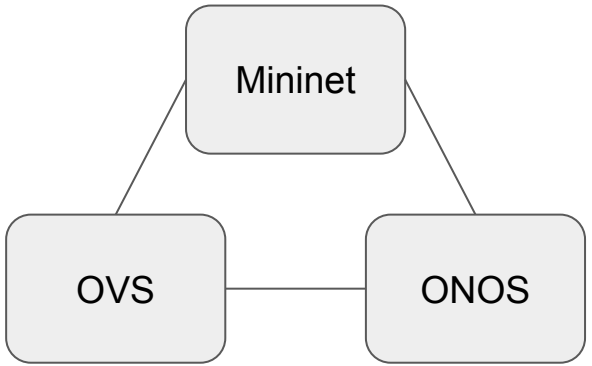
PN Lab 1

Due: 11/3

Outline

- Introduction to Lab 1
- Introduction to Mininet & OVS
- Introduction to ONOS



Outline

- Introduction to Lab 1
- Introduction to Mininet & OVS
- Introduction to ONOS

Lab1

- see SPEC

Outline

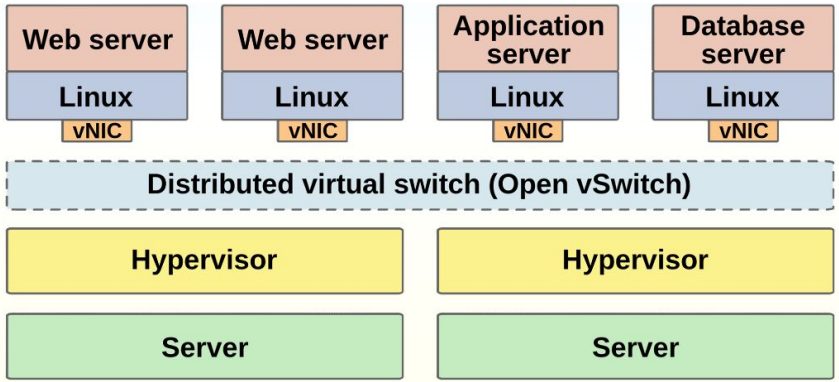
- Introduction to Lab 1
- Introduction to Mininet & OVS
- Introduction to ONOS

Mininet

- Network emulator
  - Mininet's virtual hosts, switches, links, and controllers are the real thing – they are just created using software rather than hardware – and for the most part their behavior is similar to discrete hardware elements
- Mininet CLI
  - a variety of useful commands, ex: **sudo mn --topo tree,2**
  - display xterm windows and to run commands on individual nodes
- Mininet API
  - customize your topology, switch, host, controller, or link classes
    - extend the mn command line tool using the --custom option
      - **sudo mn --custom mytopo.py --topo mytopo,3**
    - writing complete Mininet scripts in Python
- see Reference

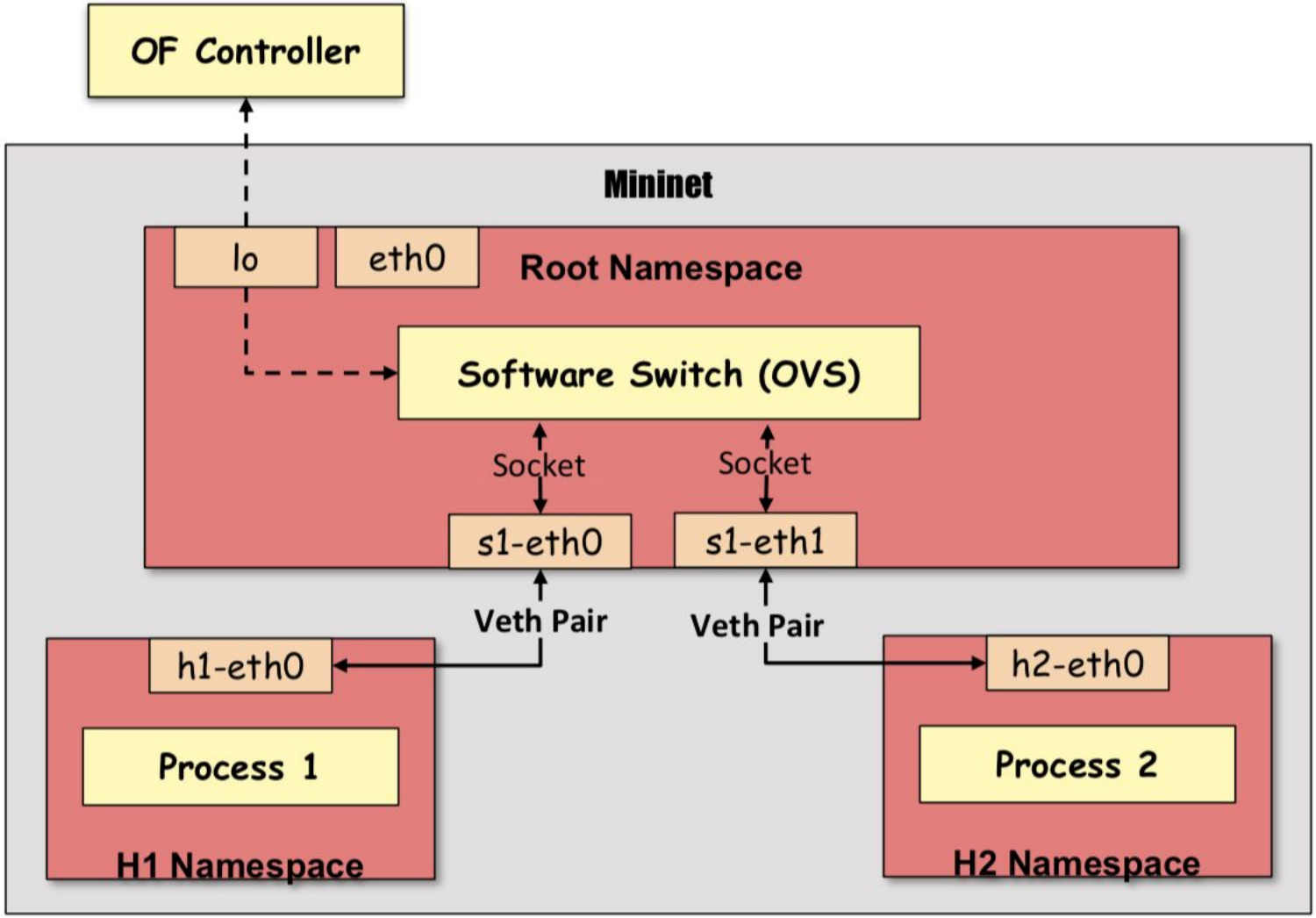
Open vSwitch

- Open source software-implemented virtual switch
  - multilayer (L2/L3), OpenFlow capable
- Typically used with hypervisors to interconnect
  - VMs within a host
  - VMs machines between different hosts across networks



Network isolation in Mininet

- Linux Network Namespace
  - logically partitions kernel network resources
  - assign virtual Ethernet (veth) interfaces to a network namespace
- see Reference:
  - Introducing Linux Network Namespaces
    - A Follow Up on Linux Network Namespaces
  - 基于Linux Network Namespace的Mininet架构分析
  - Mininet 運作原理
  - Mininet and Open vSwitch


OF Controller
Mininet
lo
eth0
Root Namespace
Software Switch (OVS)
Socket
Socket
s1-eth0
s1-eth1
Veth Pair
Veth Pair
h1-eth0
h2-eth0
Process 1
Process 2
H1 Namespace
H2 Namespace

```
sudo bash
# Create host namespaces
ip netns add h1
ip netns add h2
# Create switch
ovs-vsctl add-br s1
# Create links
ip link add h1-eth0 type veth peer name s1-eth1
ip link add h2-eth0 type veth peer name s1-eth2
ip link show
# Move host ports into namespaces
ip link set h1-eth0 netns h1
ip link set h2-eth0 netns h2
ip netns exec h1 ip link show
ip netns exec h2 ip link show
# Connect switch ports to OVS
ovs-vsctl add-port s1 s1-eth1
ovs-vsctl add-port s1 s1-eth2
ovs-vsctl show
# Set up OpenFlow controller
ovs-vsctl set-controller s1 tcp:127.0.0.1
controller ptcp: &
ovs-vsctl show
```

```
# Configure network
ip netns exec h1 ifconfig h1-eth0 10.1
ip netns exec h1 ifconfig lo up
ip netns exec h2 ifconfig h2-eth0 10.2
ip netns exec h1 ifconfig lo up
ifconfig s1-eth1 up
ifconfig s1-eth2 up
# Test network
ip netns exec h1 ping -c1 10.2
```

These were automatically done in mininet CLI / API

Outline

- Introduction to Lab 1
- Introduction to Mininet & OVS
- Introduction to ONOS
  - ONOS & ONOS Architecture
  - ONOS Basic operation

Outline

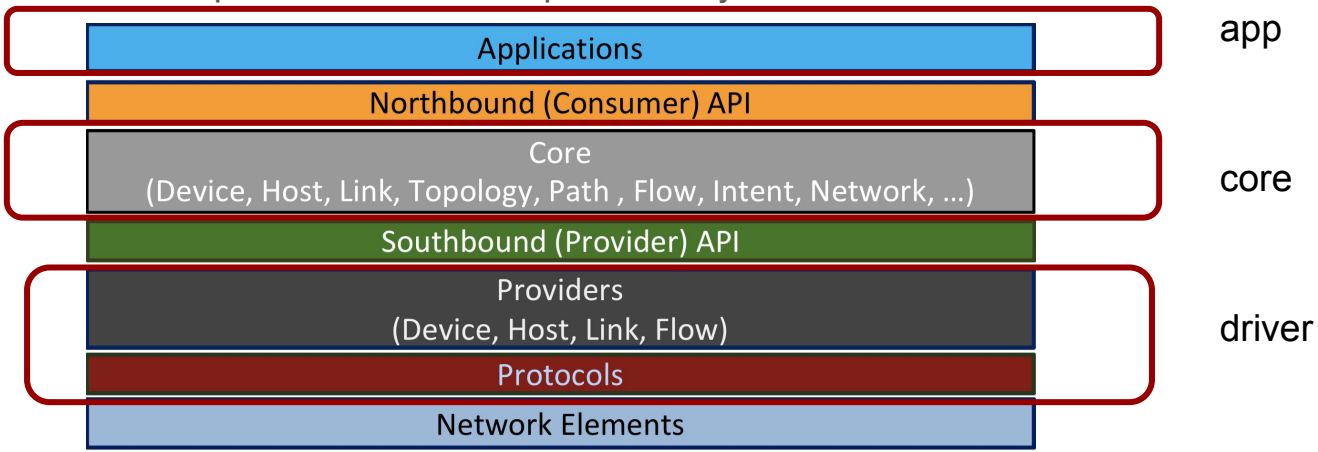
- Introduction to Lab 1
- Introduction to Mininet & OVS
- Introduction to ONOS
  - ONOS & ONOS Architecture
  - ONOS Basic operation

ONOS

- Open-source project hosted by Open Networking Foundation (ONF)
- Design Goals
  - High-Availability, Scalability and Performance
  - Abstraction
  - Modularity

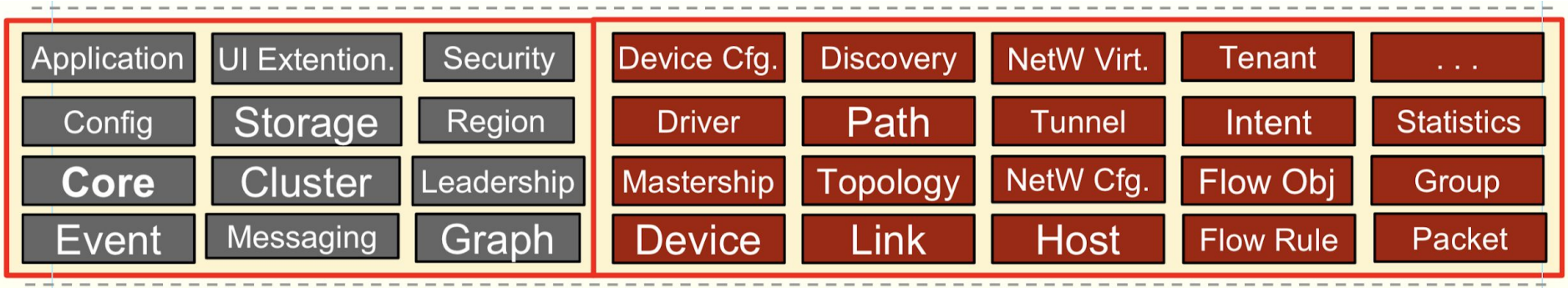
ONOS Layered Architecture

- Protocol-aware Network-facing Modules (driver)
  - acquire network state information through protocol-specific means
- Protocol-agnostic System Core
  - tracks and serves information about network state
- Applications
  - consume and act upon the information provided by the core



ONOS Core Services (Subsystems)

- Device / Link / Host / Topology Subsystem
- PathService
- **FlowRule Subsystem**
  - Manages inventory of match/action flow rules installed on infrastructure devices
  - Provides flow metrics.
- **Packet Subsystem**
  - Packet-in, Packet-out



Important services in Lab 1

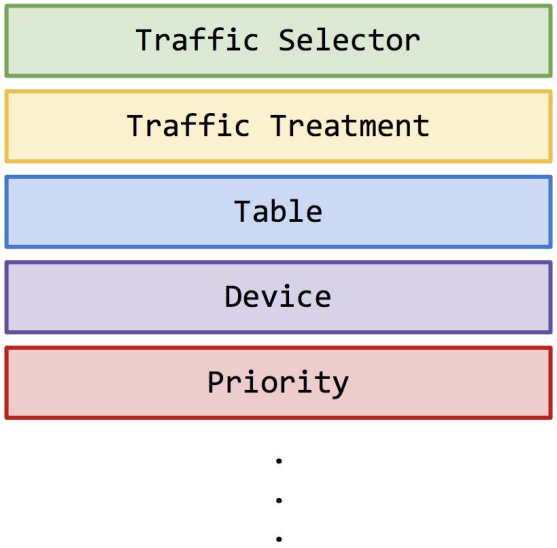
- CoreService
- FlowRuleService
- PacketService

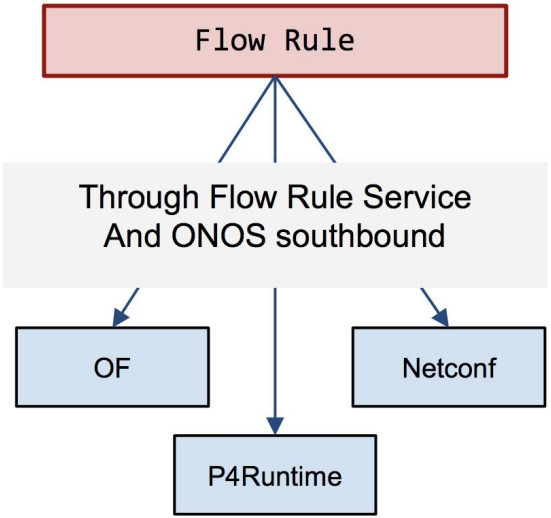
3 ways to insert flow rules (concrete to abstract)

- Flow Rule Service
  - protocol Independent, pipeline Specific
- Flow Objective Service
  - table pipeline independent
- Intent Service
  - topology independent

FlowRule Service

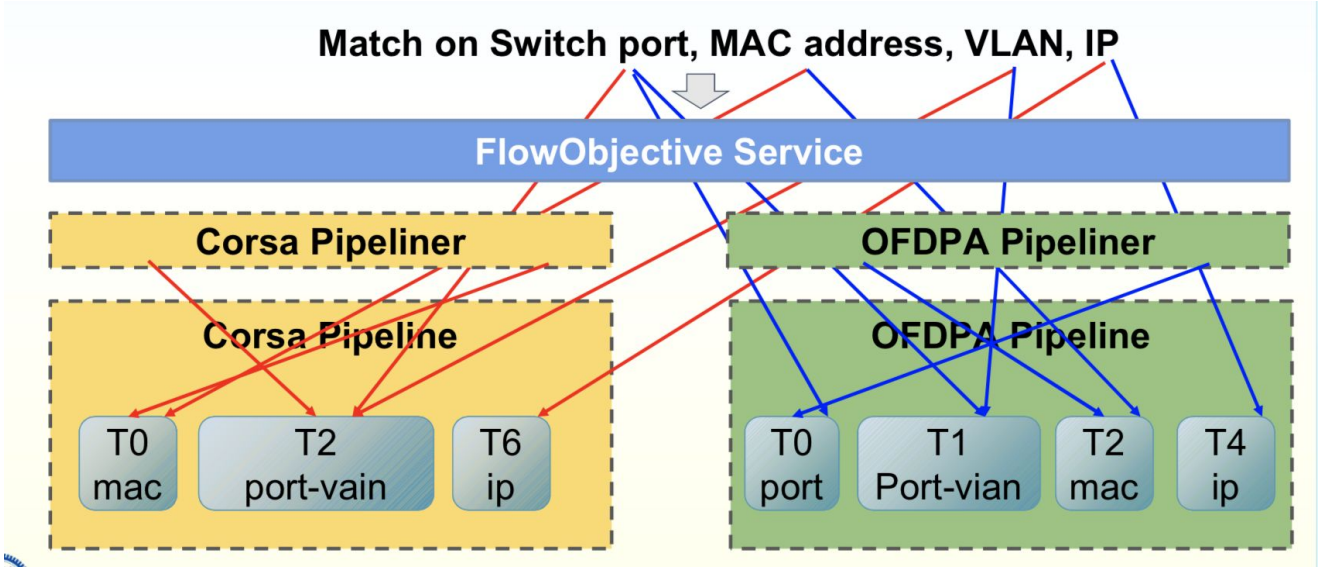
- Protocol Independent, Pipeline Specific
  - Applications needs to specify Table and Flow Rule information





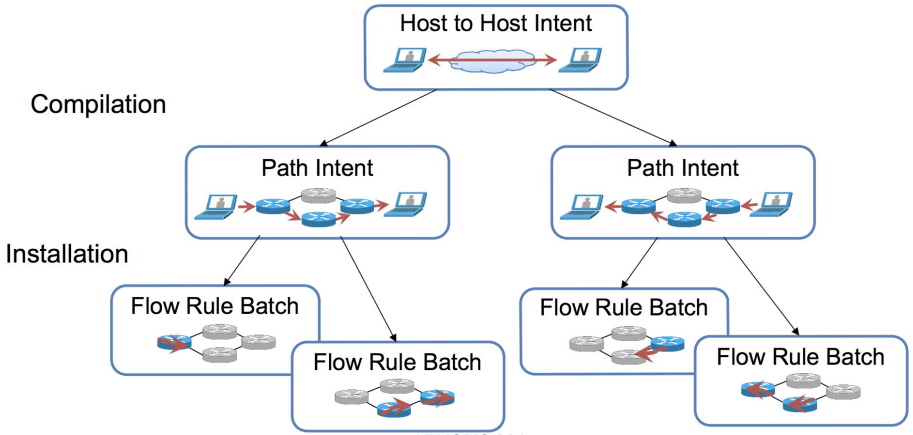
FlowObjective Service

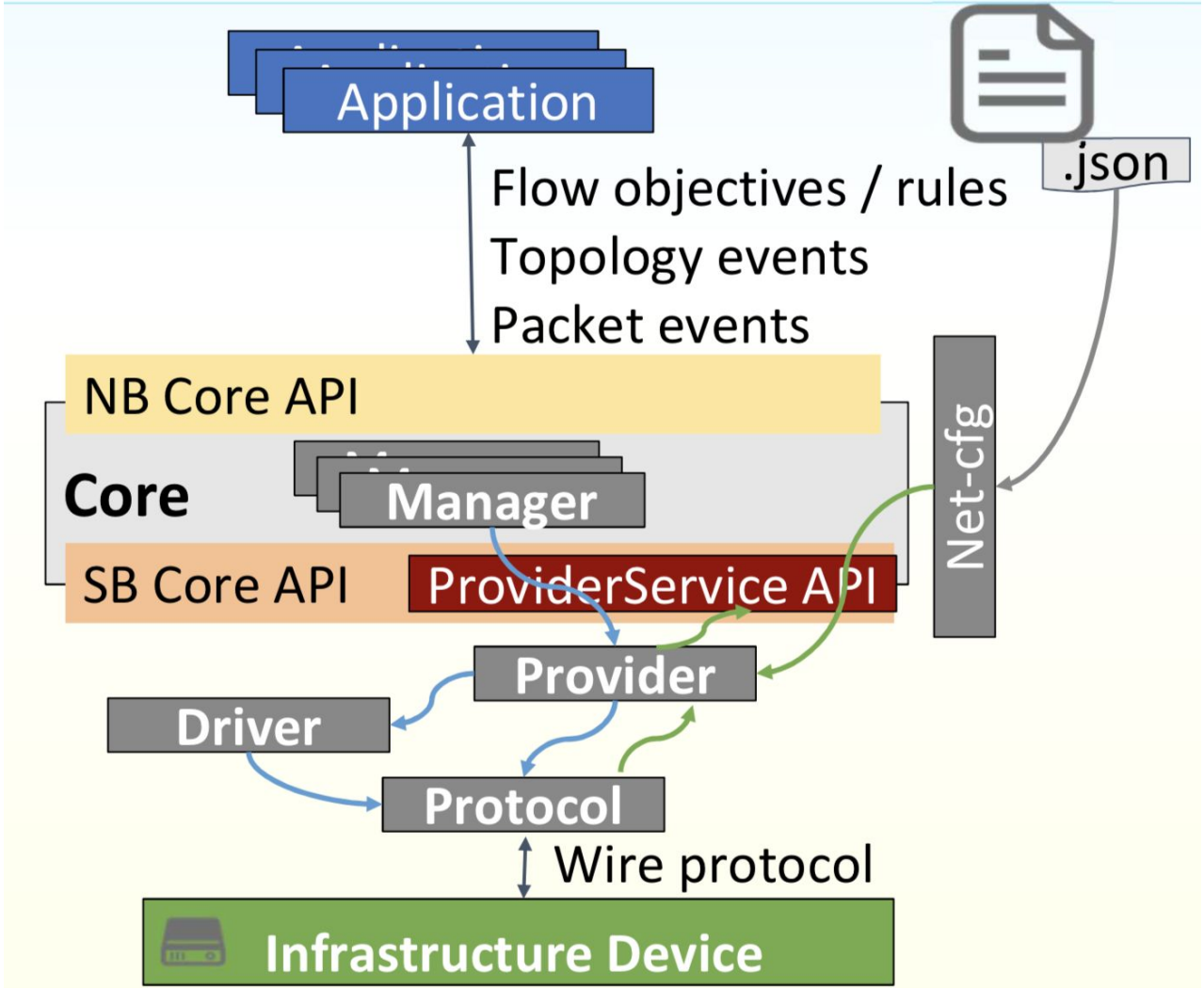
- table pipeline independent
- Invoke a Pipeliner to handle details of pipeline
- developers write applications once for all pipelines



Intent Service

- topology independent
- what should be done rather than how it is specifically programmed
- Intent compiler
  - Compiler produces more specific Intents given the environment
- Intent installer
  - Installer transforms Intents into device commands




Application
.json
Flow objectives / rules
Topology events
Packet events
NB Core API
Core
Manager
SB Core API
ProviderService API
Net-cfg
Provider
Driver
Protocol
Wire protocol
Infrastructure Device

Outline

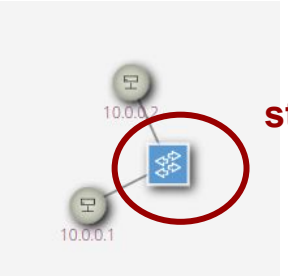
- Introduction to Lab 1
- Introduction to Mininet & OVS
- Introduction to ONOS
  - ONOS & ONOS Architecture
  - ONOS Basic operation
    - ONOS CLI
    - ONOS Web GUI

ONOS CLI

- see Lab 1 Guideline

ONOS Web GUI

- start ONOS, then visit http://localhost:8181/onos/ui
  - User: onos
  - Password: rocks
- Observe Topology
  - try: press ‘h’ to see host after ping
- Observe inserted Flow Rules
  - press the following icon



step 1

step 2