OpenStack Enhancements to Support NFV Use Cases

Steve Gordon, Red Hat
Adrian Hoban, Intel
Alan Kavanagh, Ericsson
Agenda

- OpenStack Engagement Model for NFV
- Kilo Extensions for NFV
- Evolving the Cloud to Support NFV
OpenStack Engagement Model for NFV
How did we get here?

October 2012:
Business motivation of Network Functions Virtualisation (NFV) is described in a white paper written by several leading service providers and published at the “SDN and OpenFlow World Congress”, Darmstadt-Germany.

Early 2013:
ETSI formed an Industry Specification Group (ISG) to work on specifying certain aspects of NFV with industry partners using approach based on Open Source, Proof-of-Concepts by industry consortia and ‘running systems’.

September of 2014:
Industry launched ‘Open Platform for NFV’ (OPNFV) as Linux Foundation Collaborative Project to create a reference platform for NFV, identifying upstream projects, filling the gaps by developing code and executing integration in world wide labs.
ETSI NFV ISG

● Decoder ring:
  ○ European Telecommunication Standards Institute
  ○ Network Function Virtualization
  ○ Industry Specification Group

● Putting the standards in standards body!
ETSI NFV ISG

● Phase 1:
  ○ Convergence on network operator requirements
  ○ Including applicable existing standards
  ○ Developing new requirements to stimulate innovation and open ecosystem of vendors
ETSI NFV ISG

● Phase 2:
  ◦ Grow an interoperable VNF ecosystem
  ◦ Thoroughly specify reference points and requirements defined in Phase 1
  ◦ Achieve broader industry engagement
  ◦ Clarify how NFV intersects with SDN and related standards/industry/open source initiatives.
NFV ARCHITECTURE
Open Platform for NFV (OPNFV)

- Establishing open source reference platform including:
  - NFV Infrastructure (NFVI)
  - Virtual Infrastructure Management (VIM)

- Focused on:
  - Consistency, performance, interoperability between components.
  - Working with existing upstream communities.
NFV ARCHITECTURE
Open Platform for NFV (OPNFV)

Growing list of projects:
- Requirements projects
  - E.g. Fault Management
- Integration and Testing projects
  - E.g. IPv6 enabled OPNFV
- Collaborative Development projects
  - E.g. Software Fastpath Service Quality Metrics
- Documentation projects
Telco Working Group

● Mission:
  o Identify Telco/NFV use cases
  o Define and prioritize requirements internally
  o Harmonize inputs into OpenStack projects
    ▪ Blueprint/patch creation, submission, and review.
● Move discussion closer to OpenStack projects.
OpenStack

- Large community of technical contributors in wide array of loosely governed projects.
- NFV requirements fall across many of these.
- Require buy in from these diverse groups of contributors.
OpenStack

- Most projects moving to “specification” process for approval of major changes
- Ingredients of a good specification:
  - Problem description incl. use cases
  - Concrete design proposal
  - Someone to implement!
Working Together

Success!
Current State

• Overlap exists between various groups in:
  - Mission
  - Membership
  - Scope
  - Activities

• Navigating can be tough!
Working from both ends

- ETSI NFV
- OPNFV
- Telco Working Group
- OpenStack
Working from both ends

- ETSI NFV
- OPNFV
- Telco Working Group
- OpenStack

Merging of “Worlds” happens here!
Kilo Extensions for NFV
Based on OpenStack community contributions & collaborations
Non Uniform Memory Architecture (NUMA)

- Memory Proximity
  - Performance and latency characteristics differ depending on the core a process is executing on and where the memory a process is accessing is located.

Optimising placement for memory proximity enables greater performance & efficiency
Filter Extensions: NUMA

- `numa_topology_filter`
  - Helps to co-locate CPU core allocations to a single socket (when possible)
  - Resource tracks core/socket consumption and filters to available subset of suitable platforms.

Co-location helps with cache efficiency for faster inter-process data communication.
Filter Extensions: NUMA

- **numa_topology_filter**
  - Helps to co-locate CPU core allocations to a single socket (when possible)
  - Resource tracks core/socket consumption and filters to available subset of suitable platforms.

Enables the OSes to allocate local memory for greater performance & efficiency
Filter Extensions: NUMA – I/O Awareness

- Adds ability to select the socket based on the I/O device requirement
- E.g. What if you’d prefer network access on NIC B
Filter Extensions: NUMA – I/O Awareness

- Adds ability to select the socket based on the I/O device requirement
- E.g. What if you’d prefer/require network access on NIC B?
Simultaneous Multi-Threading (SMT)

- **SMT**
  - On Intel platforms, run 2 threads at the same time per core
- **Take advantage of wide execution engine**
  - Keep it fed with multiple threads
  - Hide latency of a single thread
- **Power efficient performance feature**
  - Very low die area cost
  - Can provide significant performance benefit depending on application
  - Much more efficient than adding an entire core

SMT enhances performance and energy efficiency
Simultaneous Multi-Threading (SMT)

- Sample Linux enumeration of cores
- Linux scheduler (in the host) manages work load (process) allocation to CPUs

![Diagram showing processor sockets and execution units]
CPU Pinning – “Prefer” Policy (In Kilo)

Prefer Policy: Place vCPUs on pCPU siblings (when SMT is enabled)
CPU Pinning – “Separate” Policy (For Liberty)

Separate Policy: Scheduler will not place vCPUs from same guest on pCPU siblings
CPU Pinning – “Isolate” Policy (For Liberty)

Isolate Policy: Nova will not place vCPUs from any pCPU that has an allocated sibling
CPU Pinning – “Avoid” Policy (For Liberty)

Will not deploy Guest OS A or B on this server

Avoid Policy: Nova scheduler will not place the guest on a host with SMT enabled
Huge Page Tables

- Translation Lookaside Buffer (TLB)
  - Memory component that accelerates address translation.
  - Caches a subset of address translations from the page table.

- Huge page table sizes (e.g. 1 GB)
  - TLB caches a greater range of memory translations
  - Helps reduces TLB misses.

Small page table entries (4KB) can result in a greater number of TLB misses.
Optimize Host for NFV
- Huge Page Table and CPU Isolation

Comput Node(s)

• Edit `/etc/default/grub`
  • `GRUB_CMDLINE_LINUX="intel_iommu=on default_hugepagesz=2MB hugepagesz=1G hugepages=8 isolcpus= 1, 2, 3, 5, 6, 7, 9, 10, 11, 13, 14, 15"`

• `sudo grub-mkconfig -o /boot/grub/grub.cfg`
• `sudo reboot`
Optimize for NFV: Create Host Aggregate

• Create aggregate for NFV usage
  • nova aggregate-create nfv-aggregate
  • nova aggregate-set-metadata nfv-aggregate nfv=true

• Add hosts to the NFV aggregate
Optimize for NFV: Create Host Aggregate

- **N.B.: Good practice to create an aggregate for non-NFV use cases**
  - nova aggregate-create default-usage
  - nova aggregate-set-metadata default-usage nfv=false
  - **Update all other flavours to include the meta-data**
    - nova flavor-key <flavour-name> set aggregate_instance_extra_specs:nfv=false
  - Add hosts to the default aggregate
Optimize for NFV: /etc/nova/nova.conf

[default]
pci_alias=

pci_passthrough_whitelist=


Optimize for NFV: /etc/nova/nova.conf

[default]

Optimize for NFV: `/etc/nova/nova.conf`

```yaml
[libvirt]
cpu_mode=host-model or host-passthrough
vcpu_pin_set=1,2,3,5,6,7,9,10,11,13,14,15
```
Optimize for NFV: ml2_conf.ini

- Configure
  /etc/neutron/plugins/ml2/ml2_conf.ini
  
  [ml2]
  tenant_network_types = vlan
  type_drivers = vlan
  mechanism_drivers =
      openvswitch,sriovnicsswitch
  
  [ml2_type_vlan]
  network_vlan_ranges = physnetNFV:50:100
Optimize for NFV: ml2_conf_sriov.ini

- Configure
  
  
  /etc/neutron/plugins/ml2/ml2_conf_sriov.ini
  
  [ml2_sriov]
  
  supported_pci_vendor_devs = 8086:10fb
  
  agent_required = False
  
  [sriov_nic]
  
  physical_device_mappings = physnetNFV:eth1
Optimize for NFV: Create VNF Flavor

• `nova flavor-create nfv-node auto 1024 0 4`

• `nova flavor-key nfv-node set`
  hw:cpu_policy=dedicated
  hw:cpu_threads_policy=prefer
  capabilities:cpu_info:features=aes
  pci_passthrough:alias=niantic:1
  aggregate_instance_extra_specs:nfv=true
Optimize for NFV: Create VNF Flavor

- `nova flavor-key nfv-node set`
  
  `hw:numa_nodes=1`
  
  `hw:numa_cpus.0=0,1,2,3`
  
  `hw:numa_mempolicy=strict`
  
  `hw:numa_mem.0=1024`
  
  `hw:mem_page_size=2048`
Optimize for NFV: Create Network

- `neutron net-create` with options:
  - `-provider:physical_network=physnetNFV`
  - `-provider:network_type=vlan NFV-network`

- `neutron subnet-create` with options:
  - `NFV-network <CIDR>`
  - `-name <Subnet_Name>`
  - `-allocation-pool=<start_ip>, end=<end_ip>`

- `neutron port-create` with options:
  - `NFV-network` and `--binding:vnic-type direct`
Optimize for NFV: Boot VNF VM

- nova boot --flavor nfv-node --image <image> --nic port-id=<from port-create command> <vm name>
Other Notable Changes:

- New ML2 OVS driver for ovs+netdev-dpdk
  - High Performance User Space based vSwitching
  - High Performance path to the VM (vHost User), with new VIF type in Nova.
- Available on stackforge/networking-ovs-dpdk
- Supports DVR in VLAN and VXLAN modes
Other Notable Changes:

- **VLAN Trunking API Extension**
  - New network property that indicates requirement for transparent VLANs
  - ML2 drivers that indicate that they do not support transparent VLANs or do not have the attribute will fail to create the transparent network.
  - LB, VXLAN and GRE drivers support VLAN transparent networks
  - The VLAN and OVS drivers do not support VLAN transparent networks

- **Service VM Port Security (Disable) Extension**
  - Neutron's security group always applies anti-spoof rules on the VMs.
  - This allows traffic to originate and terminate at the VM as expected, but prevents traffic to pass through the VM. Disabling security is required in cases where the VM routes traffic through it.
VNF Deployment Considerations & Future Work
Evolution of node delivery
Customized application and Hardware

- Custom Hardware
- Platform
- Host OS
- Bin/Libs
- APPLICATION

Optimized Host OS for custom hardware

Application designed based on Custom hardware and OS

Optimized Service
Application running on Industry Standard High Volume Standard

Optimized
VNF on
Bare metal

Optimized Host OS

Application designed based on x86 Platform

HDS 8000

Industry Standard Hardware

Host OS

Bin/Libs

APPLICATION

Ericsson | Page 49
Type 2 Hardware Virtualization

<table>
<thead>
<tr>
<th>VM</th>
<th>Guest OS</th>
<th>Bin/Libs</th>
<th>APP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VM</td>
<td>Guest OS</td>
<td>Bin/Libs</td>
<td>APP</td>
</tr>
</tbody>
</table>

Hypervisor

Host OS

Physical Server

Virtual Machine

H/w Emulation: Expose Instruction
Set to Guest OS

CPU Support for Virtualization
Intel® VT-x, Intel® VT-d: H/W Emulation

App run inside VM

Any Guest
Linux Containers

Containers share same OS Kernel and are separated by “Name Spaces”

Linux Containers on bare metal

Application sharing common libraries and Kernel

Containers share same OS Kernel and are separated by “Name Spaces”
Containers inside VM

**VM-1**
- Container
  - APP
  - Bin/Libs
- Guest OS

**VM-2**
- Container
  - APP
  - Bin/Libs
- Guest OS
### Which deployment option suits my vnf?

<table>
<thead>
<tr>
<th>Hypervisor</th>
<th>baremetal</th>
<th>container</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Provides density and isolation to run different “Guest OS”</td>
<td>• Applications that consume all resources on the blade and mission critical applications</td>
<td>• Suitable for VNF/Apps that can share a common kernel</td>
</tr>
<tr>
<td>• Virtualising the h/w platform for VNF’s to run on any x86 machine</td>
<td>• Infrastructure applications that perform high user plane and control plane packet processing</td>
<td>• Offers high form of density and removal of multiple guest and hypervisor overheads</td>
</tr>
<tr>
<td>• Platform Resources e.g. CPU and Memory can be shared or dedicated and allocated to different VNF's</td>
<td>• Dedicated resource isolation due to regulatory requirement</td>
<td>• H/W acceleration support in progress</td>
</tr>
<tr>
<td></td>
<td>• No hypervisor license fee, i.e.CAPEX reduction, removes overhead and potential layers of failure</td>
<td>• Reduced isolation compared to VM’s</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>answer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• All 3 Deployment options are needed</td>
<td>• VNF/Apps will benefit differently in each deployment option</td>
<td>• By supporting all 3 deployment options in an IaaS Manager, we can support all possible VNF/Apps type deployment models</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Summary

- Transparent collaboration between ETSI-NFV, OPNFV, Telco-WG and OpenStack core projects vital to enabling OpenStack for NFV.
- Making steady but meaningful progress on NFV enablement.
- Hypervisor, bare metal and container deployment options in an IaaS system are needed to support all possible VNF/Apps types.
References - Contributing

- **ETSI:**
  - [https://portal.etsi.org/TBSiteMap/NFV/NFVMembership.aspx](https://portal.etsi.org/TBSiteMap/NFV/NFVMembership.aspx)

- **OPNFV:**
  - [https://www.opnfv.org/developers/how-participate](https://www.opnfv.org/developers/how-participate)

- **TelcoWG:**

- **OpenStack:**
  - [https://wiki.openstack.org/wiki/How_To_Contribute](https://wiki.openstack.org/wiki/How_To_Contribute)
Intel Disclaimers

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. No computer system can be absolutely secure. Check with your system manufacturer or retailer or learn more at [intel.com].

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Intel, the Intel logo, Intel® Advanced Encryption Standard New Instructions (Intel® AES-NI) are trademarks of Intel Corporation in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others.

© 2015 Intel Corporation.