Public IaaS를 위한 OpenStack 개선 전략

NHN 클라우드인프라개발랩 | 박성우

seongwoo.park@nhn.com

CONTENTS

1. NHN TOAST

TOAST

- 2015년 Public IaaS Cloud 서비스 시작
- OpenStack 기반으로 상용
  - CentOS 7 + KILO로 시작
    - Network 구성은 DVR, DVR-SNAT을 이용

<TCC: NHN이 자체 기술력으로 설계/건축한 도심형 데이터센터>

NHN

Issues

Linux-OVS Data Plane

- Linux Kernel 을 통과하면 속도 저하가 발생한다.
  - qrouter 성능이 좋지 않다.
  - ovs 성능이 충분하지 않다.
- DVR은 필연적으로 br-int에서 BUM에 의한 loop을 발생시킬 수 있다.
  - Local VLAN이 충돌하는 것을 회피해야 한다.

Neutron-DVR

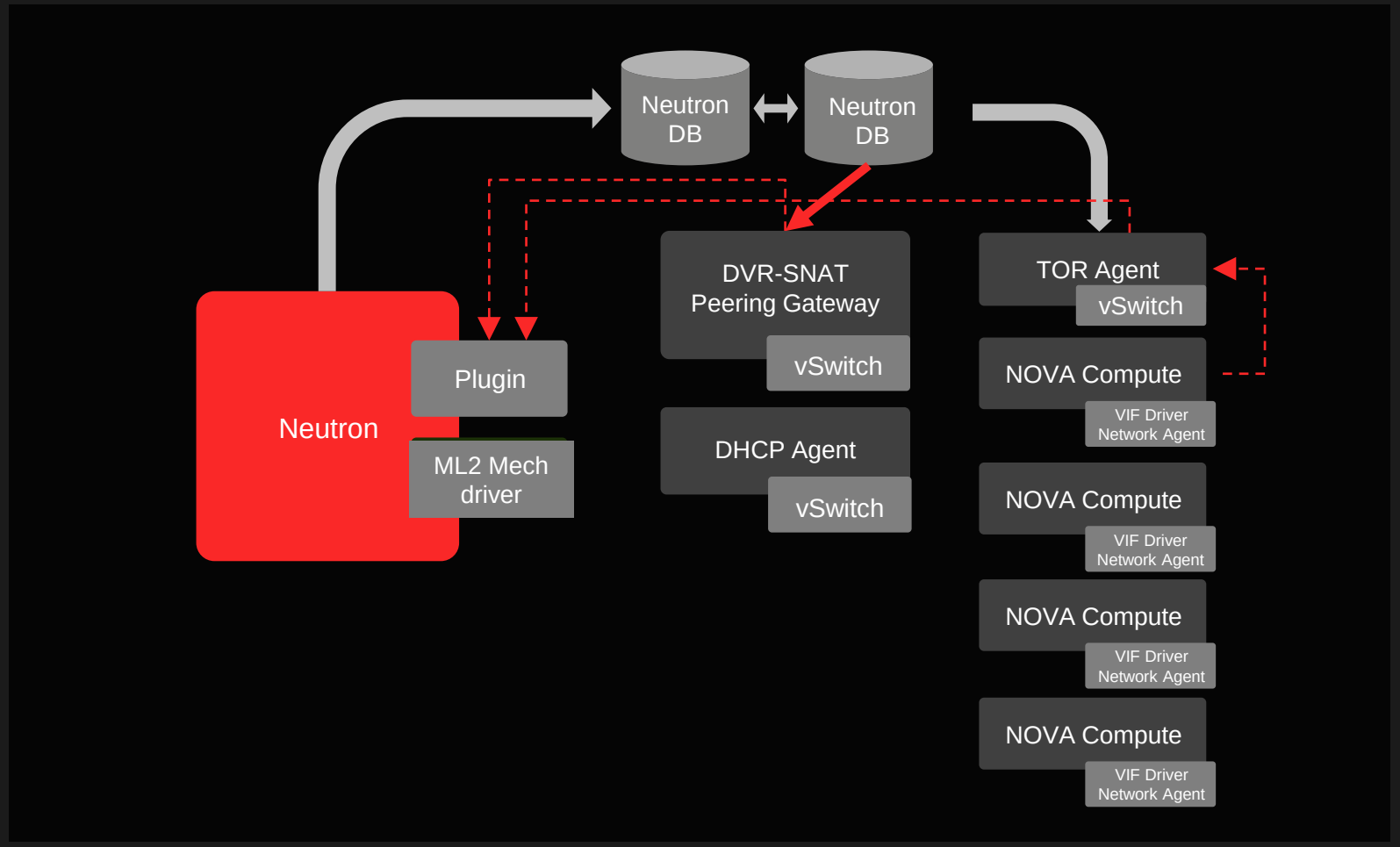
- Namespace는 building 속도를 저하시킨다.
- 전체 Node 수가 증가할 수록 RPC 증가로 Neutron 성능 저하를 일으킨다.
- DVR은 전체적인 RPC 양이 증가한다.
- ovs rule 이 상대적으로 증가한다.

Controller
Provisioning
Data Plane

Implements

구현 사항

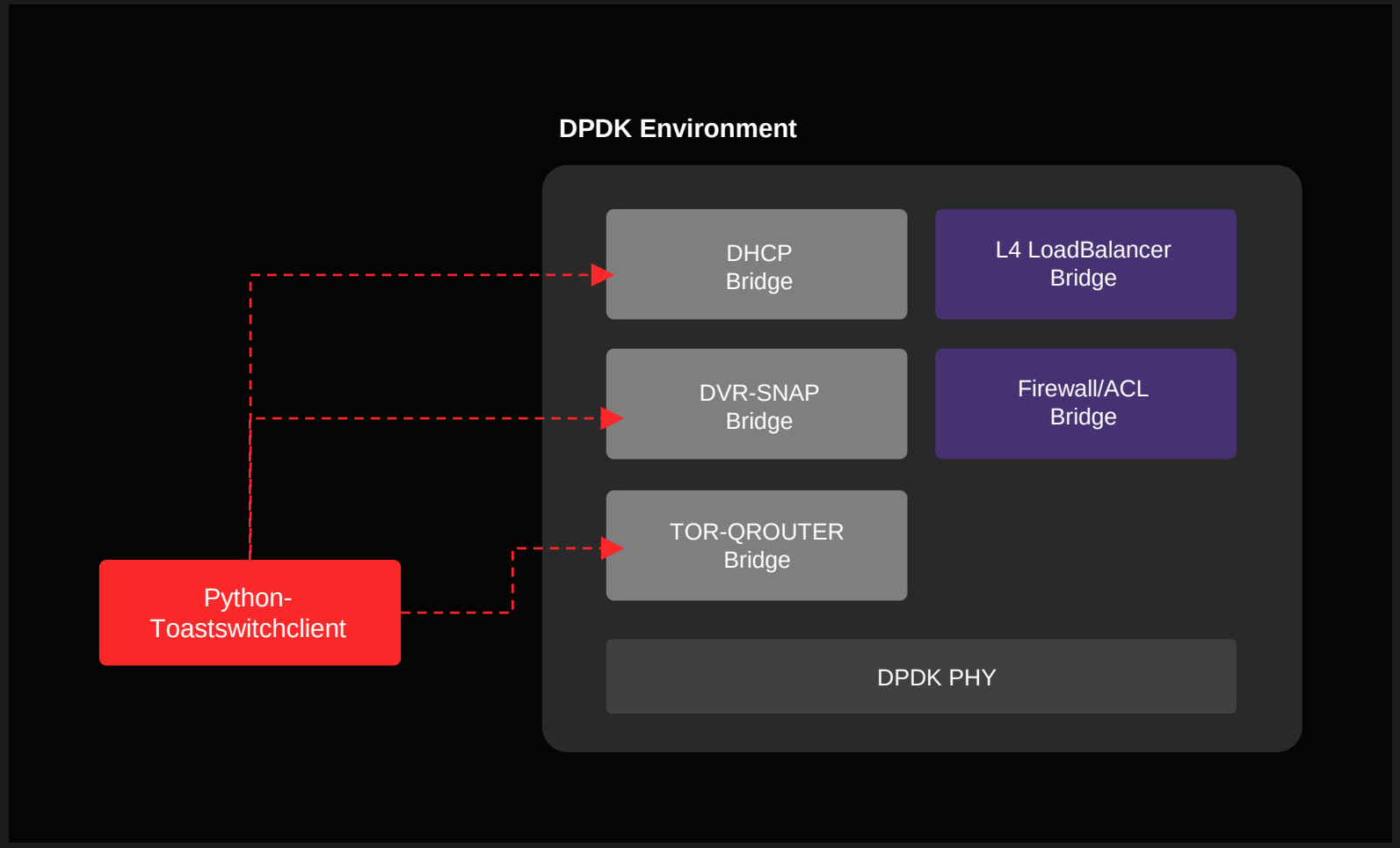
- Neutron Plugin
- Neutron ML2 Mechanism Driver
- NOVA Compute VIF Driver
  - Linux Bridge, SRIOV
  - OVS-ASAP^2
- Agent
  - Compute
  - TOR
  - SNAT
  - DHCP
  - Peering Gateway
  - ML2, L3
- CLI
- Virtual Switch



Implements

TOAST vSwitch

- Neutron 전용 vSwitch
- Functions
  - QRouter
  - DVR-SNAT
  - DHCP
  - Peering Gateway
  - VxLAN, VLAN
- Python Client를 이용하여 TCP제어
- JSON

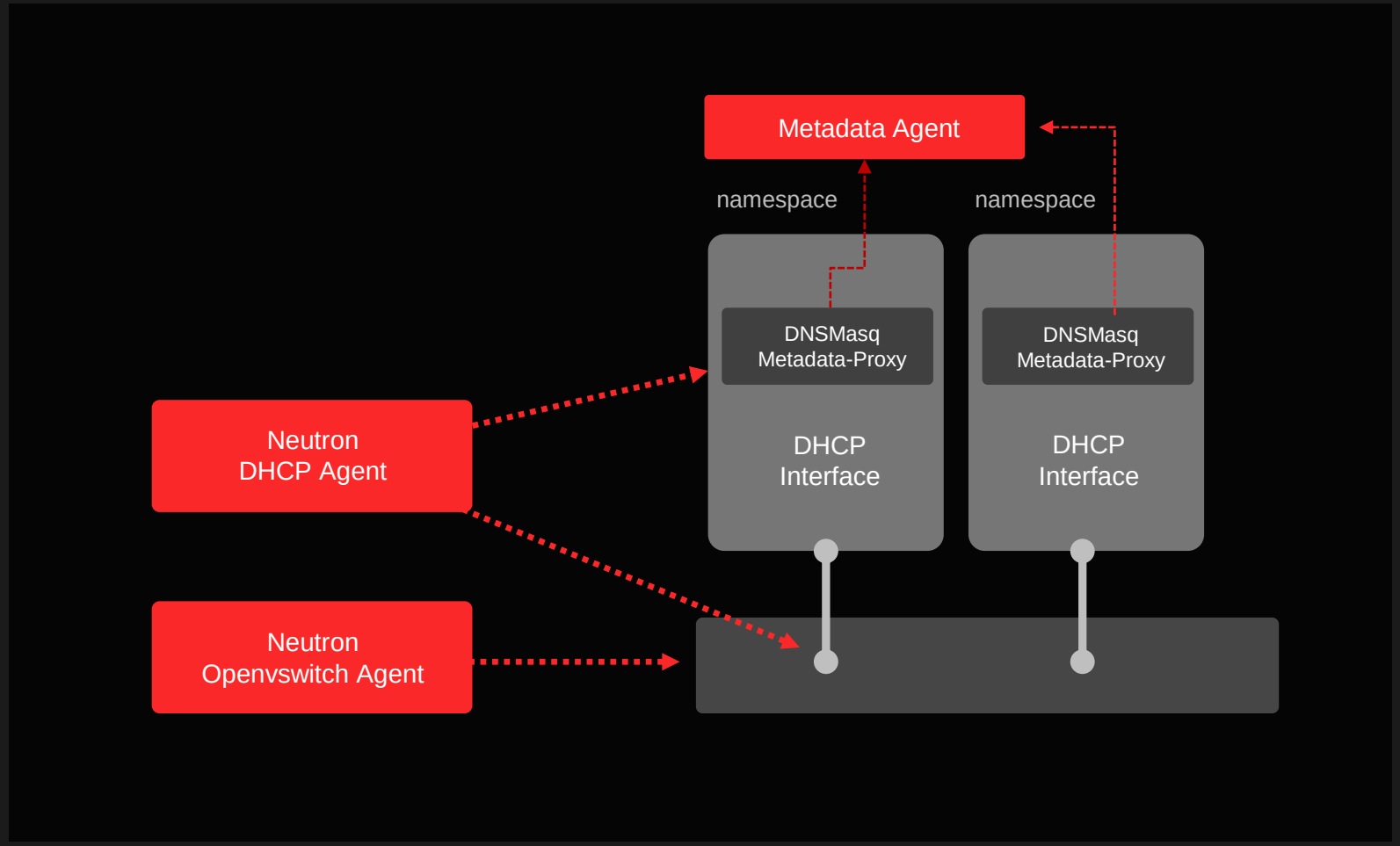


2. DHCP

DHCP

Neutron DHCP Agent

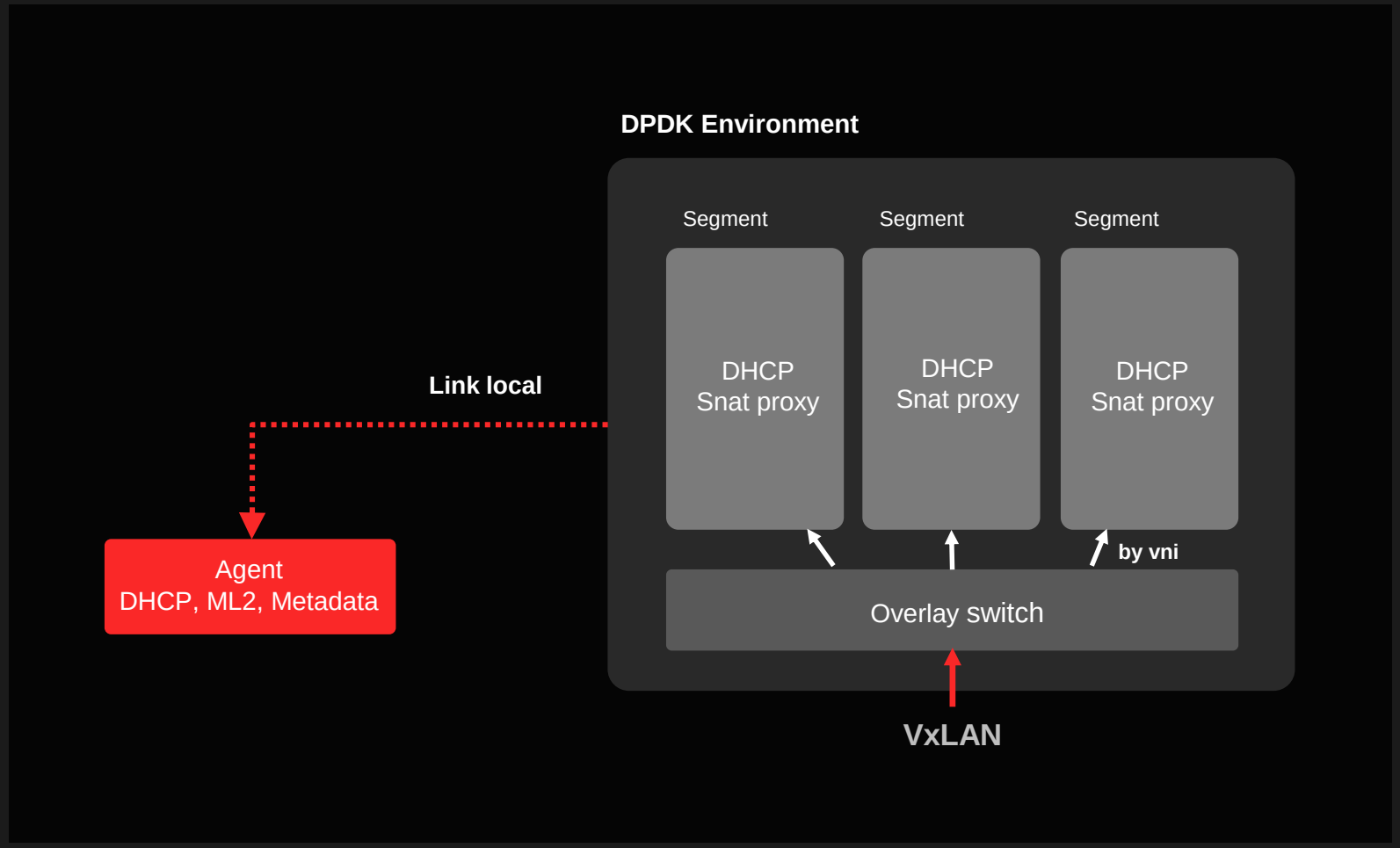
- namespace 생성
- ovs port 생성
  - flow 추가
- dnsmasq in namesapce
- metadata-proxy in namespace



DHCP

DHCP-vSwitch

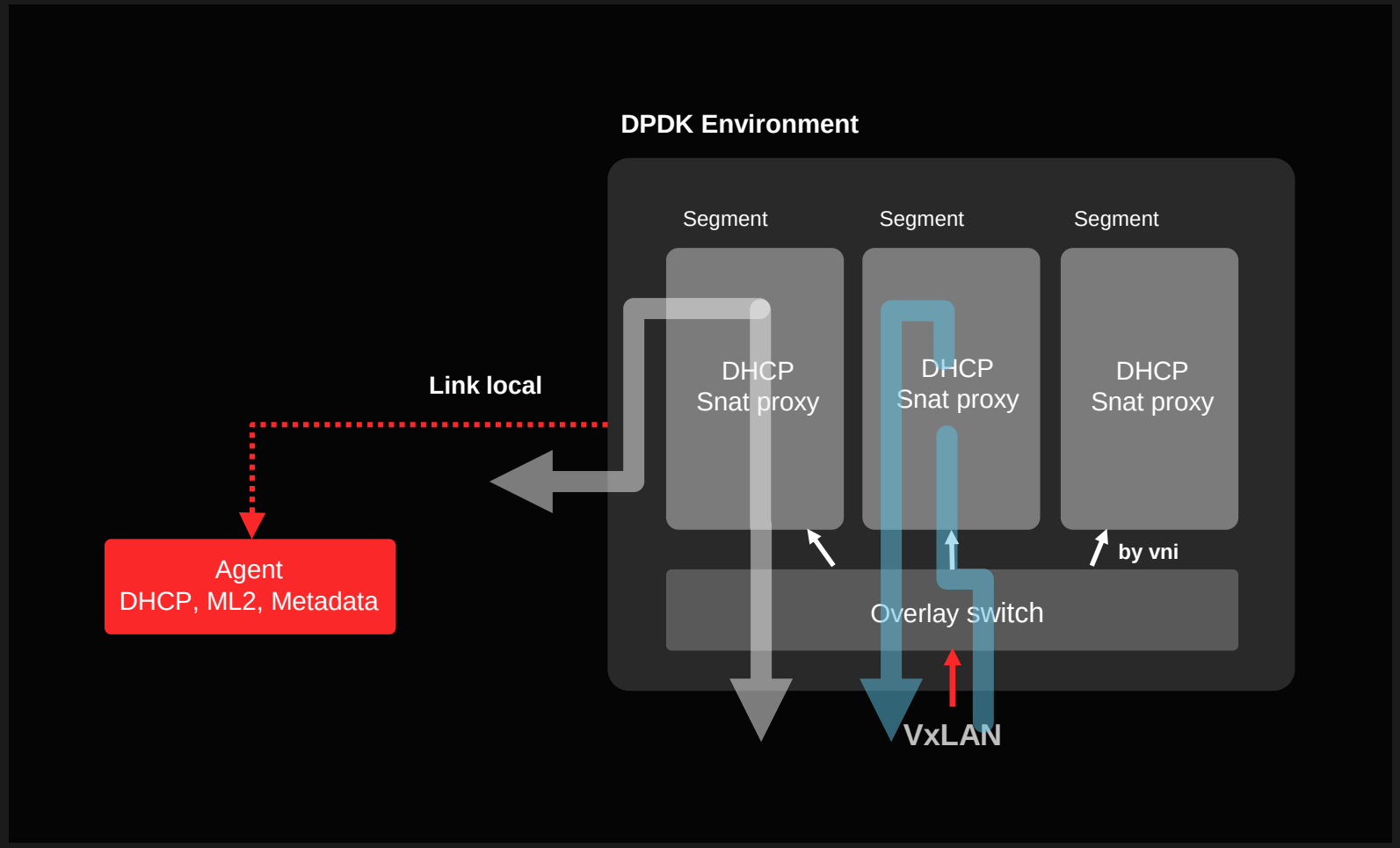
- namespace 생성
- ovs port 생성
  - flow 추가
- dnsmasq in namesapce
- metadata-proxy in namespace



DHCP

DHCP Flows

- DHCP Request는 직접 응답
- Metadata Connection
  - Host 마다 Link Local 주소 할당
  - SNAT 수행 후 Agent 내 proxy로 전달



DHCP

Synchronization

- CPU 2.4Ghz, 4 vCPU, Memory 6GB
- Neutron Network 1000개
  - 300개 Network에만 10개씩 HOST = 3000개 HOST
- Ubuntu 16.04

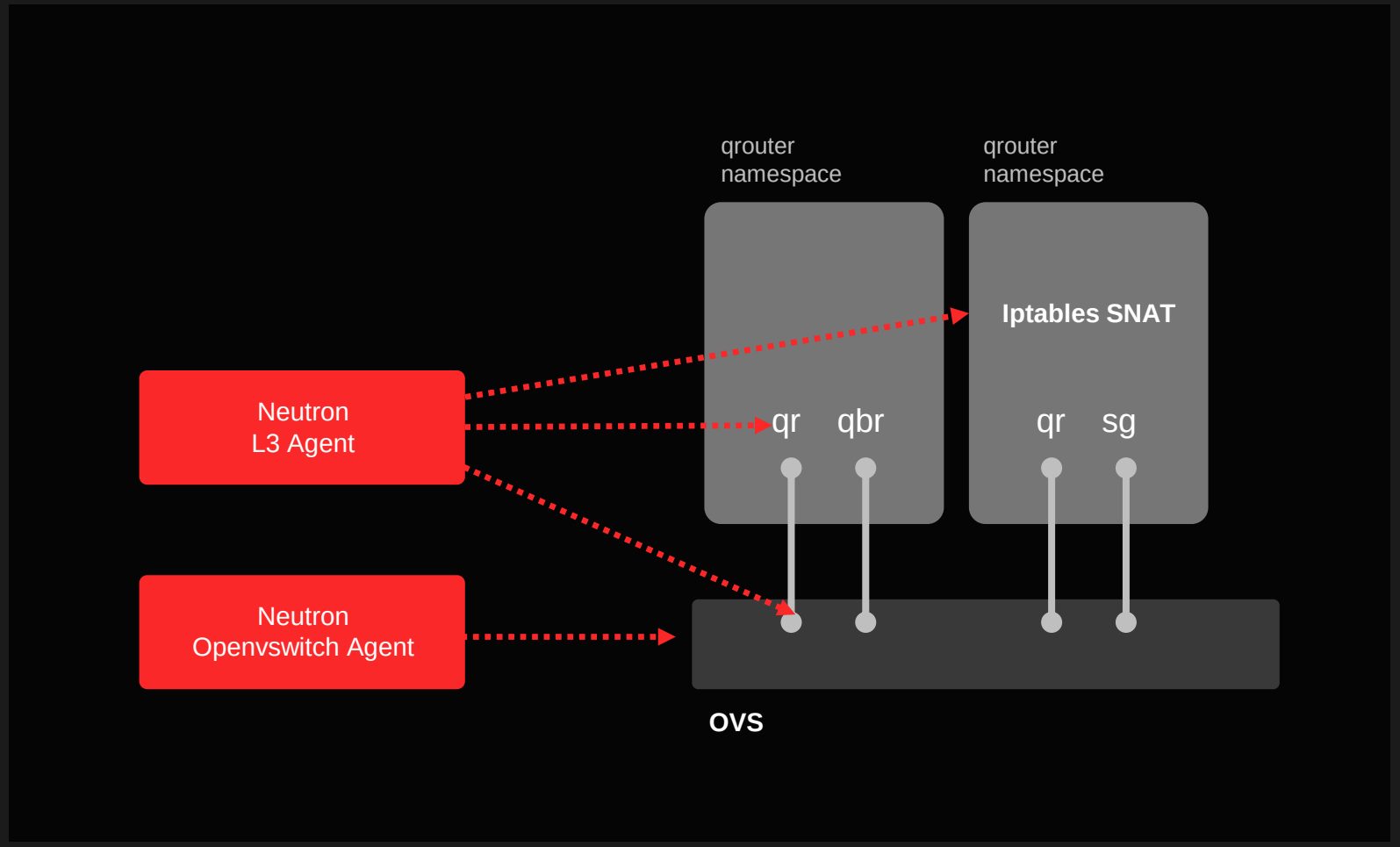
	설명	Neutron DHCP Agent	DPDK Based
T1	Cache -> vSwitch Sync.	-	21sec
T2	Neutron Network Fetching	29min 29sec	55sec
T3	Cache -> vSwitch Sync.		

3. DVR-SNAT

DVR-SNAT

Neutron L3 Agent

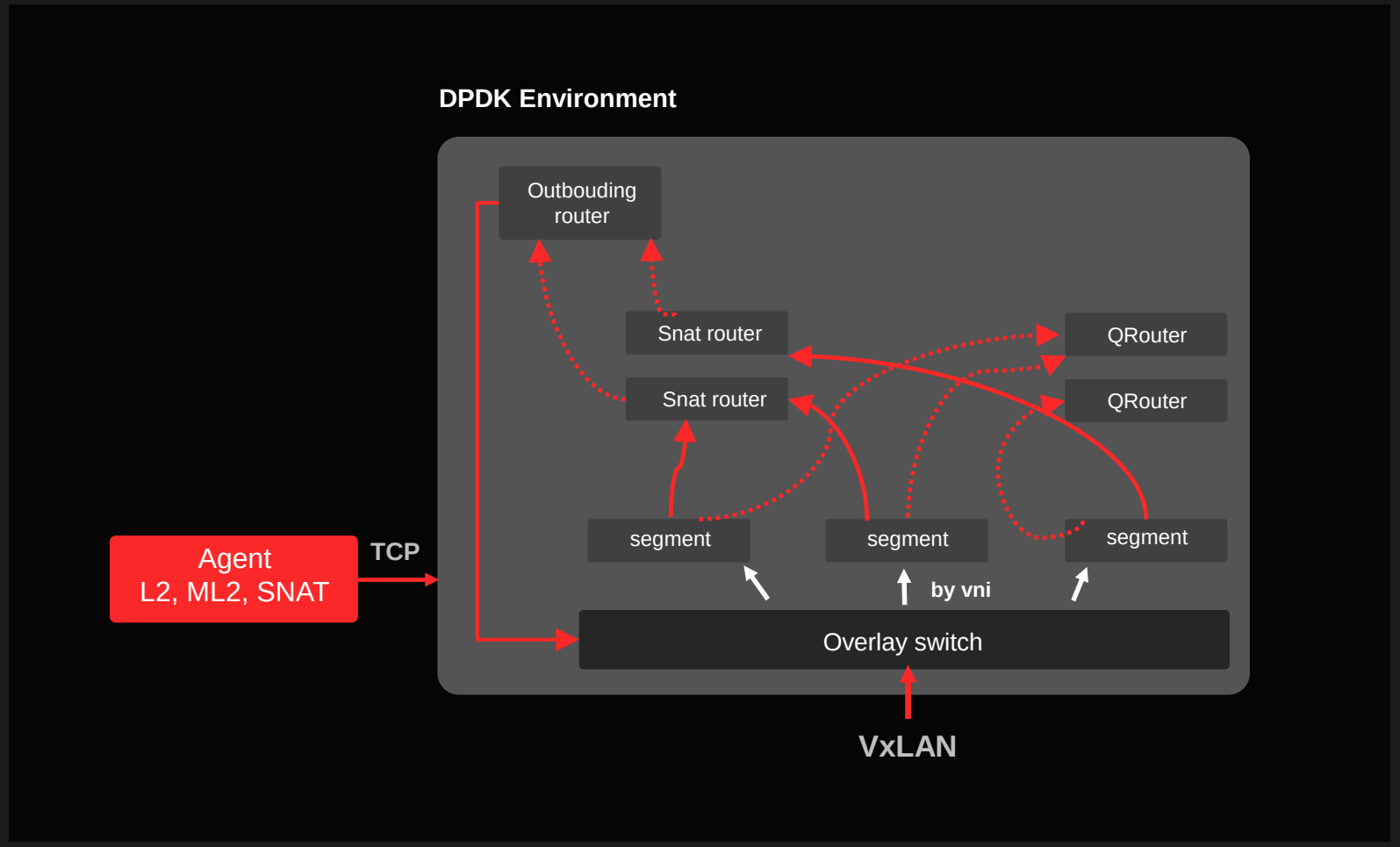
- snat namespace 생성
- qrouter namespace 생성
- ovs port 생성
  - flow 추가
- SNAT iptables in namesapce



DVR-SNAT

SNAT-vSwitch

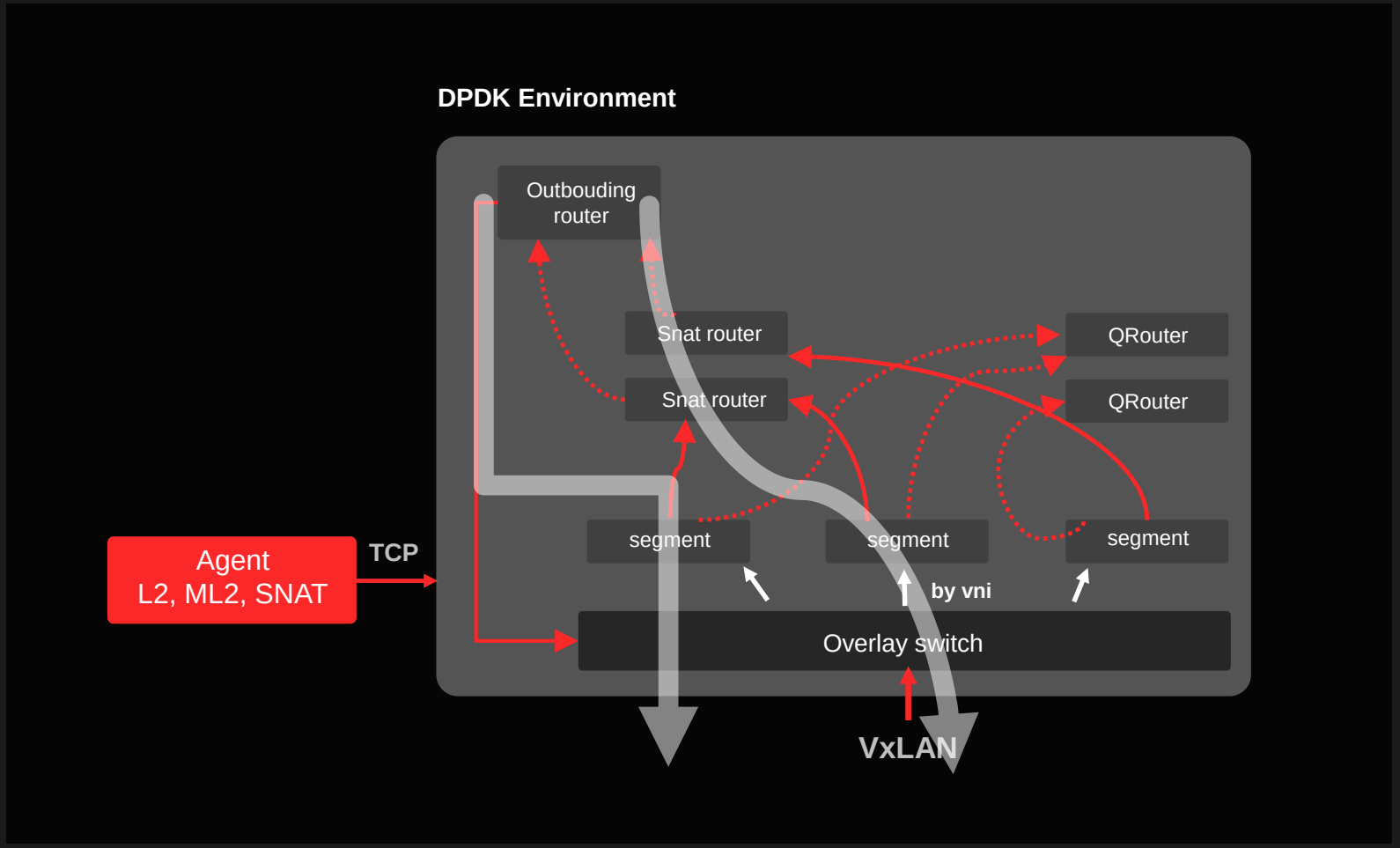
- SNAT Agent에서 host 정보를 segment 단위로 동기화
- SNAT Router, QRouter 생성
- Outbounding router
- Active-Standby
- Maintenance Migration



DVR-SNAT

SNAT Flows

- SNAT
  - Routing
  - SNAT - Connection Tracking
  - Traffic Count



DVR-SNAT

Synchronization

- CPU 2.4Ghz, 4 vCPU, Memory 6GB
- Neutron Network 1000개
  - 300개 Network에만 10개씩 HOST = 3000개 HOST
  - 단 Neutron L3 Agent는 성능 문제로 100개만 수행
- Ubuntu 16.04

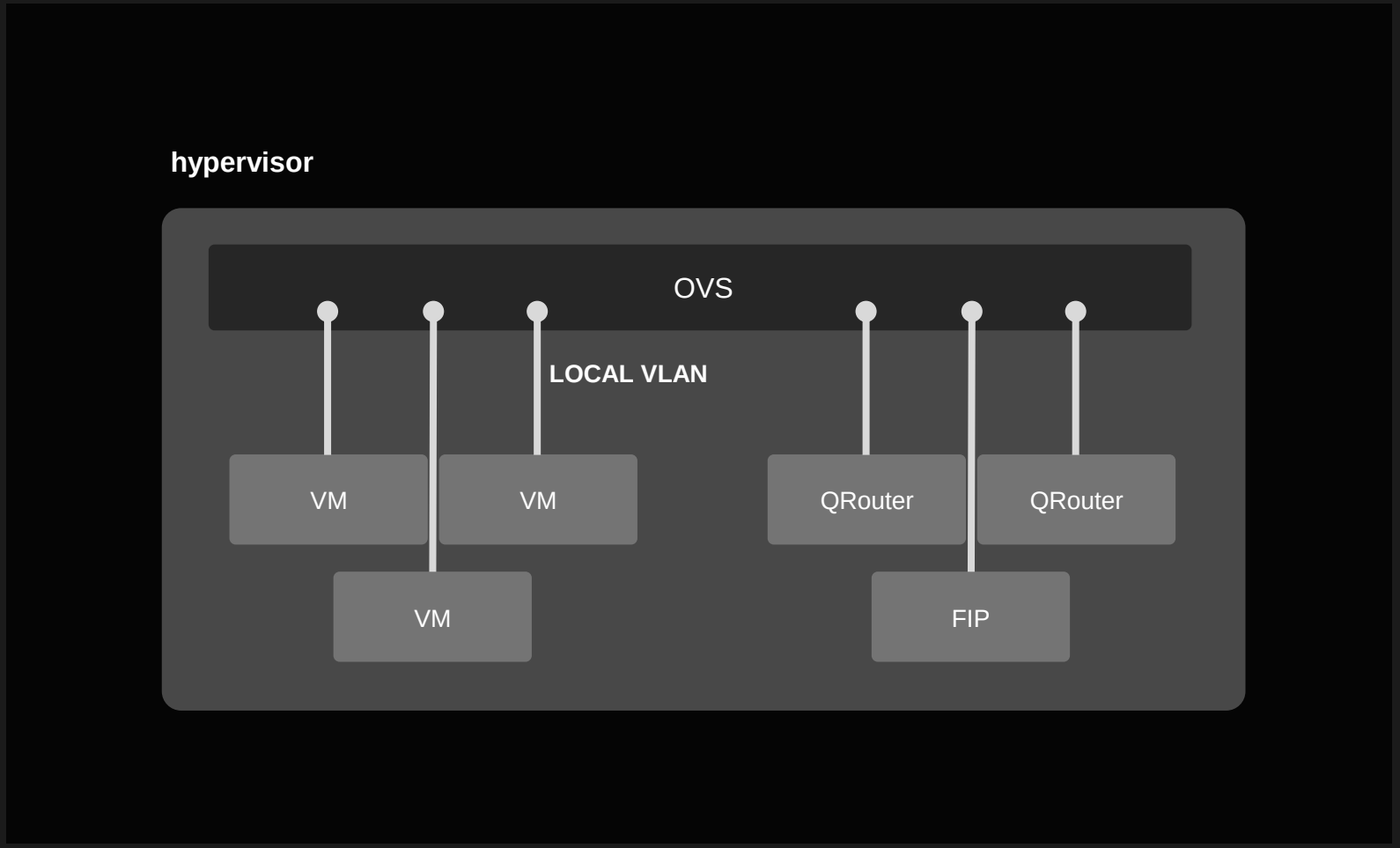
	설명	Neutron L3 Agent	DPDK Based
T1	Cache -> vSwitch Sync.	-	43sec
T2	Neutron Router Fetching	14min 39sec Network = 100 Host = 1000	59sec
T3	Cache -> vSwitch Sync.		13sec

4. TOR Controllers

TOR Controller

Nova Compute

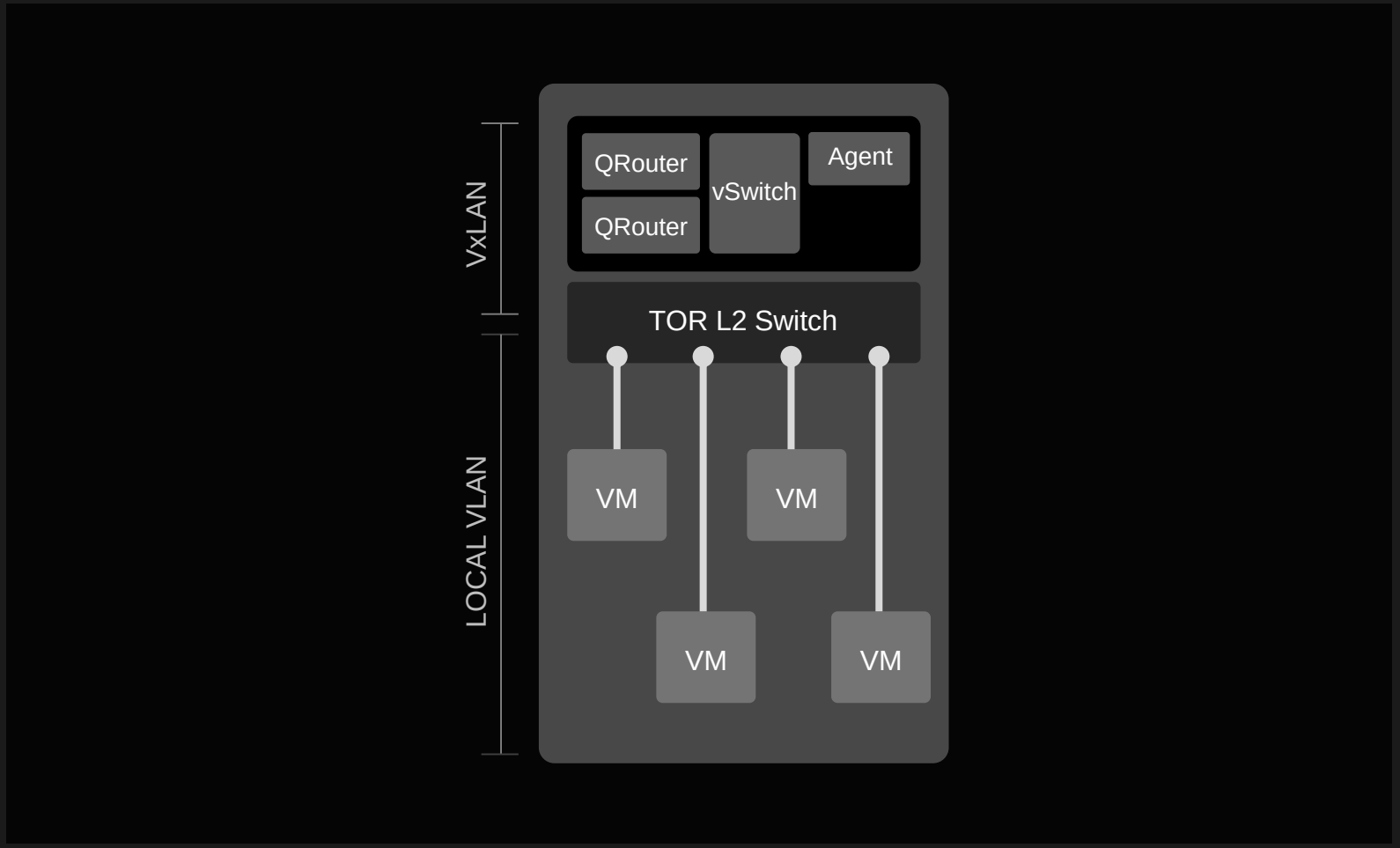
- ovs port 생성
  - flow 추가
- qrouter in namesapce
- fip network in namespace
- Local VLAN 으로 네트워크 구분



TOR Controller

TOR Agent

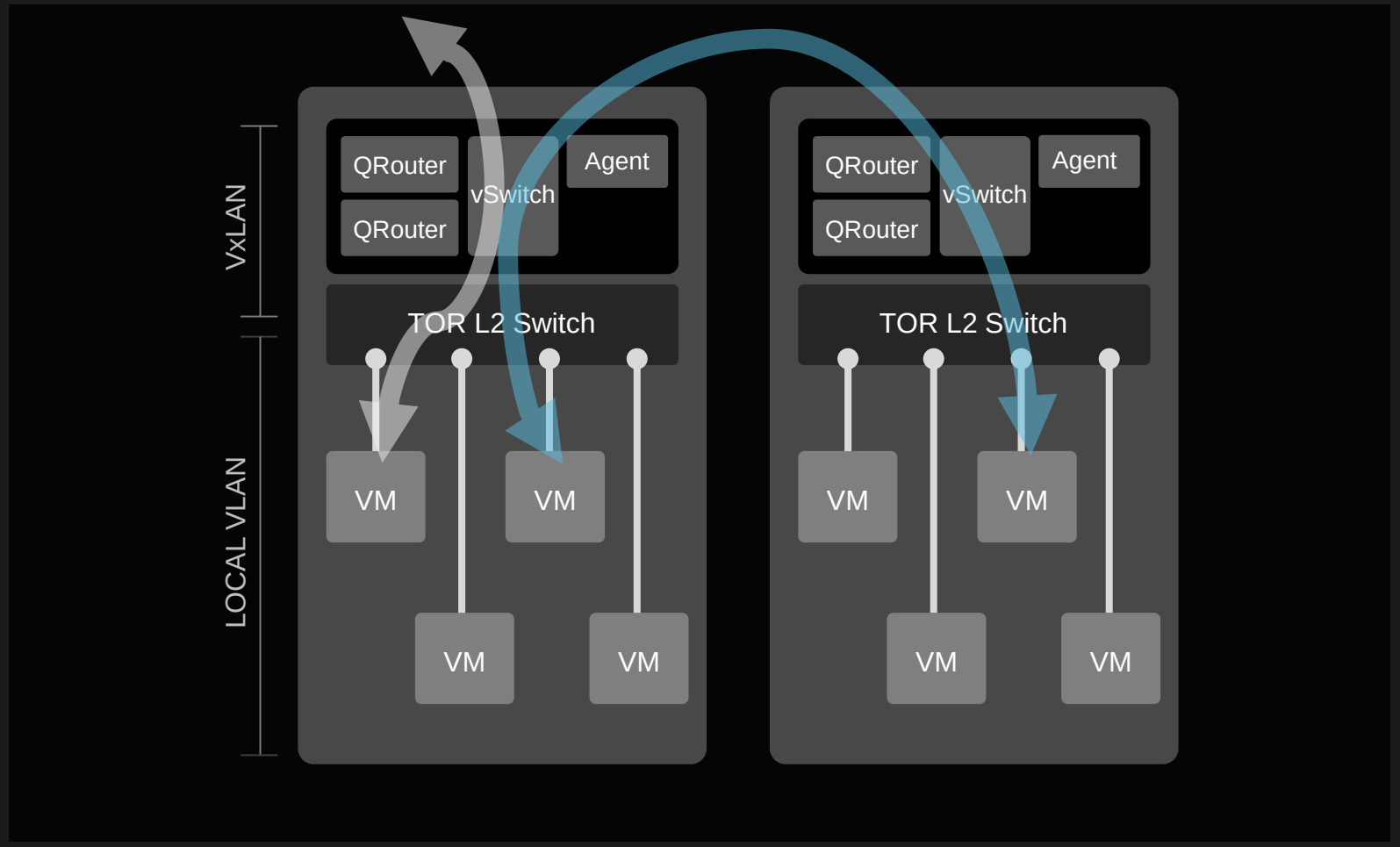
- 각 Compute Node는 TOR Agent에 API 요청
  - read에 대해서는 바로 응답.
  - write에 대해서는 Neutron으로 전달.
  - distributed controller 역할
- L3 function
  - qrouter 구성
  - Floating IP 처리
- L2 function
  - segment 구성 및 VxLAN Tunnel 구성



TOR Controller

Flows

- EW Traffic
  - vSwitch 에서 VxLAN encap
- NS Traffic
  - vSwitch QRouter를 통해 전달
  - Floating IP는 QRouter에서 DNAT
  - FIP, VTEP에 대해서 BGP



TOR Controller

Performance

- CPU 2.4Ghz, Memory 6GB @Ubuntu 18.04 KVM
- NIC Mellanox ConnectX-5 25G x2

Direction	Linux-OVS QRouter	DPDK Based QRouter	
Subnet to Subnet	200K ~ 400Kpps	30Mpps @ 6vCPU 5Mpps/Core	Mellanox PF pci-passthrouch
Floating IP	-	20Mpps @ 4vCPU 5Mpps/Core	Mellanox VF pci-passthrouch on ASAP^2

5. Plans

Plans

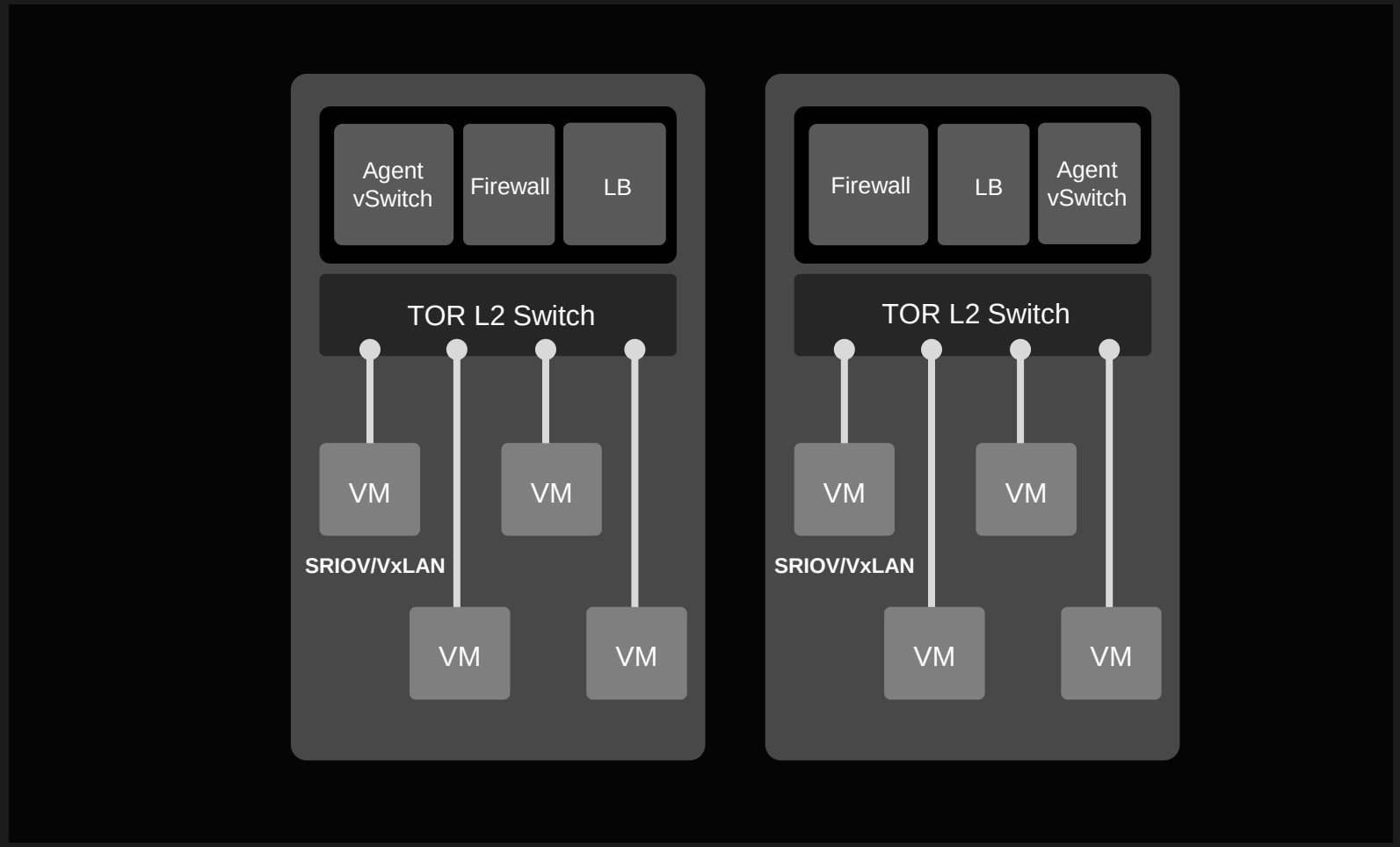
Implementing Neutron Components

- Distributed Firewall
- Distributed Layer 4 Load Balancer using DPDK
  - Amphora Cluster Manager - octavia
  - Layer 2 Direct Server Return
- Full L3 Architecture
  - Compute - VxLAN

Plans

Implementing Neutron Components

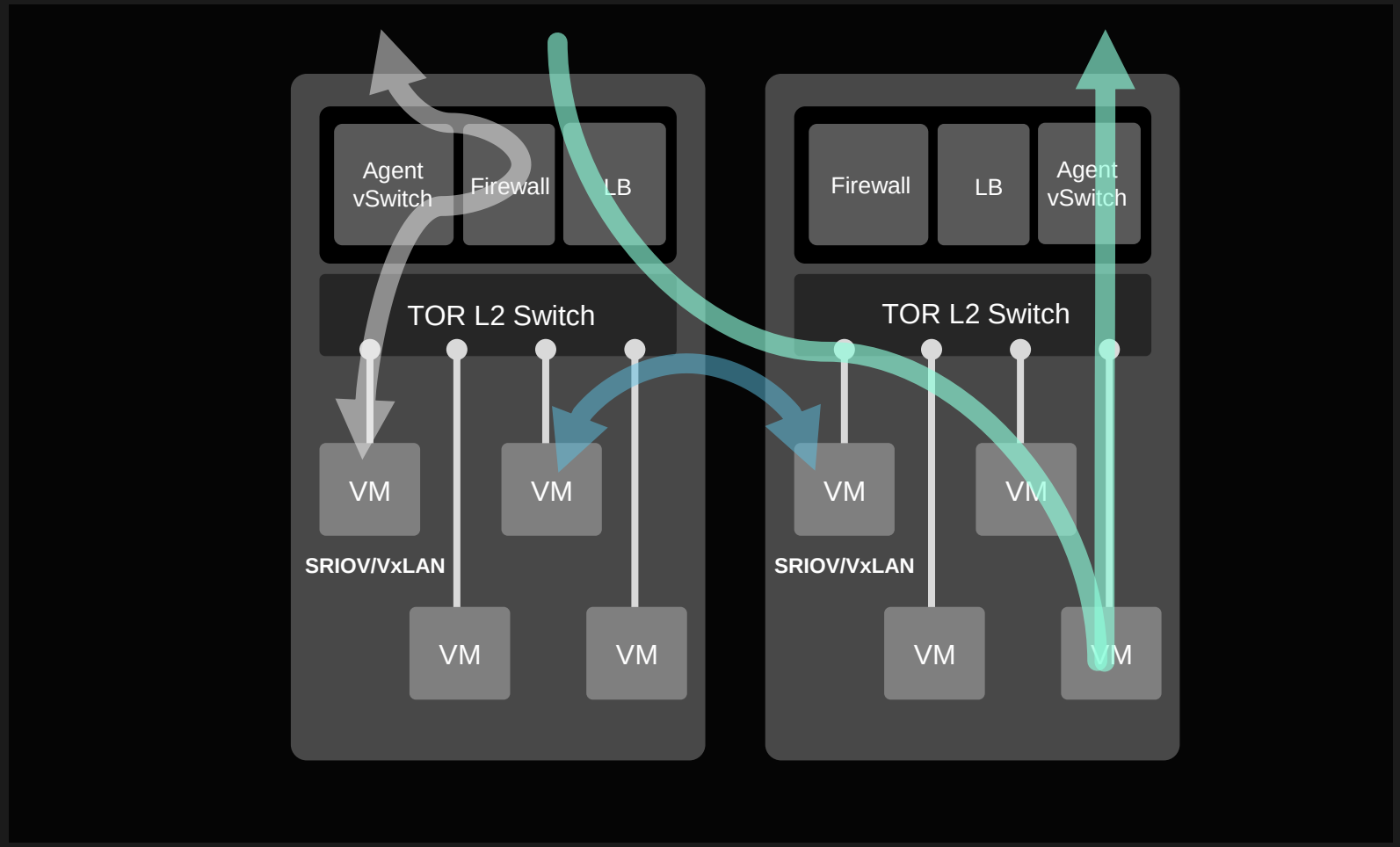
- Distributed Firewall on TOP of RACK
- Distributed L4 Load Balancer on TOP of RACK
  - Amphora Cluster Manager - Octavia



Plans

Optimizing Flows

- Layer 2 Direct Server Return
- Full L3 Architecture
  - EW traffic - HW assist
  - NS traffic - DPDK assist



Q&A

THANK YOU