Off-path SmartNIC Port Binding with OVN

OVS+OVN '21

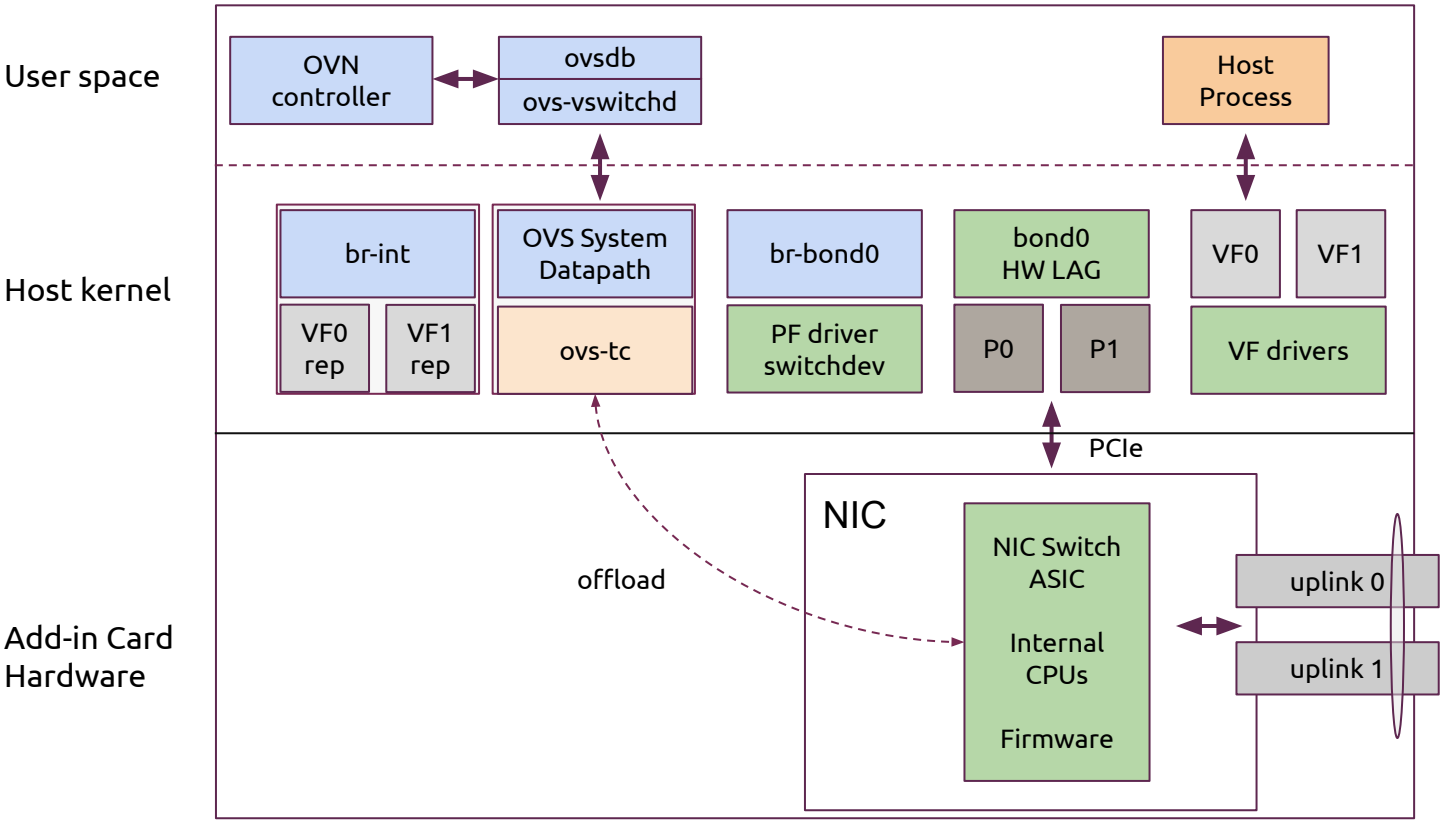
Dmitrii Shcherbakov, Senior Engineer, OVN Engineering
Frode Nordahl, Senior Engineer, OVN Engineering

CANONICAL ubuntu

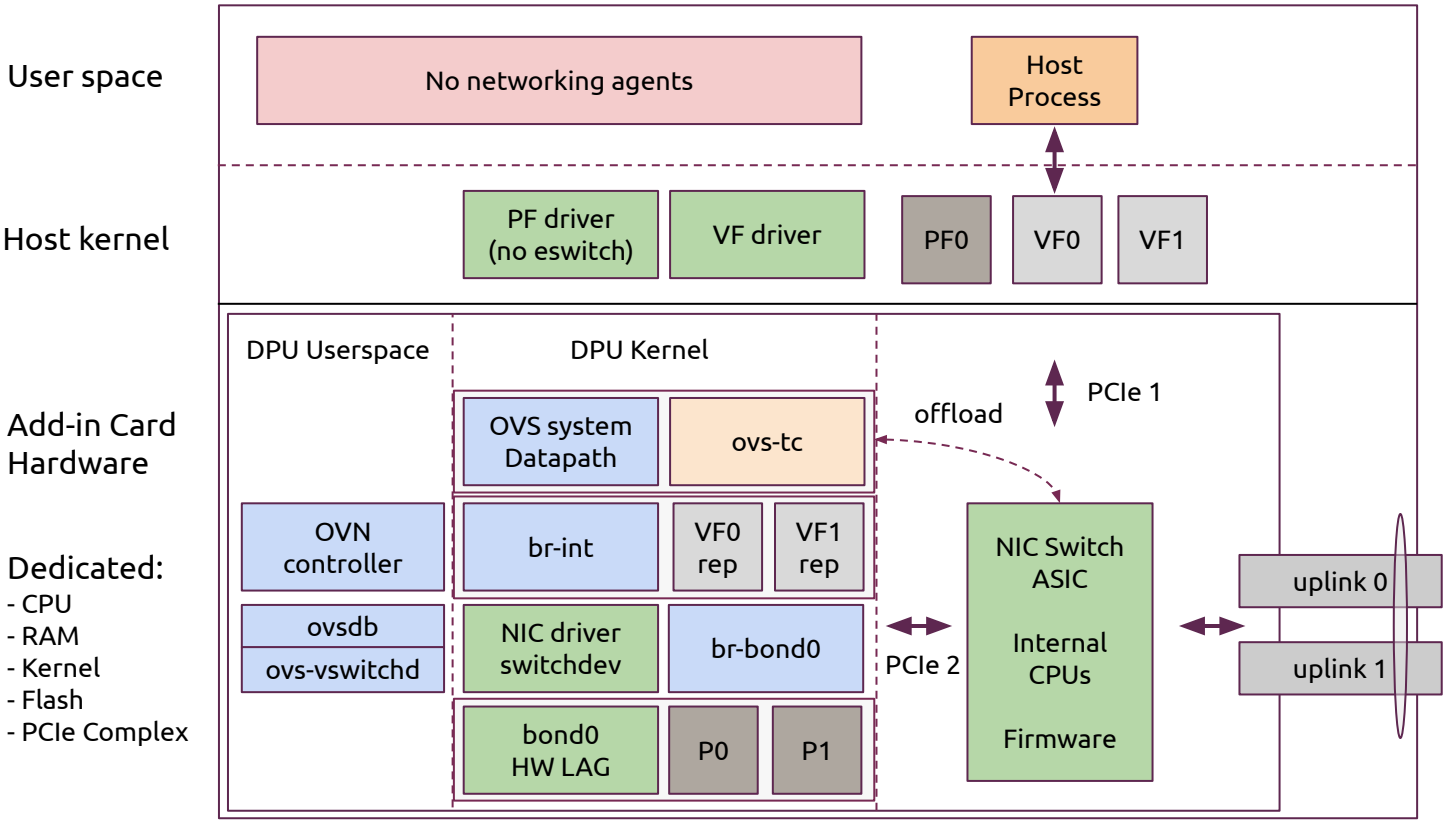
SmartNICs

- An overloaded term
- Some classifications include:
    - Packet processing flow:
        - Off-path SmartNICs
        - On-path SmartNICs
    - Hardware design:
        - ASIC-based
        - FPGA-based
- We will focus on **ASIC**-based **off-path** SmartNICs

Off-path SmartNIC



Off-path SmartNIC DPU



DPU: key takeaways

- Data Processing Unit (DPU)
  - Embedded system: **dedicated CPU**, NIC and other components
  - NIC is integrated with the main board using an I/O interconnect (e.g. PCIe)
  - NIC is shared by dedicated CPU and host CPU via **separate** I/O hierarchies
- Off-path architecture:
  - **Slow path**: packets flow via NIC cores
    - OVS system datapath (offload via tc)
    - OVS + DPDK (offload via rte_flow)
  - **Fast path**: direct flow via ASIC to the destination bypassing NIC cores

DPU: Control Plane Challenges

- Current infra software expects same-host topology (OpenStack, K8s, LXD etc.)
  - Hypervisor hostname ≠ OVS hostname on DPU
    - PCI addresses cannot be relied on to refer to PFs & VFs on different hosts
    - VF allocation on a hypervisor but programming on a DPU?
    - Hypervisor to DPU mapping? Multiple DPUs per host?
- Security boundary between host and DPU
  - How can we keep hypervisor & DPU isolation?
  - Generic host <-> DPU communication methods?

DPU: Control Plane Challenges

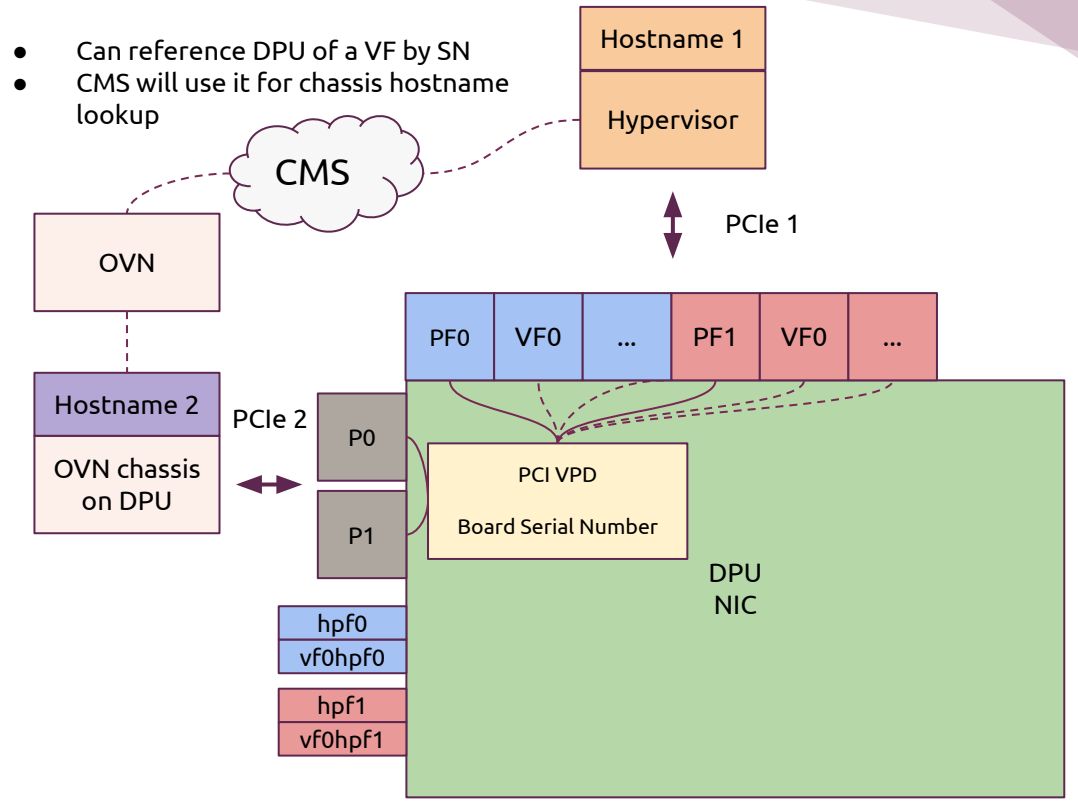
- CMS support
  - **Local** OVS & ovn-controller is expected for port/VIF plugging into bridges
  - Scheduling concerns: metadata about legacy SR-IOV vs offload vs DPU VFs
  - Missing control plane for remote VIF plugging and programming
  - Need intelligent chassis hostname lookup based on serial numbers
- OVN
  - Need to support port plugging directed by a CMS via **L**ogical **S**witch **P**ort info
  - Port plugging is traditionally outside of the OVN’s responsibility

DPU Support Design Goals

- Reuse the existing code as much as possible
- CMS-agnostic architecture (applicable to OpenStack, K8s, LXD etc.)
- Generic handling of different device families & vendors
- Upstream first approach
  - Rely on upstream projects and APIs (tc flower, OVS, switchdev, OVN)
  - Minimize new software or upstream it
- Avoid remote code execution on DPU by the hypervisor host
  - Analogy: treat DPU as a ToR Switch in a server

Auto-discovery

- Different hostnames
- Different PCIe topologies
- Same NIC
- Same PCI Vital Product Data (VPD)
- Same board serial number:
  - Unique
  - Read-only (ROM)
- Exposed on PFs if present
- Optionally on VFs



PCI/PCIe VPD

- Appeared in PCI 2.1 local bus spec, inherited in PCIe
  - fully compatible format in PCIe
- Optional in the specs
  - but modern NICs have it
- Board Serial Number: unique, read-only, factory assigned (ASCII alphanumeric)
- Exposed on PFs
  - Firmware may optionally expose VPD on VFs
- Linux 2.6.26+ exposes a binary VPD blob via sysfs

Why Board Serial?

- How and how many PFs are exposed depends on a particular device
  - Some HW can expose more PFs then there are uplinks
    - Depends on a firmware config and device family
  - Some PFs can be inactive when HW bonding is used
  - PFs are **virtual ports of the NIC switch** while we need a static entity
- MAC addresses are port-level IDs while the board SN is a board-level ID
  - MACs can reprogrammed while VPD SN cannot
  - What is a **burned-in MAC** for a **virtual** NIC switch port?
- => PF MAC usage becomes **unreliable** for DPU hostname mapping

OVN VIF Plug Provider framework

- Core OVN
  - CMS API
    - `Logical_Switch_Port options`
      - `requested-chassis`
      - `plug-type`
      - `plug-mtu-request`
    - Each VIF plug provider implementation provide namespaced options for lookup.
  - New `requested_chassis` column in Southbound DB populated by northd
    - Allows each `ovn-controller` to effectively monitor ports destined to it prior to having the port plugged
  - VIF plug providers register callbacks that OVN will use for lookup to inform insert/delete operations for ports/interfaces in the local Open vSwitch instance

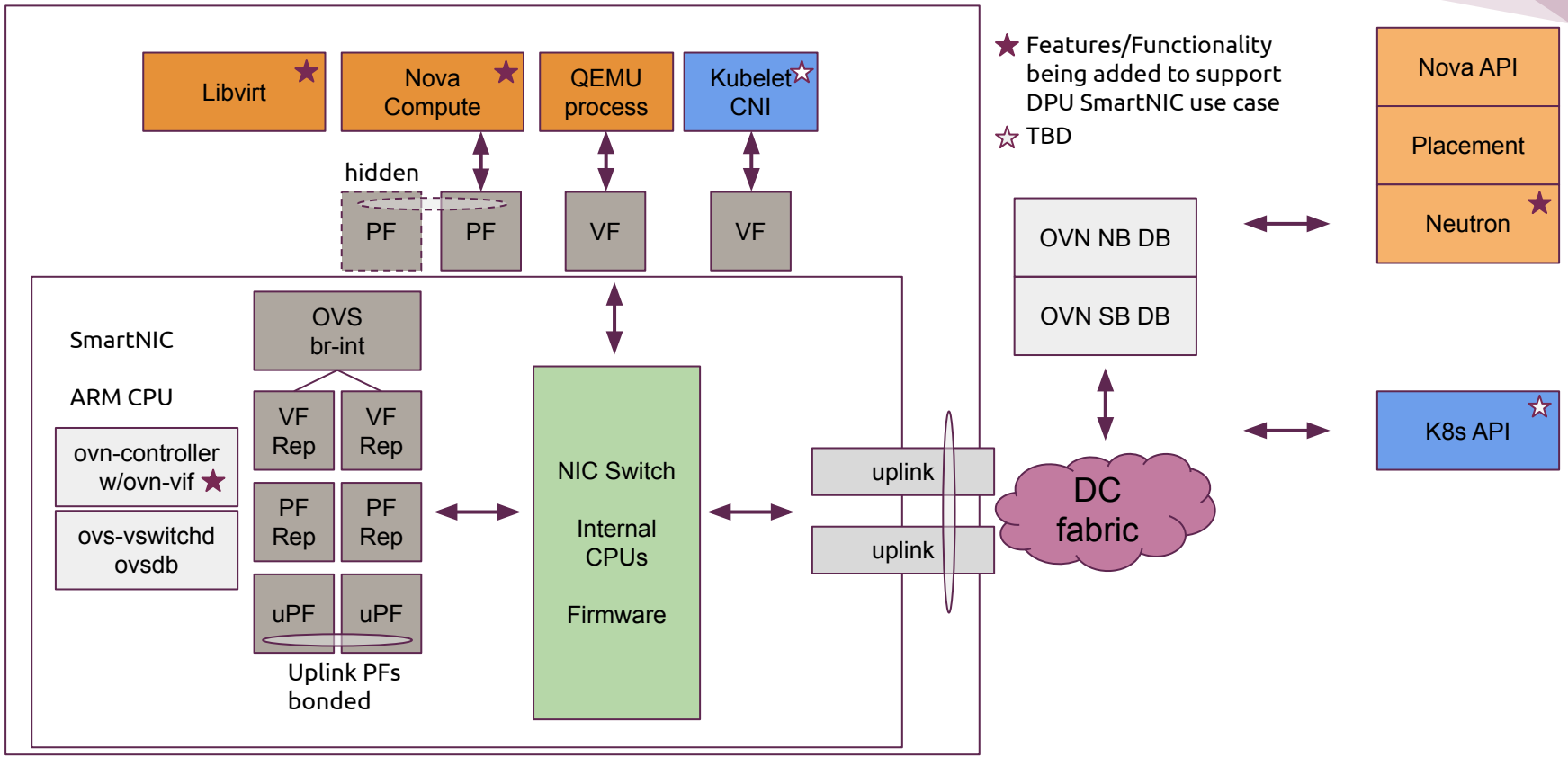
https://github.com/ovn-org/ovn/blob/main/Documentation/topics/vif-plug-providers/vif-plug-providers.rst

OVN VIF

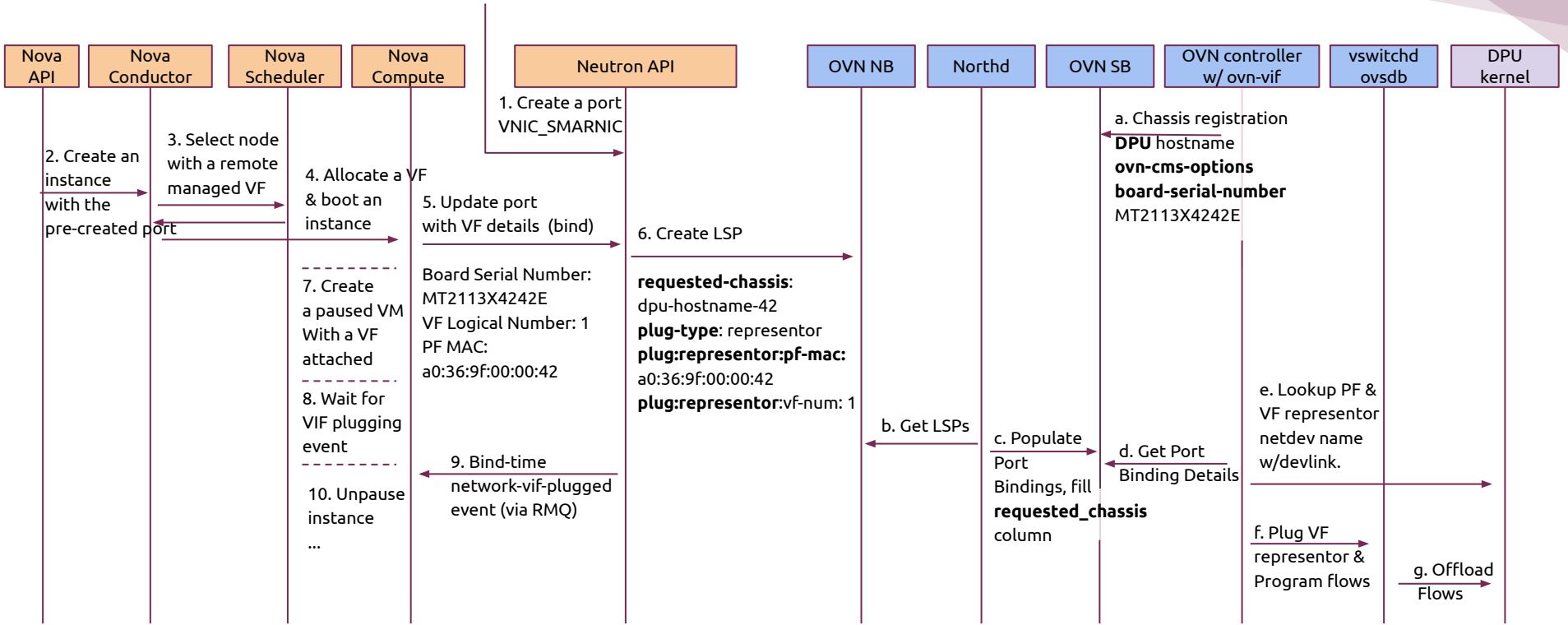
- OVN VIF
  - System and device specific code is kept in a separate project called OVN VIF
  - Ready to host multiple VIF plug provider implementations, will follow core OVN release cadence and branch strategy
  - GitHub PR based workflow, patches welcome!
    - https://github.com/ovn-org/ovn-vif

  - `representor`
    - CMS API
      - `Logical_Switch_Port options`
        - `vif-plug:representor:pf-mac`
        - `vif-plug:representor:vf-num`
    - Supports SmartNIC DPUs that provide kernel representor ports exposed through the devlink-port infrastructure
    - Provides devlink library based on the (great) Open vSwitch netlink library code
    - Support for runtime updates to lookup tables in progress

CMS integration: OpenStack & K8s



High Level Control Flow



Upstreaming Status

- OVN
  - Plug providers infrastructure upstreamed (will appear in 21.12)
  - New ovn-org/ovn-vif project created under ovn-org
- CMS
  - OpenStack Nova and Neutron specifications approved for Yoga
    - Nova and Neutron specs are merged
    - Nova and Neutron Code is submitted for review
  - Kubernetes
    - TBD

Call for Devices

- If you have a device you would like to test this on:
  - Ping us via e-mail/IRC
  - Send us a sample (or two)
- More devices tested will help us make adjustments if needed

Demo

Thank you. Questions?