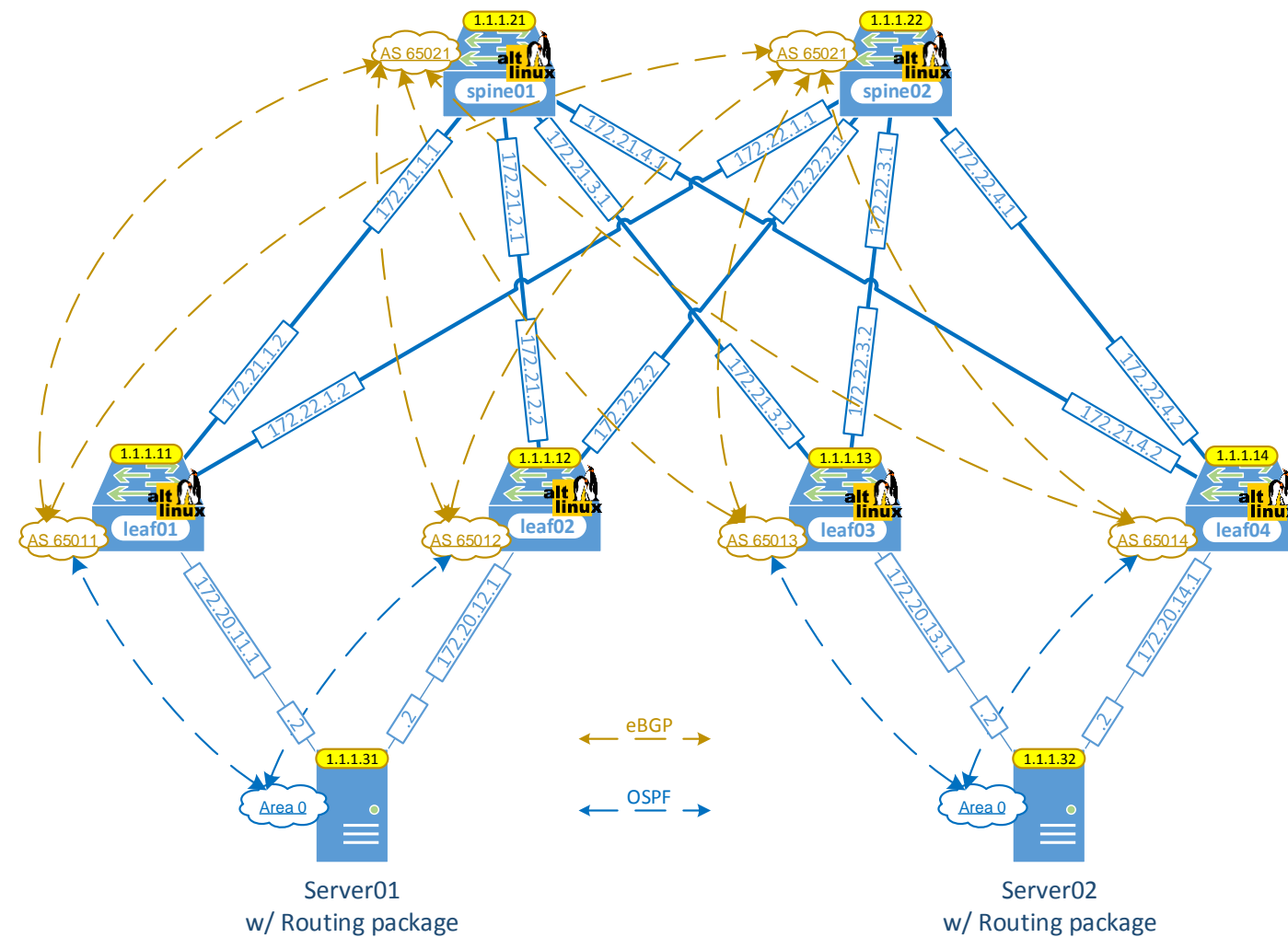


Linux-коммутаторы в реальных тестах 100GbE сети ЦОД

Mellanox & Basealt L3 "Routing on Host" PoC Physical Topology



Mellanox & Basealt L3 "Routing on Host" PoC L3 Topology



Конфигурация фабрики в Ansible: <https://github.com/kvadrage/altdemo-provision>
Виртуальная топология в Vagrant: <https://github.com/kvadrage/altdemo-vagrant>

■ Platform management

- В mlxsw большая часть реализована в hwmon драйвере
- User-space скрипты для расширенных функций

■ L3 Routing Stack

- Динамическая маршрутизация
- Реализуется в User-space (использует Netlink для формирования FIB в ядре)
- Открытые реализации: Quagga, Bird, FRR, XORP, GoBGP, SnapRoute (FlexSwitch) и др.

■ Configuration management database (CMDB)

- Стандартные решения в Linux чаще всего не подходят (ifupdown, network-scripts и т.п.)
- Нужна stateful логика (state machine)
- Изменение конфигурации не должно влиять на проходящий трафик (add/del vlan, add/del bridge и т.п.)
- Сохранение и восстановление конфигурации

■ Поддержка ONIE загрузчика

- Стандартный способ установки и загрузки NOS

■ Защита раздела с NOS

- Dual image

■ Механизмы обновления NOS

- Linux-style обновления (apt/yum upgrade)
- Обновление из единого образа системы

■ Резервирование и восстановление

- Системы
- Конфигурации

■ Мониторинг и управление коммутатором

- Классические интерфейсы: CLI, Web, API
- Мониторинг: SNMP, sFlow

■ Дополнительный функционал

- Netconf, REST API, OpenFlow, MLAG, интеграция с системами управления/оркестрации
- В зависимости от позиционирования решения

- **Доверенные среды**
 - Инфраструктуры с повышенными требованиями к безопасности и открытости
 - Финансовый сектор, Госсектор, Силовые ведомства
 - Требования к сертифицируемости решения
- **ЦОДы, где нужен Linux end-to-end**
 - CDN, IXP
 - IT Cloud, Telco Cloud
- **Web-scale инфраструктуры с минимальным TCO**
 - Web-компании со своей инфраструктурой
 - Большие масштабируемый Layer-3 фабрики
- **ОЕМ-производители**
 - Linux+Switchdev, как быстрый способ создать свою NOS
 - Эффективный путь к локализации
- **Учебные, исследовательские сети**
 - Университеты, НИИ, исследователи в области сетевых технологий, SDN
 - Экспериментальные сетевые стеки, протоколы, топологии
 - Возможность кастомизации сетевого стека и даже самого драйвера ASIC
- **Встроенные решения коммутации**
 - Сетевая фабрика для СХД
 - Blade-коммутаторы
- **Сектор Media & Entertainment, Broadcasting**
 - Миграция от SDI на IP
 - Flow-based forwarding (OpenFlow, TC Flower)



Свобода

- Любой дистрибутив Линукса
- Избавления от привязанности к вендору железа
- Установка любого сетевого приложения Linux



Простота

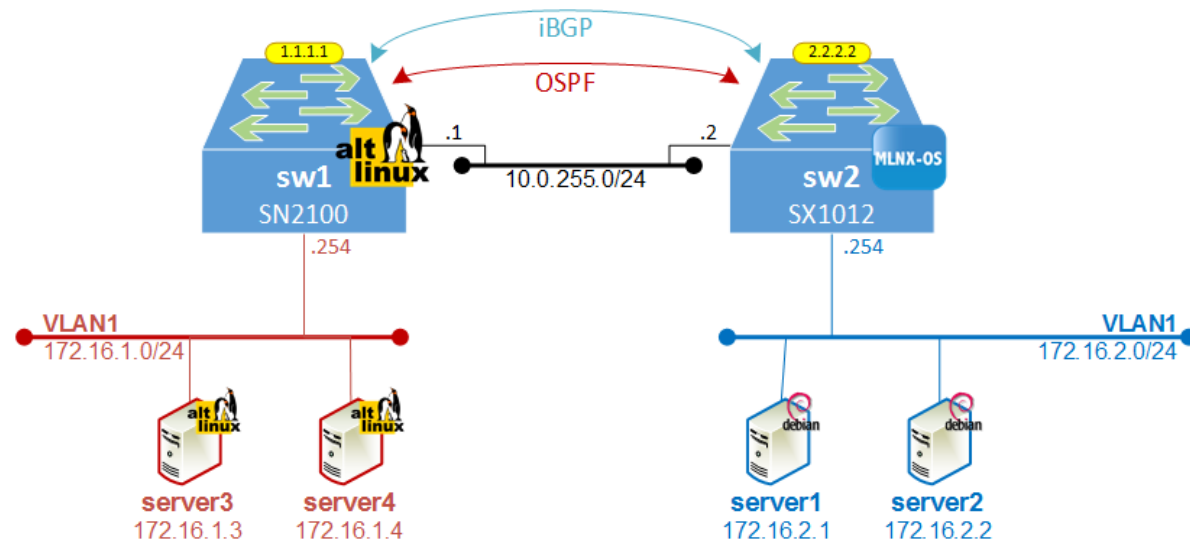
- Именно тот функционал который вам нужен (без навязанных добавлений)
- Switch as standard Linux server - easier to manage
- Портирование софта без лишних усилий



Ценность

- Поддержка High-end коммутаторов 10/25/40/50/100GbE
- 100% бесплатные open-source приложения
- Упрощенная сертификация ПО

Демонстрация Linux Switch на базе Mellanox Spectrum



```
Last login: Wed Apr 19 01:28:55 2017 from 192.168.224.2
Mellanox
Good morning, root

=====
- Hostname.....: sw1
- OS.....: ALT Sisyphus (unstable) (sisyphus)
- HW Manufacturer.....: Mellanox Technologies Ltd.
- HW Information.....: Model: "MSN2100-CB2F" S/N: "MT1640X00207"
- Users.....: Currently 1 user(s) logged on

=====
- Current user.....: root
- CPU usage.....: 0.00, 0.00, 0.00 (1, 5, 15 min)
- Memory used.....: 704 MB / 7942 MB
- Swap in use.....: 0 MB
- Processes.....: 148 running
- System uptime.....: 12 days 2 hours 26 minutes 15 seconds

=====
- Interfaces.....: 5 up, 11 down
- IP Forwarding.....: IPv4: Yes, IPv6: No

Type sudo vtysh to enter Quagga CLI

[root@sw1 ~]#
```

```
[buy@server3 ~]$ iperf -c 172.16.1.4 -i 1 -P 8 -t 1
-----
Client connecting to 172.16.1.4, TCP port 5001
TCP window size: 85.0 KByte (default)
-----
[ 4] local 172.16.1.3 port 37354 connected with 172.16.1.4 port 5001
[ 5] local 172.16.1.3 port 37356 connected with 172.16.1.4 port 5001
[ 3] local 172.16.1.3 port 37358 connected with 172.16.1.4 port 5001
[ 7] local 172.16.1.3 port 37362 connected with 172.16.1.4 port 5001
[ 8] local 172.16.1.3 port 37364 connected with 172.16.1.4 port 5001
[ 9] local 172.16.1.3 port 37366 connected with 172.16.1.4 port 5001
[10] local 172.16.1.3 port 37368 connected with 172.16.1.4 port 5001
[ 6] local 172.16.1.3 port 37360 connected with 172.16.1.4 port 5001
[ ID] Interval          Transfer          Bandwidth
[ 4] 0.0- 1.0 sec      570 MBytes      4.78 Gbits/sec
[ 5] 0.0- 1.0 sec      685 MBytes      5.75 Gbits/sec
[ 3] 0.0- 1.0 sec      512 MBytes      4.30 Gbits/sec
[ 9] 0.0- 1.0 sec      586 MBytes      4.91 Gbits/sec
[ 9] 0.0- 1.0 sec      586 MBytes      4.89 Gbits/sec
[10] 0.0- 1.0 sec      554 MBytes      4.65 Gbits/sec
[10] 0.0- 1.0 sec      554 MBytes      4.64 Gbits/sec
[ 6] 0.0- 1.0 sec      550 MBytes      4.61 Gbits/sec
[SUM] 0.0- 1.0 sec    4.49 GBytes    38.4 Gbits/sec
```



Спасибо!