Интеграция Cisco ACI с контейнерными платформами

Развёртывание, детали функционирования, диагностика

Александр Скороходов
Архитектор Cisco по технологиям
Октябрь 2020

Напоминание…

Что такое Kubernetes (K8s)?

- Kubernetes - система оркестрации для контейнеров; автоматизирует их развертывание, масштабирование и управление контейнеризированными приложениями
- Основана на опыте Google по разработке её внутренней системы Google Borg System, в 2015 Google передала как Open Source проект в Cloud Native Computing Foundation (CNCF)
- Сообщество Kubernetes выпускает новый релиз примерно каждые три месяца, текущая версия 1.19 (на октябрь 2020)
- Поддерживается различными облачными средами
- Богатая экосистема плагинов для планирования, хранения, сети

Red Hat OpenShift Container Platform (OCP)

- OCP – контейнерная платформа с функциональностью platform as a service (PaaS).
- Коммерческое решение Red Hat, основанное на open source проекте OKD (ранее OpenShift Origin).
- OCP построена на основе Kubernetes и оптимизирована для непрерывной разработки приложений и multi-tenant deployment.
- OCP включает набор дополнительных функций безопасности и инструментов для разработчиков и операторов

Сетевое взаимодействие контейнеров: Container Networking Interface (CNI)

- Модульное сетевое решение для контейнеров на основе плагинов
- Все CNI плагины работают по-разному, но в целом отвечают за:
  - Передачу пакетов внутри и между хостами (обычно с использованием оверлеев)
  - Знание того, где находится каждый Pod
  - Функции IPAM
  - Реализацию политик безопасности (опционально)
- Архитектура CNI используется Kubernetes: http://blog.kubernetes.io/2016/01/why-Kubernetes-doesnt-use-libnetwork.html

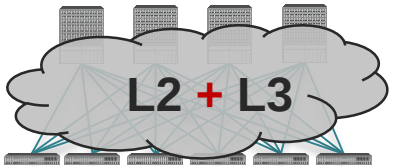
Сетевая реализация в OpenShift

- 3 режима встроенной сетевой поддержки:
  - **ovs-subnet**: «плоская» сеть, где любой pod может взаимодействовать с любым pod и service.
  - **ovs-multitenant**: изоляция pod и service между проектами
  - **ovs-networkpolicy**: администраторы проектов настраивают свои политики изоляции с использованием объектов NetworkPolicy
- Или использование стороннего SDN решения с использованием CNI Plugin - например, ACI CNI Plugin

Cisco ACI для контейнерных сред

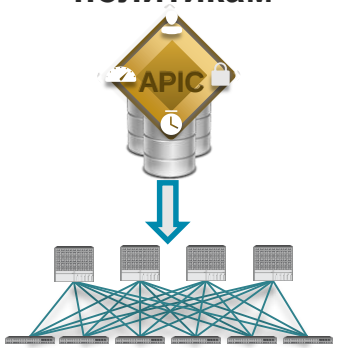
Ключевые аспекты Cisco ACI

Современная сетевая фабрика



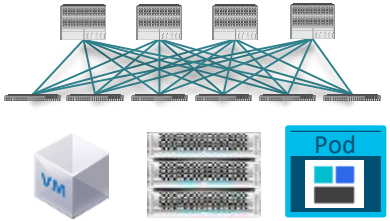
- Коммутация L2+L3 через опорную IP сеть
- Распределенный «шлюз по умолчанию»
- Хостовые маршруты (/32): оптимизация и уход от фладинга
- Внедрение в одном и во многих ЦОД

Центральное управление по политикам



- Единое управление через GUI, CLI, API
- Развёртывание, настройка и мониторинг: FCAPS
- Декларативный подход

Физические серверы + VM + контейнеры

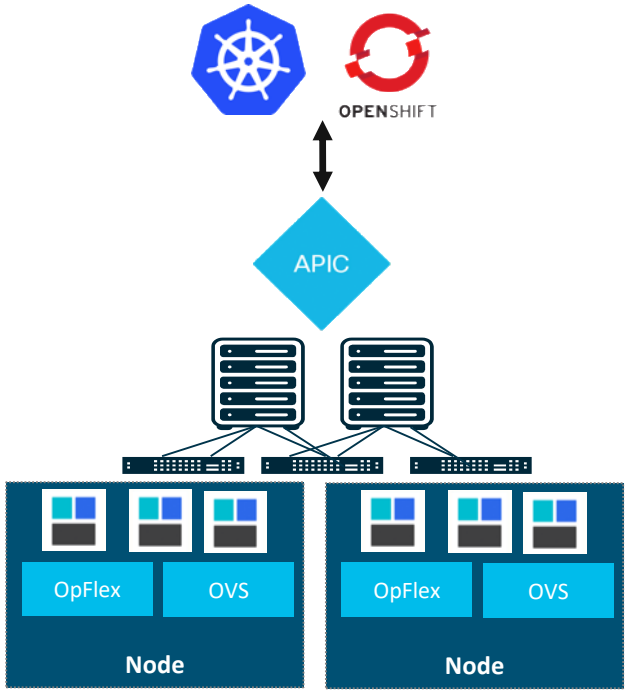


- Интеграция со средами виртуализации
- Поддержка контейнерных платформ
- Сетевой транспорт, безопасность, мониторинг

Интегрированные политики безопасности

- Принцип «белого списка» по умолчанию
- Политики безопасности, независимые от адресов
- Микросегментация
- Сегментация транспорта и управления

Cisco ACI и интеграция с контейнерами



ACI и контейнеры

- Единая сеть : контейнеры, BM и физические серверы с распределённой коммутацией L2/L3
- Балансировка нагрузки на внешние сервисы средствами фабрики для обеспечения производительности и HA, опционально SNAT
- Изоляция при помощи multi-tenancy, а так же интеграция правил доступа Kubernetes Network Policy с политиками ACI
- Видимость и статистика в APIC контроллере на уровне namespace, deployment, service, pod

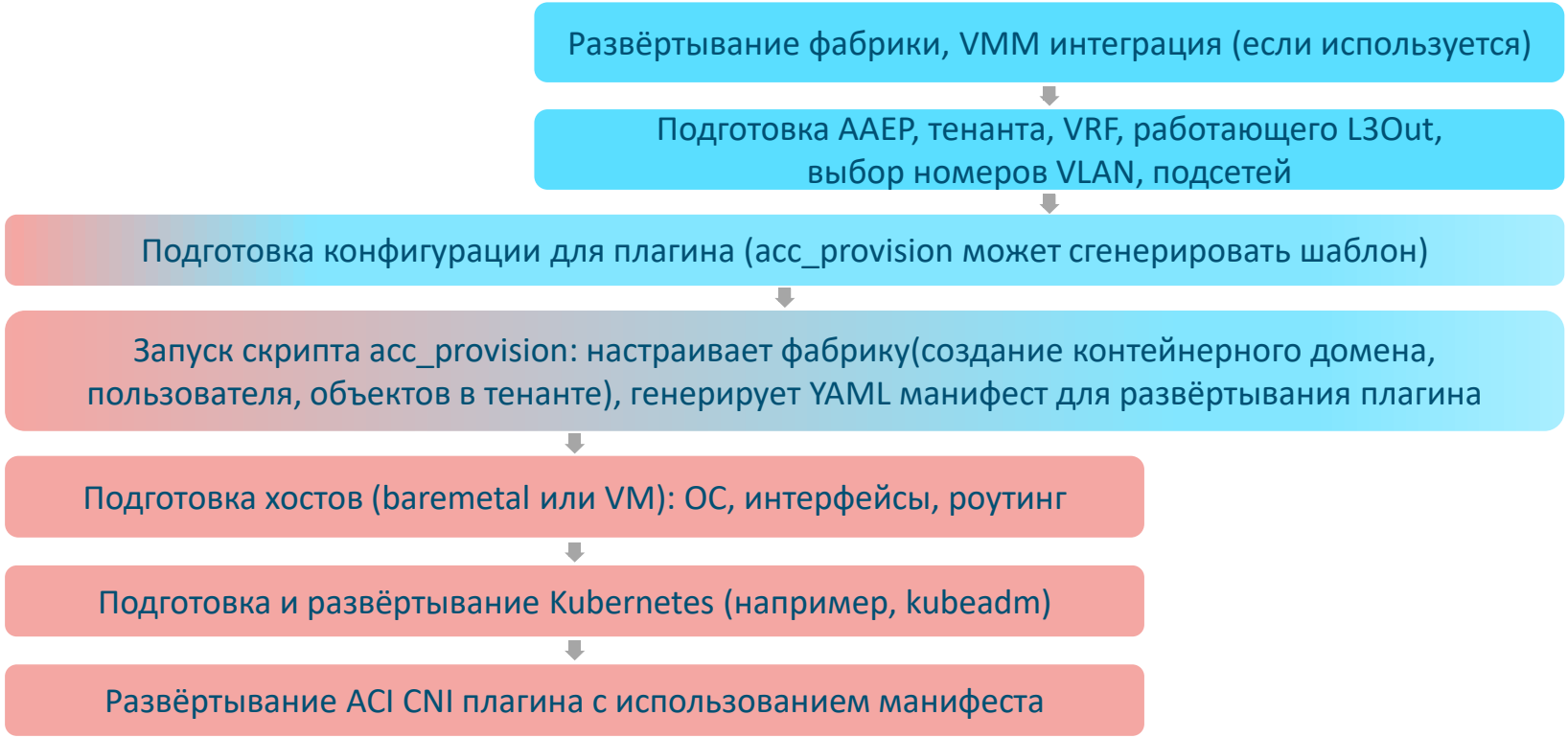
Официальная матрица поддержки ACI для виртуализации и контейнеров

https://www.cisco.com/c/dam/en/us/td/docs/Website/datacenter/aci/virtualization/matrix/virtmatrix.html

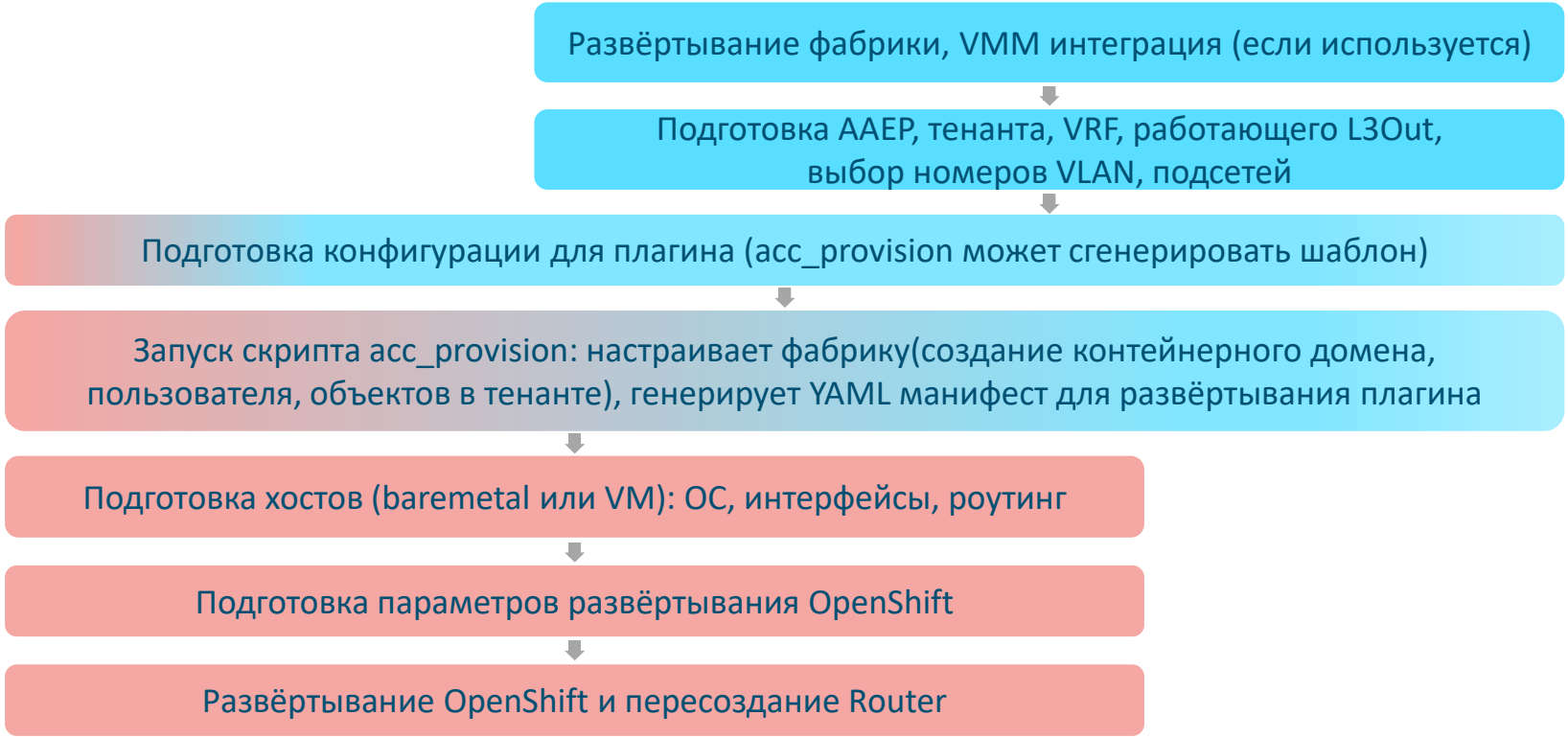
	3.0(1)	3.0(2)	3.1(1)	3.1(2)	3.2(1)	3.2(2)	3.2(3)	3.2(4) to 3.2(6)	3.2(7) to 3.2(9)	4.0(1)	4.0(2) 4.0(3)	4.1(1) 4.1(2)	4.2(1) to 4.2(3)	4.2(4)	4.2(5)	5.0(1)	5.0(2)
OpenShift 3.6	x	x			x	x	x	x	x	x	x	x	x	x	x	x	x
OpenShift 3.9	x	x	x	x	x	Note	Note	Note	Note	Note	Note	Note	Note	Note	Note	Note	Note
OpenShift 3.10	x	x	x	x	x	Note	Note	Note	Note	x	x	Note	Note	Note	Note	Note	Note
OpenShift 3.11	x	x	x	x	x	Note	Note	Note	Note	x	x	Note	Note	Note	Note	Note	Note
OpenShift 4.3	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	Note	Note
Kubernetes Upstream 1.6	Note			x	x	x	x	x	x	x	x	x	x	x	x	x	x
Kubernetes Upstream 1.7	x	x			x	x	x	x	x	x	x	x	x	x	x	x	x
Kubernetes Upstream 1.9	x	x	x	x	Note	Note	Note	x	x	x	x	x	x	x	x	x	x
Kubernetes Upstream 1.10	x	x	x	x	Note	Note	Note	Note	Note	Note	x	x	x	x	x	x	x
Kubernetes Upstream 1.11	x	x	x	x	x	Note	Note	Note	Note	Note	Note	x	x	x	x	x	x
Kubernetes Upstream 1.12	x	x	x	x	x	Note	Note	Note	Note	Note	Note	Note	x	x	x	x	x
Kubernetes Upstream 1.13	x	x	x	x	x	Note	Note	Note	Note	x	x	Note	Note	x	x	x	x
Kubernetes Upstream 1.14	x	x	x	x	x	x	x	x	x	x	x	x	Note	x	x	x	x
Kubernetes Upstream 1.15	x	x	x	x	x	x	x	x	x	x	x	x	Note	Note	Note	Note	Note
Kubernetes Upstream 1.16	x	x	x	x	x	x	x	x	x	x	x	x	Note	Note	Note	Note	Note
Kubernetes Upstream 1.17	x	x	x	x	x	x	x	x	x	x	x	x	Note	Note	Note	Note	Note
Docker Enterprise 2.1 (UCP 3.1)	x	x	x	x	x	x	x	x	x	x	x	x	Note	Note	Note	x	x
Docker Enterprise 3.0 (UCP 3.2)	x	x	x	x	x	x	x	x	x	x	x	x	Note	Note	Note	Note	Note

Развёртывание интеграции

Последовательность развёртывания (Kubernetes)



Последовательность развёртывания (OpenShift)



Подсети, VLANы, интерфейсы

Подсеть	Параметр инсталляции K8S/OpenShift	VLAN до хоста/VM	Адреса	EPG / BD	Интерфейс на хостах/ VM	Маршрутизация на хостах/VM	Видна за пределами фабрики
Node	--	Node VLAN (kubeapi_vlan)	Вручную	kube-nodes/ kube-node-bd	Да	Default route (рекомендуется)	Да (обычно)
Pod (pod_subnet)	Да	--	CNI	kube-default (и др)/ kube-pod-bd	Нет	--	Нет (обычно)
External service (extern_dynamic, extern_static)	Да	--			Нет	--	Да
Cluster IP subnet	Да	--			--	--	Нет
Node service (node_svc_subnet)	--	Service VLAN (service_vlan)		<shadow epg>/ xxxx_bd_kubernetes -service	Нет	--	Нет
SNAT адреса	--	--			Нет	--	Да
ACI Infra	--	Infra VLAN (infra_vlan)	Fabric DHCP		Да	224.0.0.0/4	Нет

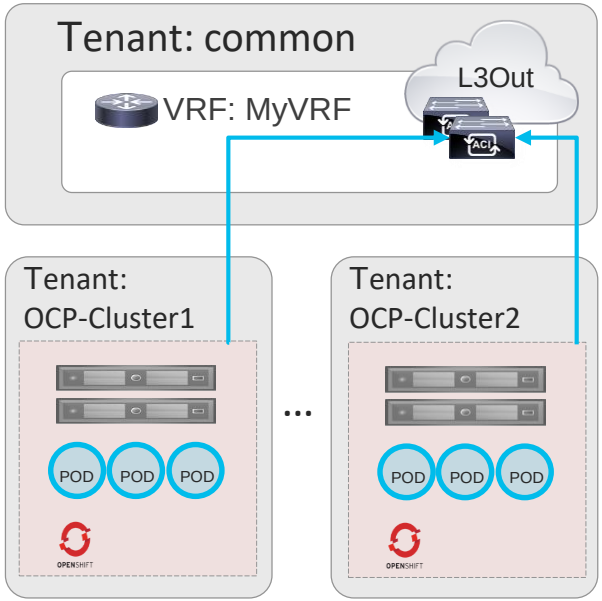
Красным помечены параметры конфигурационного файла для **acc_provision**
Имена BD/EPG с новыми версиями могут отличаться

Подсети, VLANы, интерфейсы

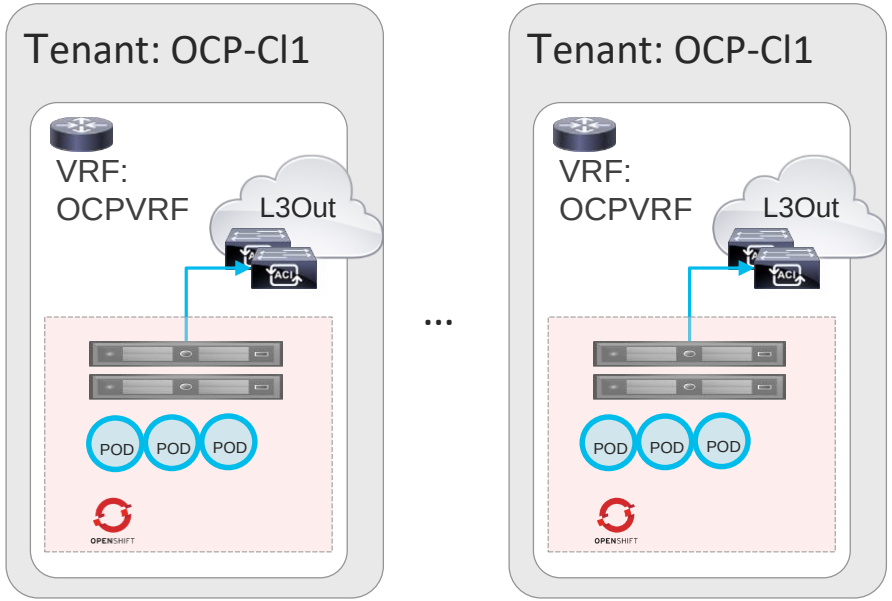
Подсеть	**Параметр K8S, значение kubeadm по умолчанию**	**Параметр OpenShift**
Pod (pod_subnet)	pod-network-cidr	osm_cluster_network_cidr 10.128.0.0/14
External service (extern_dynamic)	--	OpenShift_master_ingress_ip_network_cidr / ingressIPNetworkCIDR 172.29.0.0/16
External service (extern_static)	--	Не поддерживается при интеграции с Opehshift
Cluster IP subnet	service-cidr 10.96.0.0/12	OpenShift_portal_net 172.30.0.0/16

Подготовка ACI фабрики
2 модели внедрения

Вариант 1:
Кластер в отдельном тенанте,
разделяемый VRF и L3Out в common



Вариант 2:
Кластер в отдельном тенанте,
выделенный VRF и L3Out



Начиная с плагина версии 5.0 поддерживается несколько кластеров в одном тенанте

Подготовка конфигурации для плагина
(acc_provision может сгенерировать шаблон)

На любом Linux хосте с инсталлированным пакетом acc-provision:

```
user@service:~$ acc-provision  --list-flavors
INFO: Available configuration flavors:
…..
INFO: kubernetes-1.17:  Kubernetes 1.17
…..
user@service:~$ acc-provision -f kubernetes-1.17 --sample > cni_config_sample.yaml
```

```
aci_config:
  system_id: K8S_Demo            # Every opflex cluster must have a distinct ID
  apic_hosts:                    # List of APIC hosts to connect for APIC API
  - 10.5.30.143
  - 10.5.30.144
  - 10.5.30.145
  vmm_domain:                    # Kubernetes container domain configuration
    encap_type: vxlan            # Encap mode: vxlan or vlan
    mcast_range:                 # Every opflex VMM must use a distinct range
      start: 225.120.1.1
      end: 225.120.255.255
    mcast_fabric: 225.1.2.6
    nested_inside:               # Include if nested inside a VMM;
      type: vmware               # Specify the VMM vendor (supported: vmware)
      name: ACI_DVS              # Specify the name of the VMM domain
```

Подготовка конфигурации для плагина
…продолжение настроек

```
# The following resources must already exist on the APIC.
  aep: ESX-AAEP                  # The AEP for ports/VPCs used by this cluster
  vrf:                           # This VRF used to create all kubernetes EPs
    name: VRF1
    tenant: K8S_Demo             # This can be system-id or common
  l3out:
    name: L3Out                  # Used to provision external IPs
    external_networks:
    - ExtEPG                     # Used for external contracts

net_config:      # Networks used by ACI containers
  node_subnet: 10.5.32.169/29    # Subnet to use for nodes
  pod_subnet: 10.200.0.1/24      # Subnet to use for Kubernetes Pods/
  extern_dynamic: 10.5.32.153/29 # Subnet to use for dynamic external IPs
  extern_static: 10.5.32.161/29  # Subnet to use for static external IPs
  node_svc_subnet: 10.201.0.1/24 # Subnet to use for service graph
  kubeapi_vlan: 3903             # The VLAN used by the physdom for nodes
  service_vlan: 3904             # The VLAN used by LoadBalancer services
  infra_vlan: 3900               # The VLAN used by ACI infra
  interface_mtu: 1600            # min = 1280 for ipv6, max = 8900 for VXLAN
registry:        # Configuration for container registry
  image_prefix: noiro            # e.g: registry.example.com/noiro
kube_config:
  ovs_memory_limit: "20Gi"       # override if needed, default is "1Gi“
istio_config:
  install_istio: False           # default is True
```

Запуск скрипта acc-provision:
настраивает фабрику(создание контейнерного домена, пользователя), генерирует YAML манифест для развёртывания плагина

- Скрипт acc-provision – вся необходимая подготовка к интеграции

```
$ acc-provision -c config.yaml -o deploy.yaml -f kubernetes-1.17 -a -u admin -p P@ssw0rd
```

- Получает на вход файл настроек (config.yaml) – см предыдущие слайды
- Генерирует манифест (deploy.yaml) для развёртывания ACI CNI плагина на кластере Kubernetes
- Настраивает фабрику:
  - Создаёт VMM домен и связанные с ним пулы, добавляет его в AEP
  - Создаёт BD/AP/EPG, контракты и прочее в указанном тенанте
  - Создаёт пользователя с нужными полномочиями и доступом по сертификату

Подготовка хостов (baremetal или VM):
ОС, интерфейсы, роутинг

- Поддерживаются baremetal хосты и VM, подключенные к ACI VMM домену VMWare vSphere
  - Для лабораторной среды удобнее VM – благодаря клонированию, «снимкам» (snapshots) и автоматизации сетевых подключений с помощью VMM
- Выбор ОС в зависимости от типа платформы:
  - Ubuntu 18.04+ для Kubernetes, RHEL для OpenShift
- Необходимо два субинтерфейса на интерфейсе, подключенном к ACI:
  - **Node subnet (kubeapi_vlan):** основной интерфейс ОС, шлюз по умолчанию (!!!)
  - **Infra VLAN:** выдача адреса по DHCP фабрикой, транспорт для OpFlex и VXLAN, маршрут для multicast адресов
  - Субинтерфейс для service VLAN не нужен (!!!)
  - Обеспечьте необходимый MTU
  - Опционально: интерфейс управления

Пример работающей конфигурации для Ubuntu
/etc/netplan/50-cloud-init.yaml и /etc/dhcp/dhclient-ens160.3900.conf

```
network:
    ethernets:
        ens160:
            match:
                macaddress: 00:50:56:b0:4d:0b
            optional: true
            mtu: 1600
    vlans:
        ens160.3903:
            id: 3903
            link: ens160
            addresses:
            - 10.5.32.171/29
            gateway4: 10.5.32.169
            nameservers:
                addresses:
                - 10.5.30.2
                search:
                - company.com
        ens160.3900:
            id: 3900
            link: ens160
            dhcp4: true
            routes:
            - to: 224.0.0.0/4
              scope: link
```

```
send dhcp-client-identifier 01:00:50:56:b0:4d:0b
request subnet-mask, domain-name, domain-name-servers, host-name;
send host-name gethostname();

option rfc3442-classless-static-routes code 121 = array of unsigned integer 8;
option ms-classless-static-routes code 249 = array of unsigned integer 8;
option wpad code 252 = string;

also request rfc3442-classless-static-routes;
also request ms-classless-static-routes;
also request static-routes;
also request wpad;
also request ntp-servers;
```

Сетевая среда на хостах

До инсталляции Docker/Kubernetes

```
$ ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1600 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:50:56:b0:4d:0b brd ff:ff:ff:ff:ff:ff
    inet6 fe80::250:56ff:feb0:4d0b/64 scope link
       valid_lft forever preferred_lft forever
3: ens160.3900@ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1600 qdisc noqueue state UP group default qlen 1000
    link/ether 00:50:56:b0:4d:0b brd ff:ff:ff:ff:ff:ff
    inet 10.0.232.96/16 brd 10.0.255.255 scope global dynamic ens160.3900
       valid_lft 604715sec preferred_lft 604715sec
    inet6 fe80::250:56ff:feb0:4d0b/64 scope link
       valid_lft forever preferred_lft forever
4: ens160.3903@ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1600 qdisc noqueue state UP group default qlen 1000
    link/ether 00:50:56:b0:4d:0b brd ff:ff:ff:ff:ff:ff
    inet 10.5.32.171/29 brd 10.5.32.175 scope global ens160.3903
       valid_lft forever preferred_lft forever
    inet6 fe80::250:56ff:feb0:4d0b/64 scope link
       valid_lft forever preferred_lft forever
```

Infra VLAN

Адрес по DHCP

Node VLAN (kubeapi_vlan)

```
$ ip route
default via 10.5.32.169 dev ens160.3903 proto static
10.0.0.0/16 dev ens160.3900 proto kernel scope link src 10.0.232.96
10.5.32.168/29 dev ens160.3903 proto kernel scope link src 10.5.32.171
224.0.0.0/4 dev ens160.3900 proto static scope link
```

0/0 через Node подсеть

Multicast через Infra VLAN

Работа через прокси-сервера под Linux
Не специфично для контейнеров или ACI CNI, но…

- Источник доброй половины проблем при развёртывании!
- Некоторые программы используют переменные среды http_proxy/ https_proxy, некоторые HTTP_PROXY/HTTPS_PROXY
- Обязательно исключайте локальные адреса через no_proxy/NO_PROXY
  - Некоторые программы понимают CIDR нотацию (192.168.0.0/16), некоторые требуют перечисления исключаемых адресов списком (!)
- Определите все переменные через /etc/environment
- Для Docker создайте файл /etc/systemd/system/docker.service.d/http-proxy.conf:

```
[Service]
Environment="HTTP_PROXY=http://proxy.company.com:80/"
Environment="HTTPS_PROXY=http://proxy.company.com:80/"
Environment="NO_PROXY=127.0.0.1,localhost,10.5.32.0/24"
```

Подготовка и развёртывание Kubernetes
Kubeadm

- Воспользуйтесь документацией по инсталляции kubeadm – она достаточно краткая и простая: https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/
- На этапе инсталляции пакетов – выберите нужную версию:

```
$ sudo apt-get install -y kubelet=1.17.13-00 kubeadm=1.17.13-00 kubectl=1.17.13-00
```

- При запуске kubeadm на мастер-ноде передайте желаемую Pod subnet, и, если нужно, Cluster-IP подсеть:

```
$ sudo kubeadm init --pod-network-cidr=10.200.0.1/24 --service-cidr=10.202.0.1/24
```

- В конце работы kubeadm будет показана команда kubeadm join … для запуска на других хостах, включаемых в кластер

Развёртывание плагина с использованием манифеста

- После того, как кластер собран (хосты появились в выдаче **kubectl get nodes** со статусом NotReady), разверните плагин с использованием подготовленного манифеста:

```
$ kubectl apply –f deploy.yaml
```

- По умолчанию, берет образы из Docker Hub
- Если используется локальный репозиторий (registry), его нужно указывать в конфигурационном файле для acc-provision

Особенности в случае OpenShift

- Подготовка узлов к развёртыванию – согласно документации OpenShift
- Настройка субинтерфейсов и подготовка конфигурации ACI и манифеста – аналогично шагам для Kubernetes (отличие: не указывайте extern_static)
- Инсталляция ACI CNI Plugin поддерживается штатным инсталлятором OpenShift (openshift-ansible): нужно просто внести настройки в playbook (пример на следующем слайде), включая ссылку на манифест
- Координация адресов подсетей ACI и OpenShift – см таблицу ранее
- После инсталляции необходимо пересоздать router (шаги указаны в документе по интеграции ACI с OpenShift)

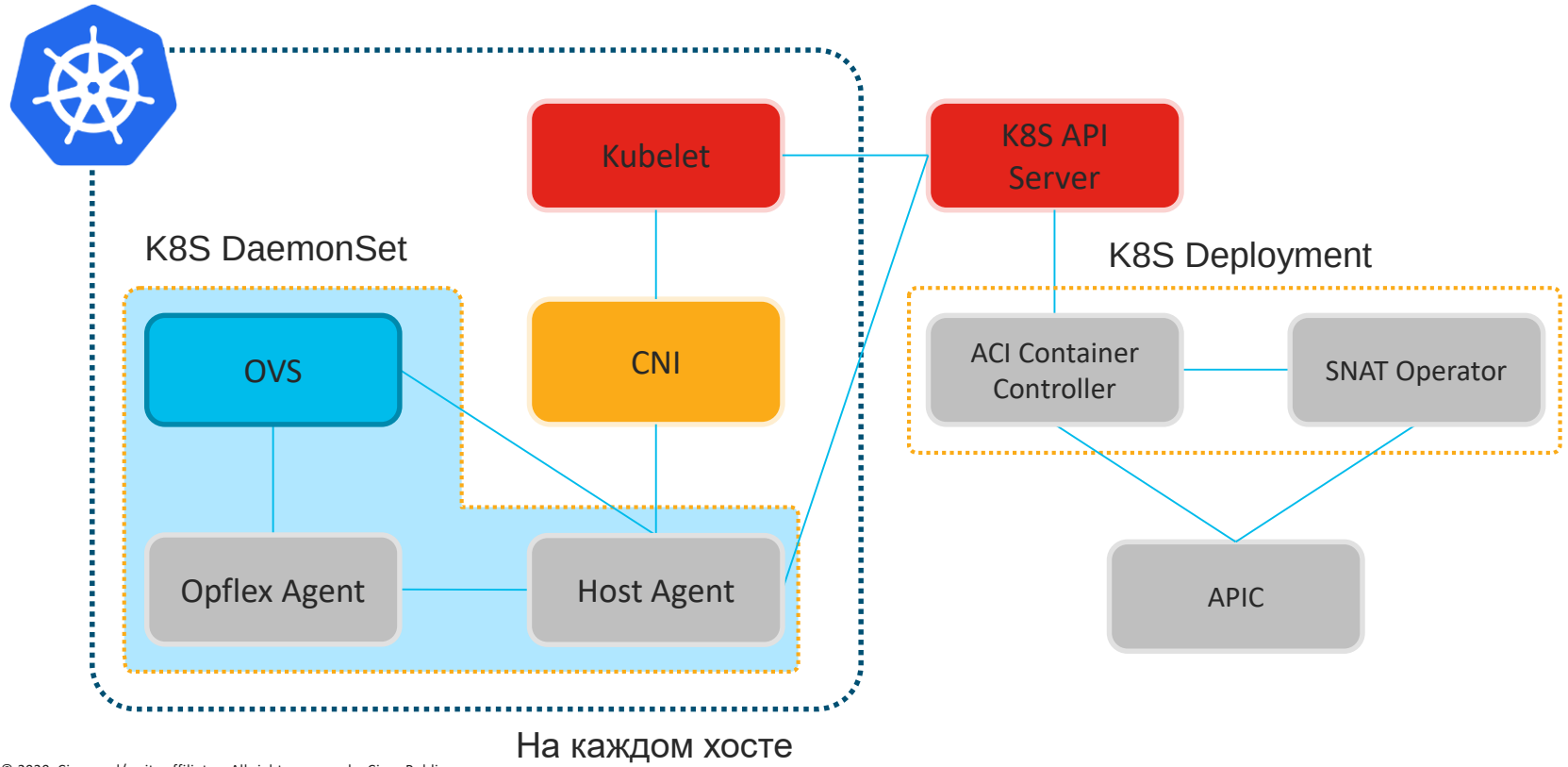
Пример конфигурации инсталлятора OpenShift
Например, в /etc/ansible/hosts

```
[OSEv3:vars]
...
os_sdn_network_plugin_name='cni'
openshift_use_openshift_sdn=False
openshift_use_aci=True
aci_deployment_yaml_file='/root/os_cni.yaml'

openshift_master_ingress_ip_network_cidr = 10.5.32.128/29
...
```

Архитектура ACI CNI плагина

Архитектура ACI CNI



Компоненты ACI CNI плагина

- aci-containers-controller
  - Deployment из одного Pod
  - Реализация функционала IPAM
  - Управление состоянием подключения конечных хостов
  - Применение политики (на основе annotations)
  - Управление балансировкой нагрузки
  - Загружает конфигурацию на APIC контроллер

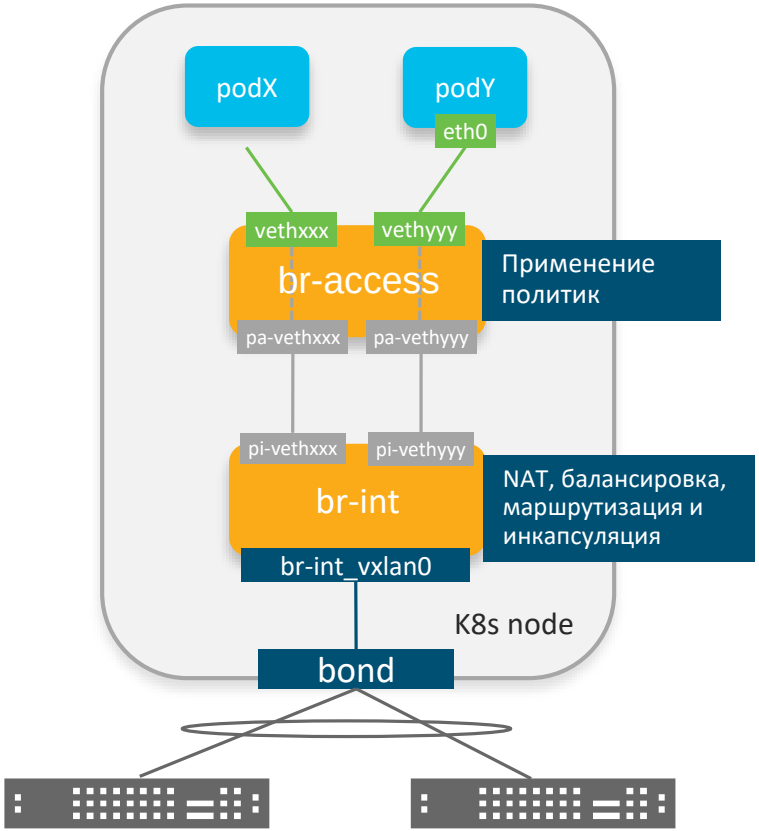
Компоненты ACI CNI плагина
На каждом хосте

- aci-containers-host: DaemonSet состоящий из 3 контейнеров:
  - aci-containers-host:
    - Endpoint metadata
    - Управление IP адресами POD
    - Настройка интерфейсов контейнеров
  - mcast-daemon:
    - Обработка Broadcast, unknown unicast и multicast
  - opflex-agent:
    - Поддержка Stateful Security Groups
    - Управление настройкой OVS
    - Рендеринг политик при помощи правил openflow на OVS
    - Управление балансировкой нагрзуки (connection tracking, NAT, и т.д.)

Компоненты ACI CNI плагина
На каждом хосте

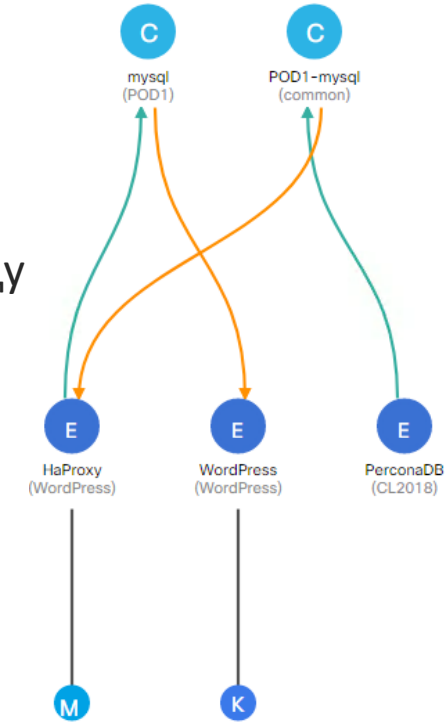
- aci-containers-openvswitch
  - Коммутирует трафик между контейнерами и физическими интерфейсами
  - Реализует политики фильтрации, балансировки, SNAT
  - Выполняет инкапсуляцию

Реализация связности и применения политик на хосте
Open vswitch (OVS)



Взаимодействие между контейнерами и не-контейнерами

- В продуктивных средах высокопроизводительные сервисы (например БД) запускаются на физических серверах или виртуальных машинах
- Возникает необходимость организации взаимодействия между контейнерами и ВМ/физическими серверами
- Просто настройте контракт между EPG, а ACI сделает все остальное
- Поддерживается между любыми VMM доменами и физическими доменами



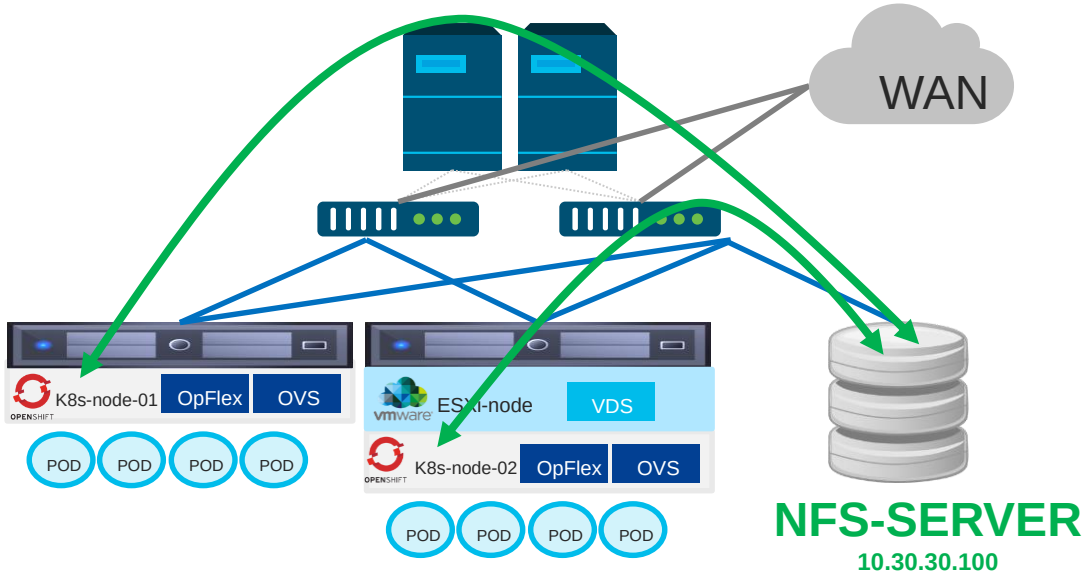
Возможность доступа контейнеров к централизованному хранению без «узких мест»

OPENSHIFT

APIC

ACI реализует распределённую коммутацию, маршрутизацию и безопасность на OVS и leaf

```
[…]
 spec:
      containers:
      - name: nginx
        image: nginx:1.7.9
        volumeMounts:
        - mountPath: /srv/www
          name: www-data
          readOnly: true
        ports:
        - containerPort: 80
      volumes:
      - name: www-data
        nfs:
           server: 10.30.30.100
           path: "/var/nfs"
```

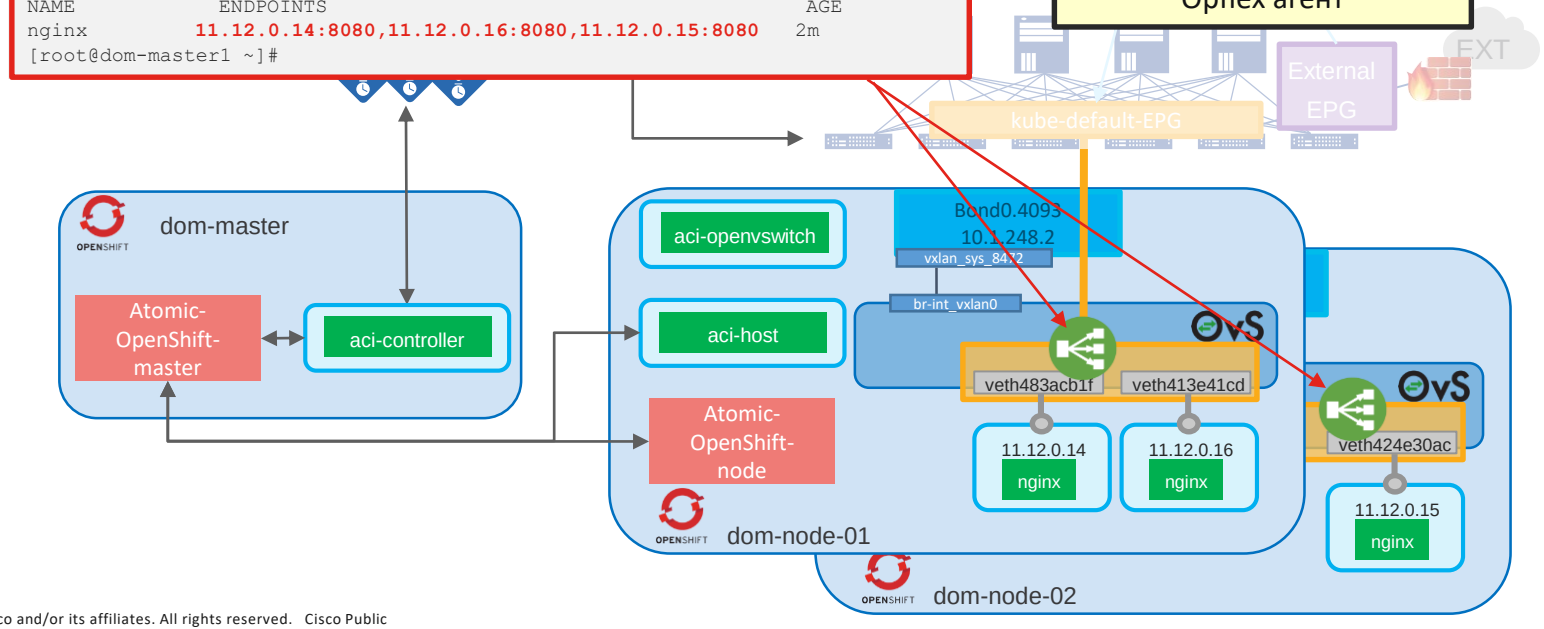


Реализация сервисов

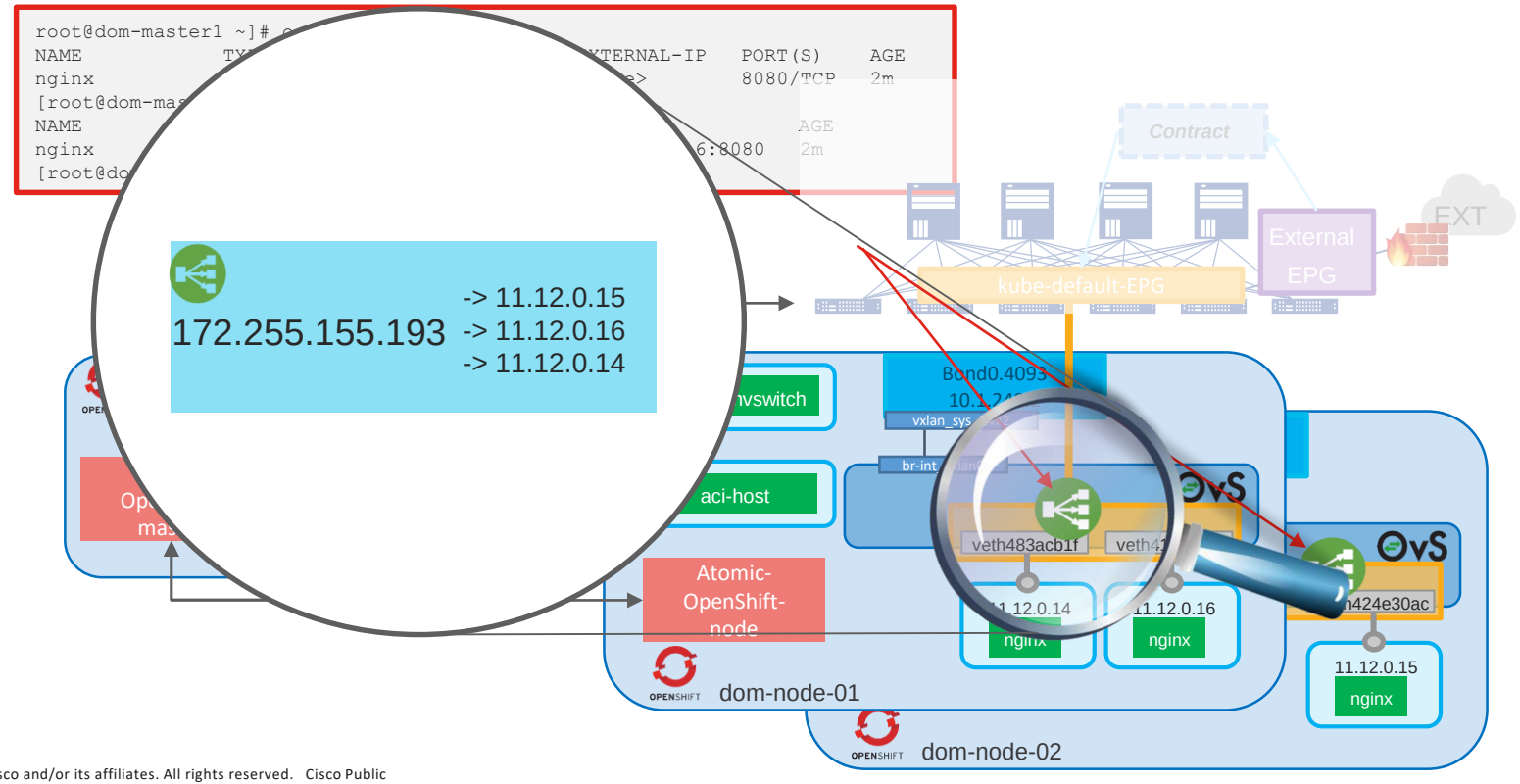
Сервисы ClusterIP
Распределённая балансировка на хостах

```
root@dom-master1 ~]# oc get service
NAME            TYPE        CLUSTER-IP        EXTERNAL-IP   PORT(S)    AGE
nginx           ClusterIP   172.255.155.193   <none>        8080/TCP   2m
[root@dom-master1 ~]# oc get endpoints
NAME            ENDPOINTS                                              AGE
nginx           11.12.0.14:8080,11.12.0.16:8080,11.12.0.15:8080        2m
[root@dom-master1 ~]#
```

Конфигурация сервиса вызывает настройку правил балансировки OVS через Opflex агент

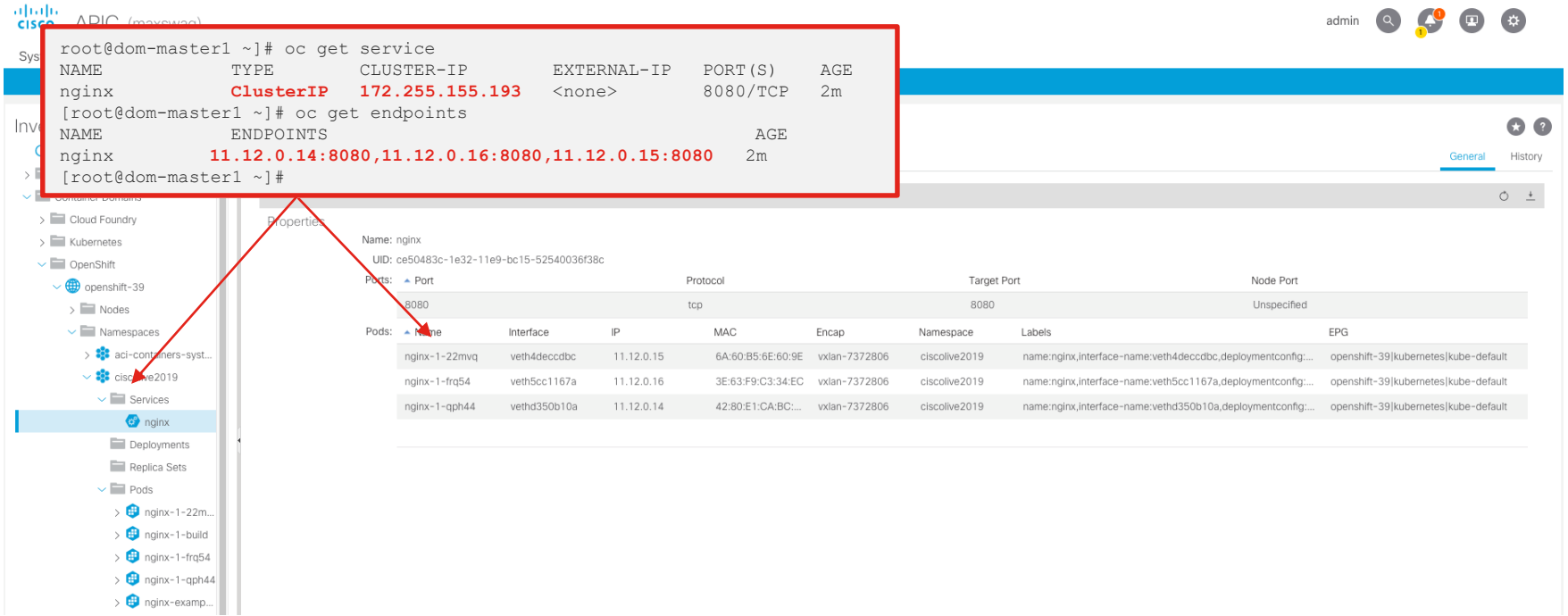


Сервисы ClusterIP
Распределённая балансировка на хостах



Сервисы ClusterIP
Видимость в GUI

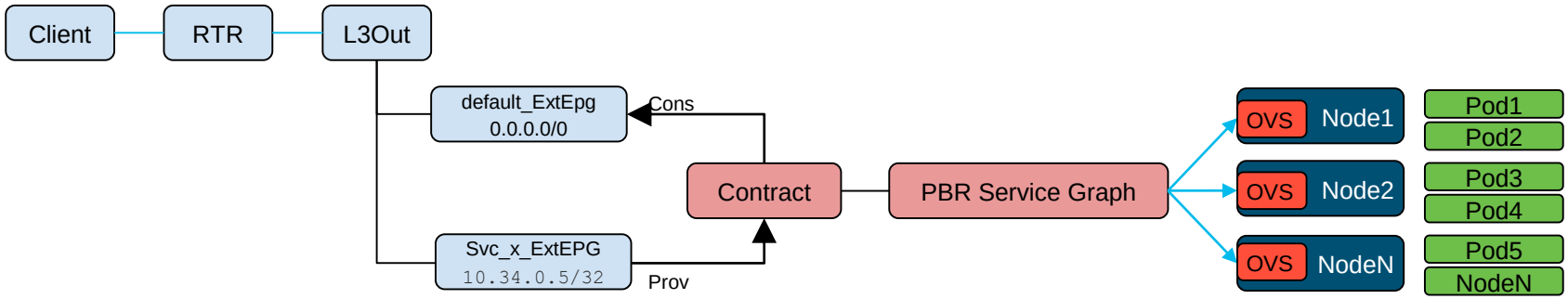
```
root@dom-master1 ~]# oc get service
NAME            TYPE        CLUSTER-IP        EXTERNAL-IP   PORT(S)    AGE
nginx           ClusterIP   172.255.155.193   <none>        8080/TCP   2m
[root@dom-master1 ~]# oc get endpoints
NAME            ENDPOINTS                                          AGE
nginx           11.12.0.14:8080,11.12.0.16:8080,11.12.0.15:8080    2m
[root@dom-master1 ~]#
```



Реализация внешних сервисов (LoadBalancer)
Сервисный граф

Каждый раз когда настраивает сервис ACI CNI контроллер настраивает:

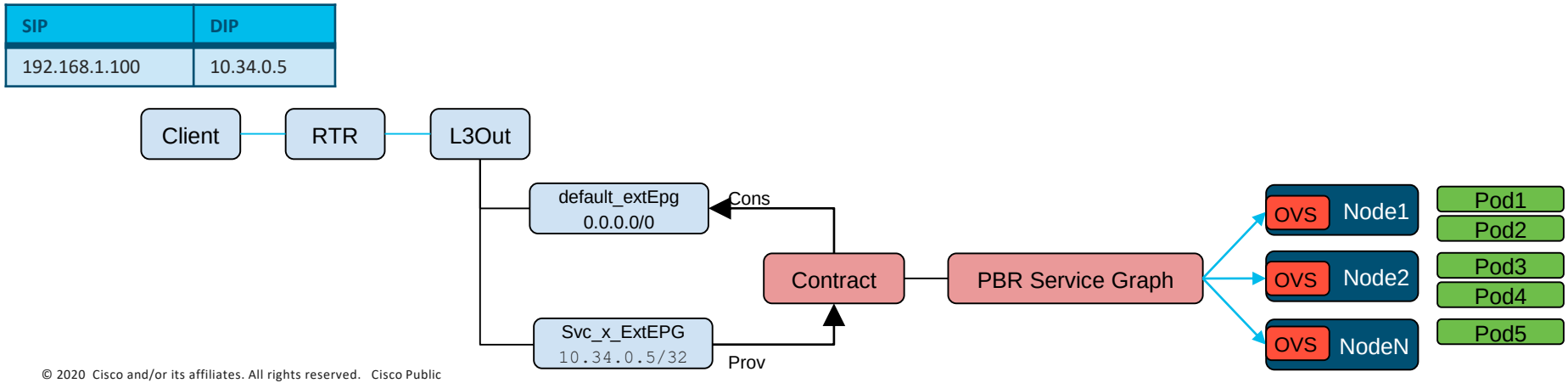
- External EPG с маской /32 и адресом Service IP
- Новый контракт между svc_ExtEPG и default_ExtEPG*
- Service Graph и PBR в котором настраиваются адреса всех POD где запущен сервис



* определяется в конфигурационном файле acc-provision

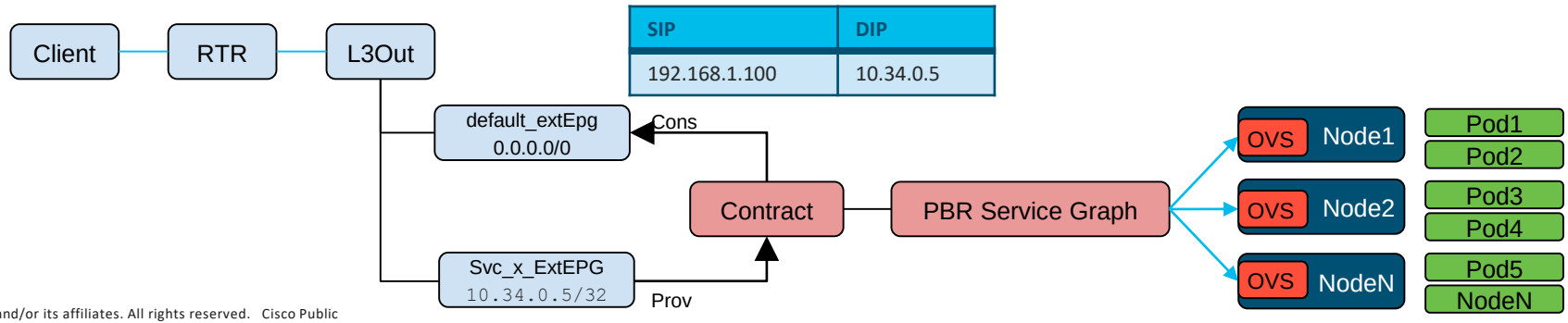
Реализация внешних сервисов (LoadBalancer)
Входящий трафик

1. Клиент шлет запрос на адрес 10.34.0.5, ACI выполняет Longest Prefix Match (LPM) по адресу источника SIP и классифицирует трафик как приходящий от default_extEPG
2. ACI выполняет routing lookup 10.34.0.5, этот IP не существует физически внутри фабрики, этот второй LPM над адресом назначения DIP классифицирует его как приходящий на Svc_x_ExtEPG
3. Срабатывает правило PBR перенаправления и трафик балансируется на один из узлов кластера K8S



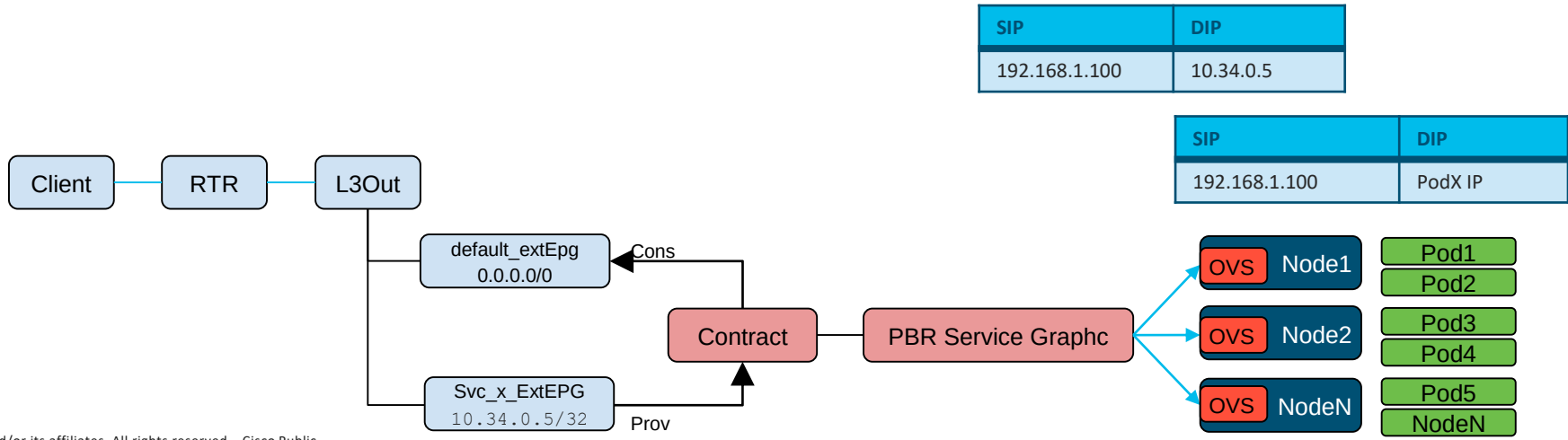
Реализация внешних сервисов (LoadBalancer)
Входящий трафик

1. Клиент шлет запрос на адрес 10.34.0.5, ACI выполняет Longest Prefix Match (LPM) по адресу источника SIP и классифицирует трафик как приходящий от default_extEPG
2. ACI выполняет routing lookup 10.34.0.5, этот IP не существует физически внутри фабрики, этот второй LPM над адресом назначения DIP классифицирует его как приходящий на Svc_x_ExtEPG
3. Срабатывает правило PBR перенаправления и трафик балансируется на один из узлов кластера K8S



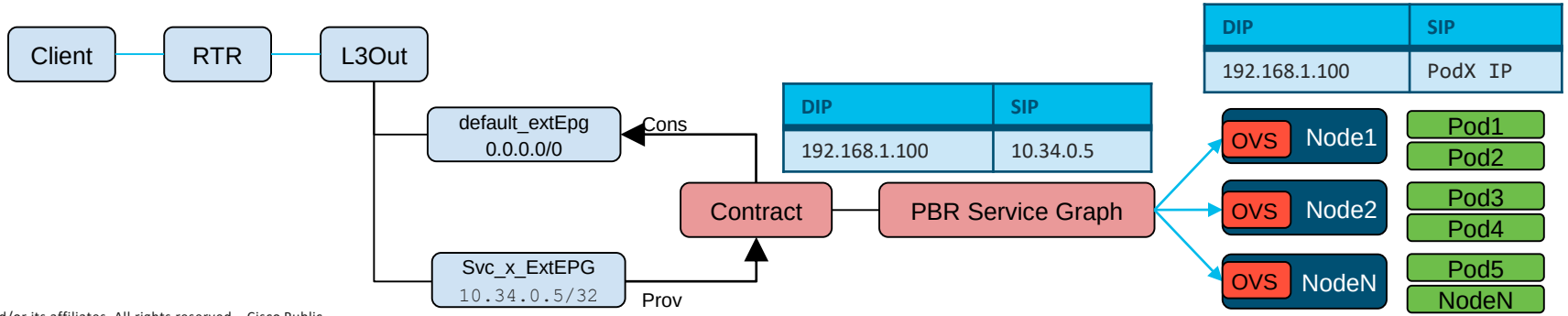
Реализация внешних сервисов (LoadBalancer)
Входящий трафик

4. OVS на хосте K8S выполняет D-NAT с адреса external service IP на адрес Pod
5. Если на одном хосте запущено несколько Pod, реализующих сервис, то OVS выполнит еще один шаг балансировки нагрузки между локальными Pod



Реализация внешних сервисов (LoadBalancer)
Исходящий трафик

6. PodX отвечает клиенту
7. OVS выполняет S-NAT и переписывает external Service IP
8. Правило PBR не срабатывает поскольку EPG источником является Shadow EPG PBR узла
9. Трафик маршрутизируется назад к клиенту (и разрешается контрактом)

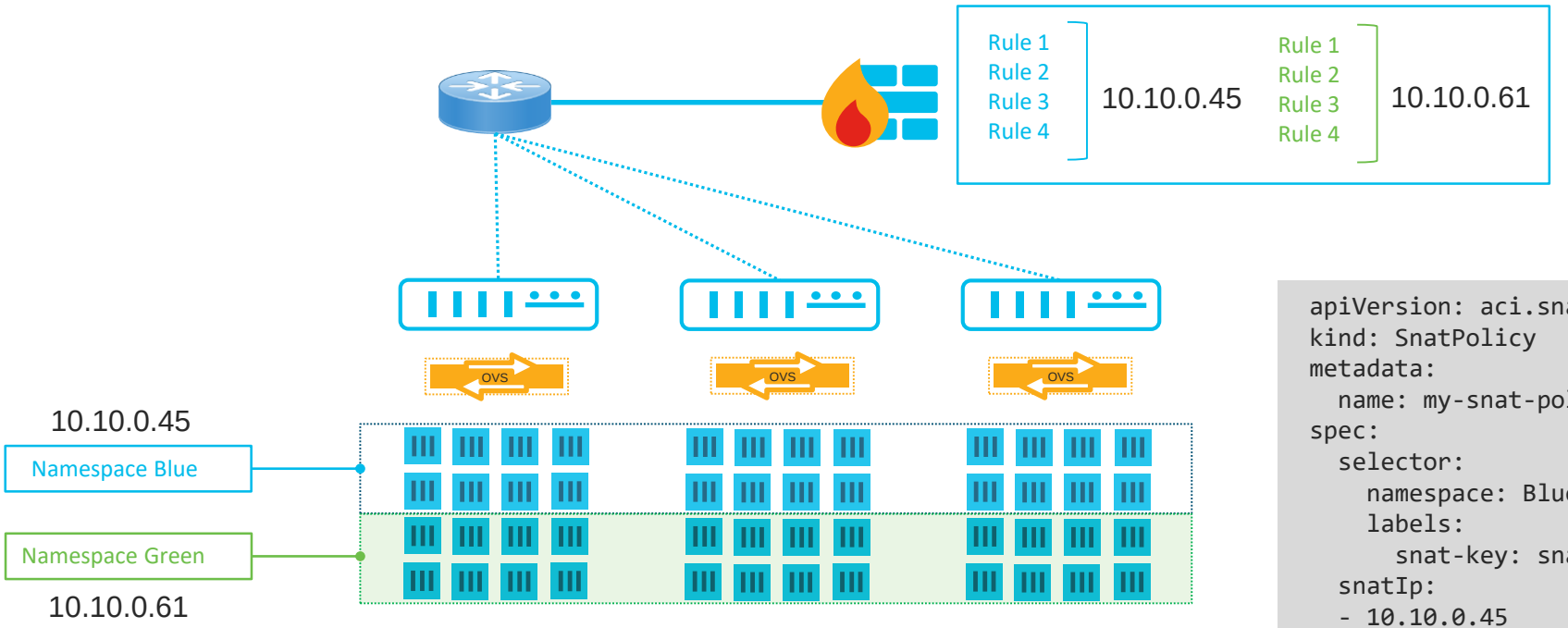


SNAT

SNAT для исходящего трафика

- Проблема в традиционной реализации:
  - Исходящий трафик Pod-ов в традиционную инфраструктуру обычно отправляется с адреса worker node
  - Нельзя определить источник (pod/deployment/namespace)
  - Проблема безопасности и диагностики
- Решение благодаря интеграции ACI с контейнерными средами:
  - Применение SNAT политик
  - Source адреса или диапазоны могут быть «привязаны» к конкретным элементам контейнерного ландшафта
  - Легко идентифицировать контейнерный трафик при выходе в традиционную инфраструктуру
  - Можно при необходимости также разрешить выход Pod IP без трансляции

Применение ACI SNAT для классификации на МСЭ



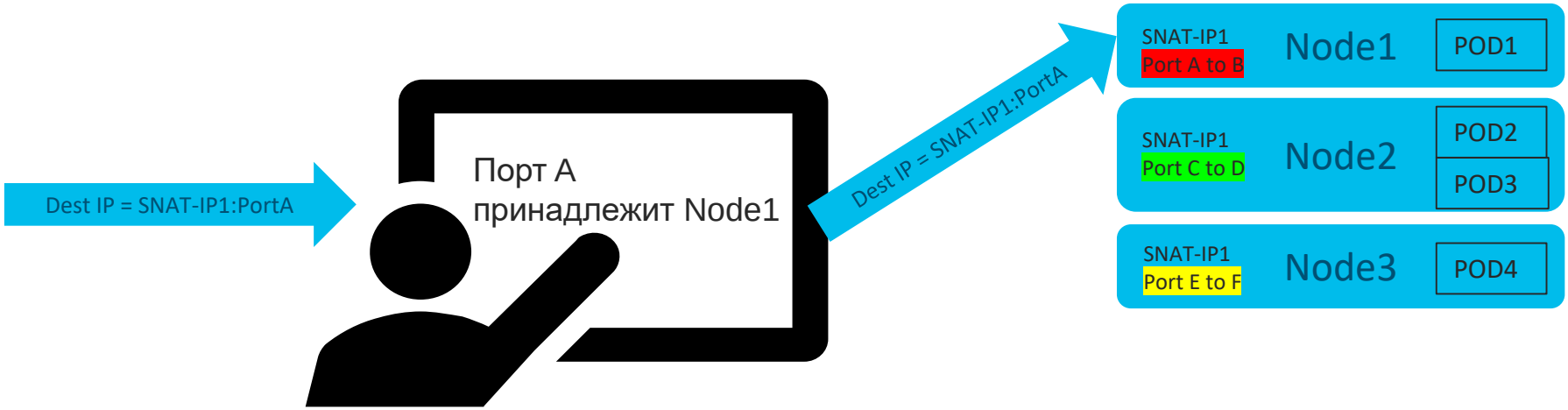

```
apiVersion: aci.snat/v1
kind: SnatPolicy
metadata:
  name: my-snat-policy
spec:
  selector:
    namespace: Blue
    labels:
      snat-key: snat-value
  snatIp:
  - 10.10.0.45
```

SNAT для исходящего трафика
Принцип реализации

- Политики SNAT описываются как Custom Resource Definitions (CRD)
- Отслеживание соединений и трансляция выполняются Open vSwitch (OVS) каждого хоста
- Правила OVS программируются Opflex агентом хоста
- aci-containers-controller программирует сервисный граф в ACI для обработки возвратного трафика
- Адреса SNAT не относятся к конкретной подсети, обеспечить маршрутизацию к ним – задача администратора
- SNAT применяется только к TCP/UDP трафику, исходящему через L3Out
  - Не к трафику внутри фабрики
  - Не поддерживается ICMP

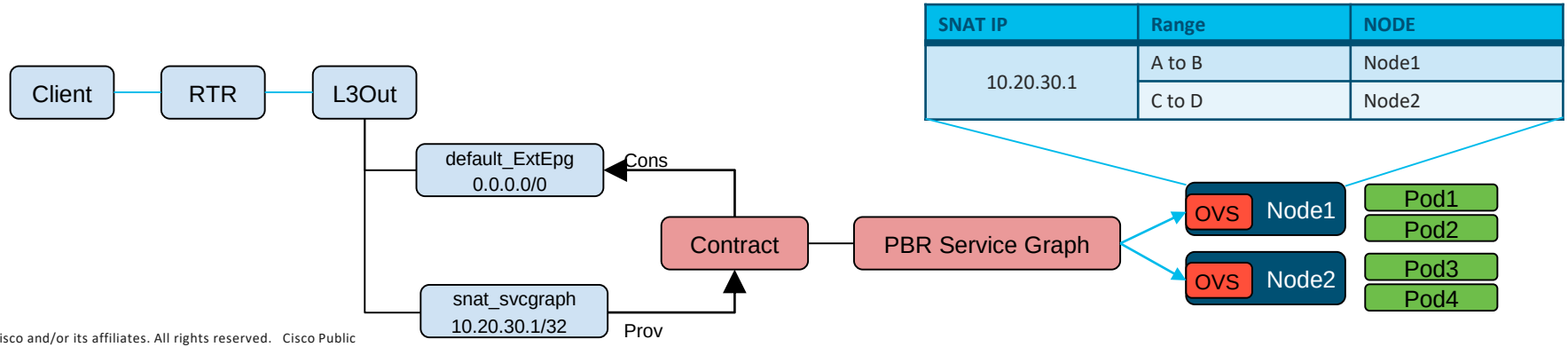
Распределённая обработка возвратного трафика

- Каждому хосту выделяется диапазон TCP и UDP портов на SNAT IP: это позволяет определить, на какой ноде расположен Pod, инициировааший соединение



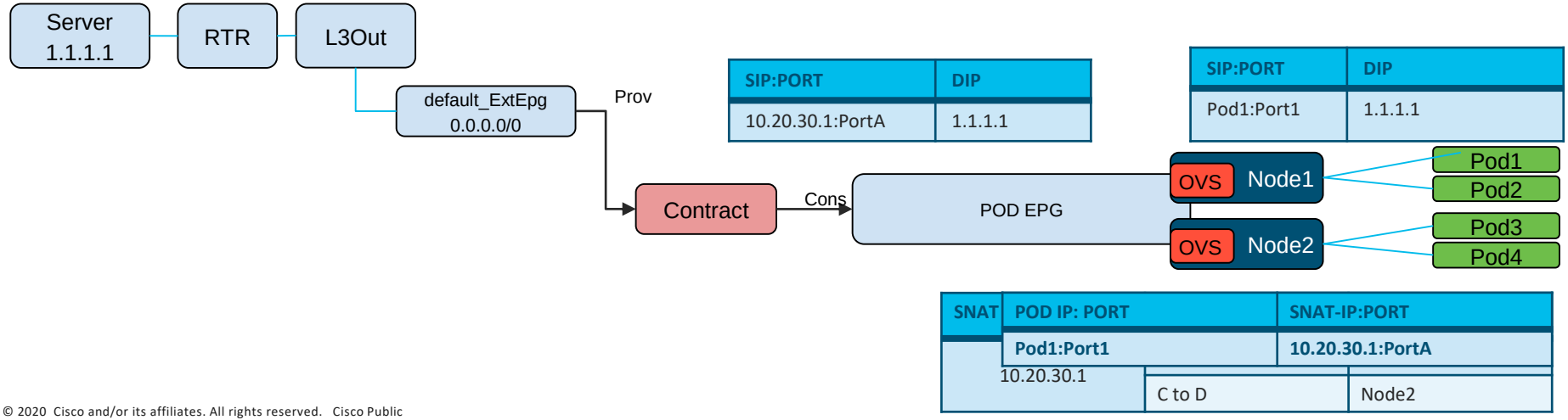
Реализация SNAT
Сервисный граф

- Когда SNAT политика настраивается в первый раз, aci-container-controller :
  - Создаёт External EPG snat_svcgraph
  - Добавляет SNAT IP под snat_svcgraph
  - Создаёт контракт snat_svcgraph между snat_svcgraph ExtEPG и default_ExtEPG
  - Создаёт сервисный граф с PBR перенаправлением на узлы кластера
  - Выделяет каждому хосту диапазон портов для SNAT IP



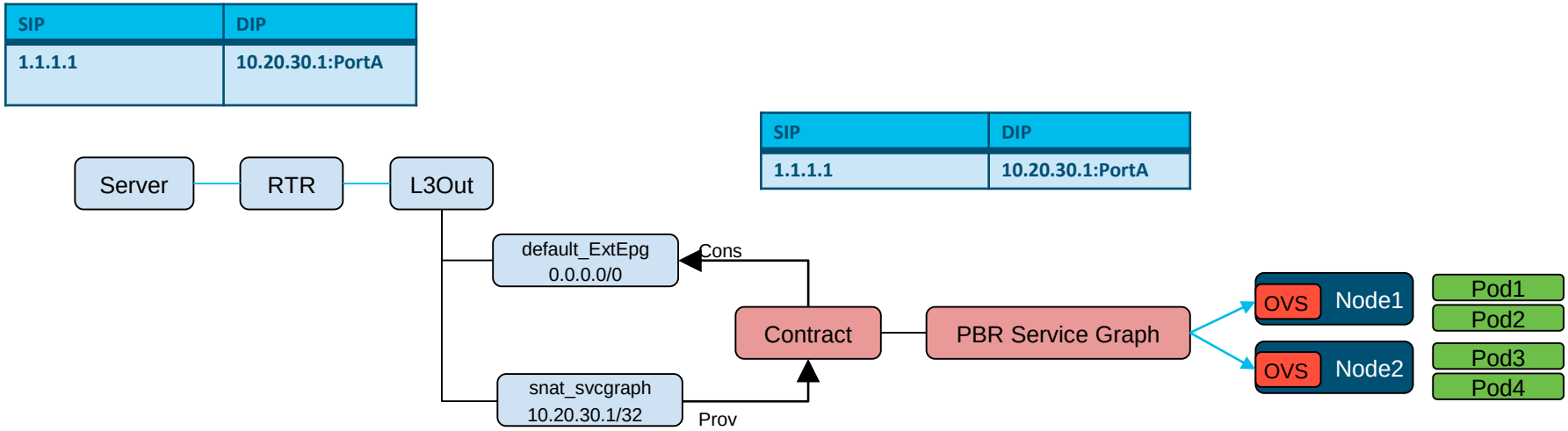
Реализация SNAT
Исходящий трафик

1. Pod1 на Node1 обращается к 1.1.1.1
2. OVS проверяет, что коммуникация разрешена контактами ACI/политиками K8S и требует SNAT
3. OVS выбирает SNAT-IP 10.20.30.1 и порт A из диапазона данного хоста
4. OVS транслирует Pod IP в SNAT-IP и отслеживает соотношение POD-IP:Port и SNAT-IP:SNAT-Port
5. Происходит обычная передача в ACI



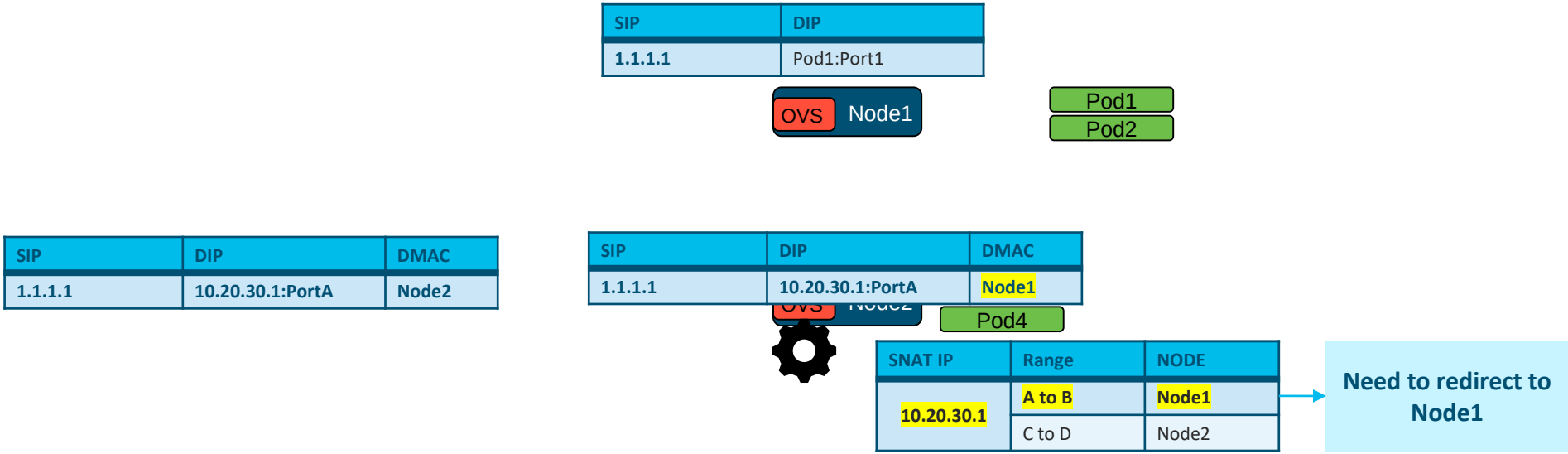
Реализация SNAT
Входящий трафик

1. Сервер 1.1.1.1 отвечает адресу 10.20.30.1, ACI классифицирует трафик в default_extEPG
2. IP получателя 10.20.30.1 соответствует External EPG snat_svcgraph, к трафику применяется контракт между группами
3. Выполняется перенаправление PBR с балансировкой, трафик попадает на один из хостов



Реализация SNAT
Входящий трафик

4. Pod, которому должен быть передан возвратный трафик, может находиться на другом хосте. Если так, что правила OVS «перебрасывают» трафик на нужный хост, переписывая MAC адрес пакета на соответствующий нужному хосту (ориентируясь на номер TCP/UDP порта)
5. Once the traffic reaches the destination Node, OVS connection tracking rules translate the SNAT-IP:SNAT-Port to the pod's Source-IP:Port.



Диагностика проблем

Инструменты диагностики и рекомендации

- Проверка связности на хостах: **ip addr, ip route**
- Просмотр состояния контейнеров: **kubectl get pods --all-namespaces**
- Сбор отчёта (для отправки в TAC или собственного анализа): **acikubectl debug cluster-report -o cluster-report.tar.gz**
- Документация по инсталляции и траблшутингу (список в конце слайдов)
- Трассировка iptables / Open vSwitch (см сессию BRKCLD-3181)
- …
- Обращение в Cisco TAC (интеграция полностью поддерживается Cisco)

В заключение…

Дополнительная информация

- Cisco ACI and Kubernetes Integration
  https://www.cisco.com/c/en/us/td/docs/switches/datacenter/aci/apic/sw/kb/b_Kubernetes_Integration_with_ACI.html
- Cisco ACI and OpenShift Integration
  https://www.cisco.com/c/en/us/td/docs/switches/datacenter/aci/apic/sw/kb/b_Cisco_ACI_and_OpenShift_Integration.html
- Cisco ACI CNI Plugin for Red Hat OpenShift Container Platform Architecture and Design Guide
  https://www.cisco.com/c/en/us/td/docs/switches/datacenter/aci/apic/white_papers/Cisco-ACI-CNI-Plugin-for-OpenShift-Architecture-and-Design-Guide.html
- Cisco ACI Troubleshooting Kubernetes and OpenShift https://www.cisco.com/c/en/us/td/docs/switches/datacenter/aci/apic/sw/4-x/troubleshooting/Cisco-ACI-Troubleshooting-Kubernetes-and-OpenShift.html
- Cisco DevNet: ACI CNI plug-in for Kubernetes learning track https://developer.cisco.com/learning/tracks/acik8s
- Сессии CiscoLive 2020:
  - https://www.ciscolive.com/c/dam/r/ciscolive/emea/docs/2020/pdf/BRKACI-2505.pdf
  - https://www.ciscolive.com/c/dam/r/ciscolive/emea/docs/2020/pdf/BRKACI-3330.pdf
  - https://www.ciscolive.com/c/dam/r/ciscolive/emea/docs/2020/pdf/BRKCLD-3181.pdf


CISCO