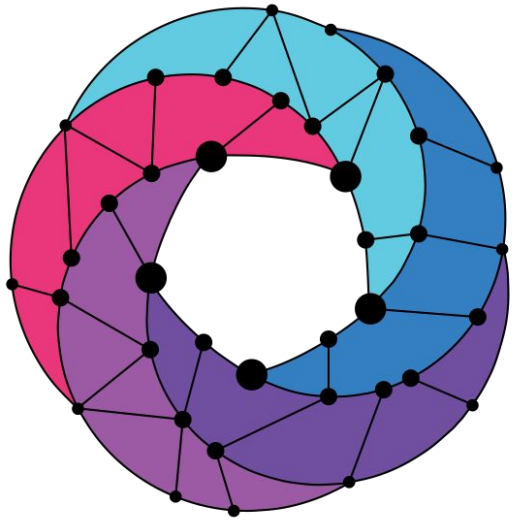


Open Source
MANO

OSM#9 Hackfest

Enabling high performance on VNFs

Gianpietro Lavado (Whitestack)

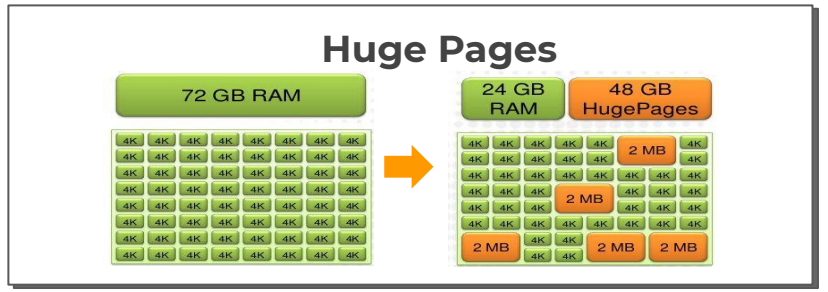
ETSI

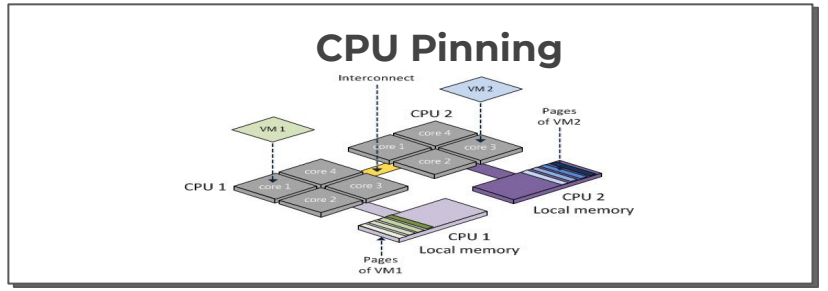
EPA (Enhanced Platform Awareness)

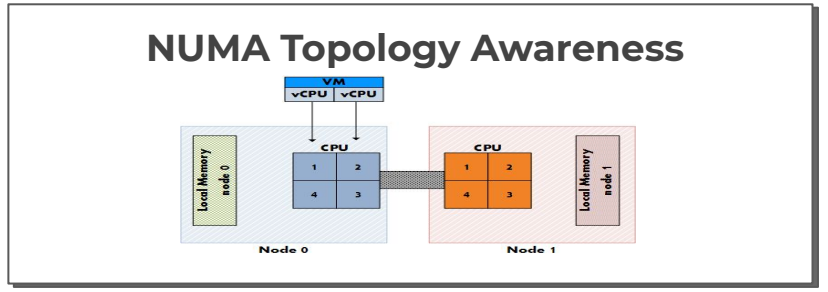
- **EPA covers a set of techniques for getting more performance for virtual instances in a given NFVI.**
- **EPA features** like the use of large hugepages memory, dedicated CPUs, strict NUMA node placement, the use of passthrough and SR-IOV interfaces, **can be used in OSM's VNF descriptors since Rel ZERO**.
- If your VIM supports EPA, then you don't need to do anything extra to use it from OSM. VIM connectors in OSM take advantage of EPA capabilities if the VIM supports it. All you need to do is build your descriptors and deploy.

EPA examples

- **Huge pages:** allocates big chunks of memory to avoid having too many entries to look up.
- **CPU Pinning:** binds a set of CPUs to a VM, improving the performance by avoiding degrading events such as cache misses.
- **NUMA (Non-Uniform Memory Access):** it provides separate memory for each processor to avoid the decrease on performance when several processors attempt to address the same memory.
- **SR-IOV (Single Root Input/Output Virtualization):** specification that allows the isolation of a PCI Express resource (e.g. a network interface) for manageability and performance reasons.

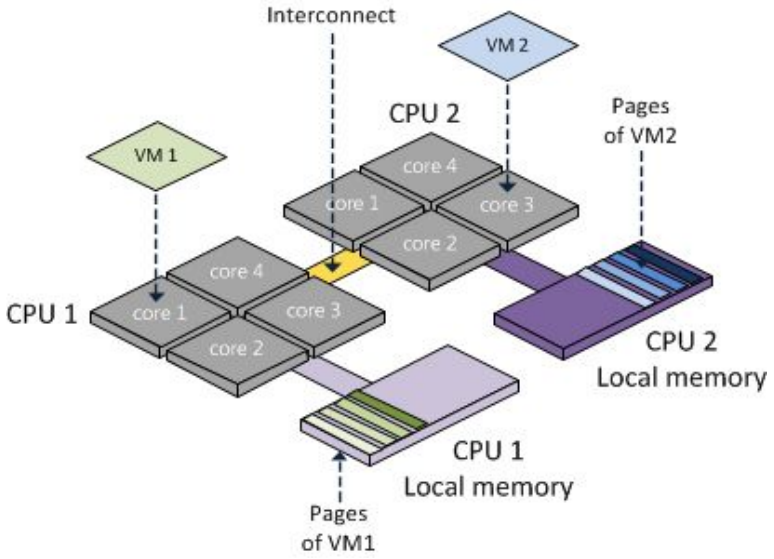




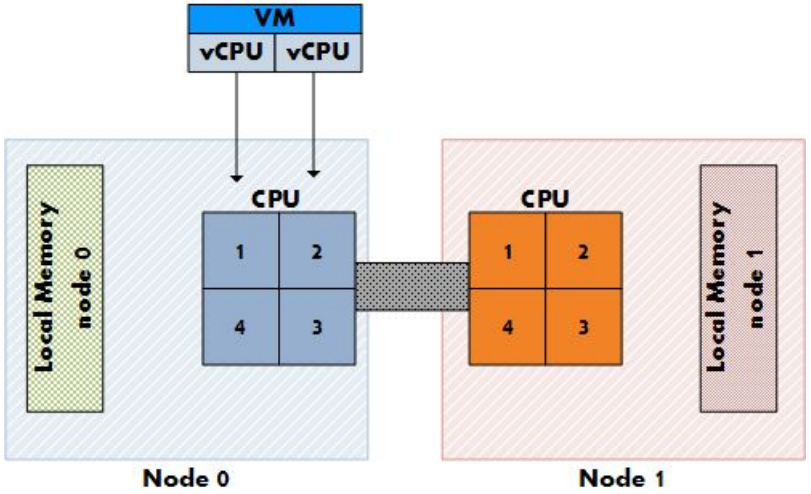


CPU Pinning & NUMA Awareness

Most NFVI/VIMs support **CPU Pinning** and **NUMA Topology Awareness** capabilities without any need for configuration.



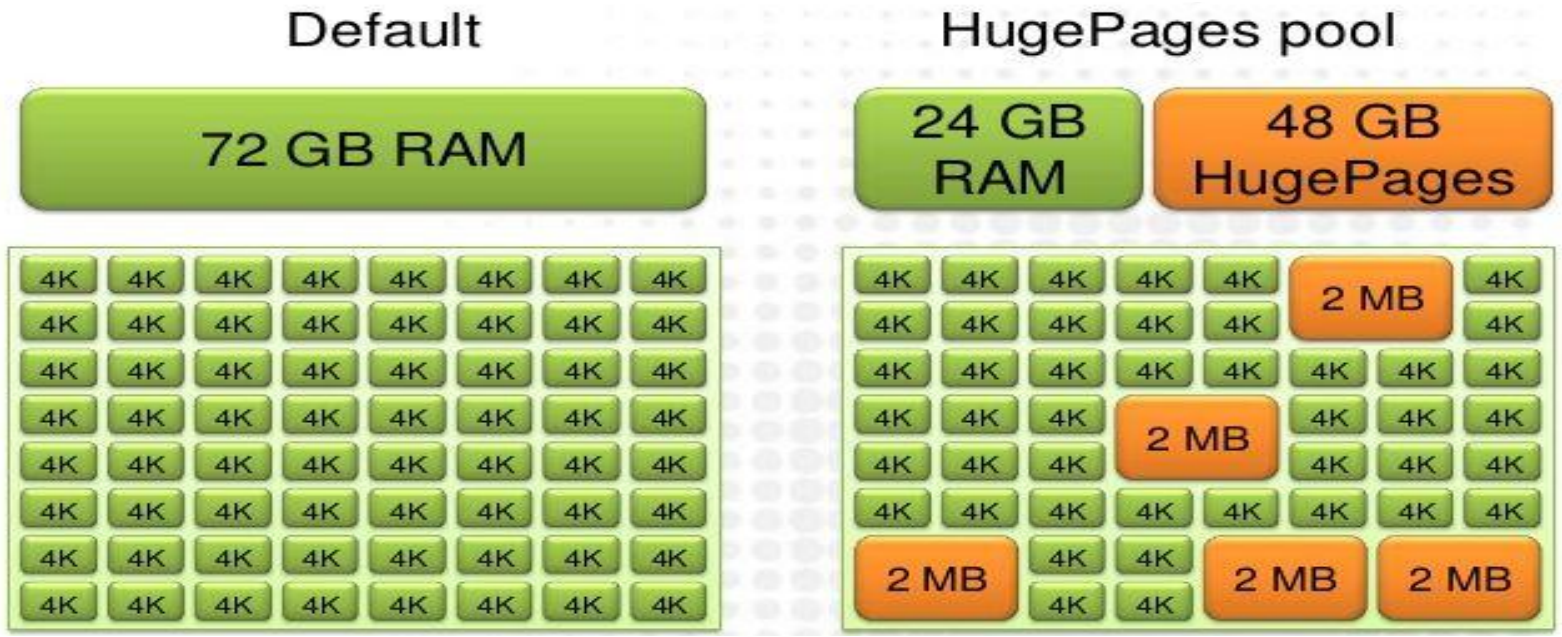
CPU Pinning: being able to pin a VM to specific CPUs



NUMA Topology Awareness: making the VM aware of the physical CPU topology

Memory Huge Pages

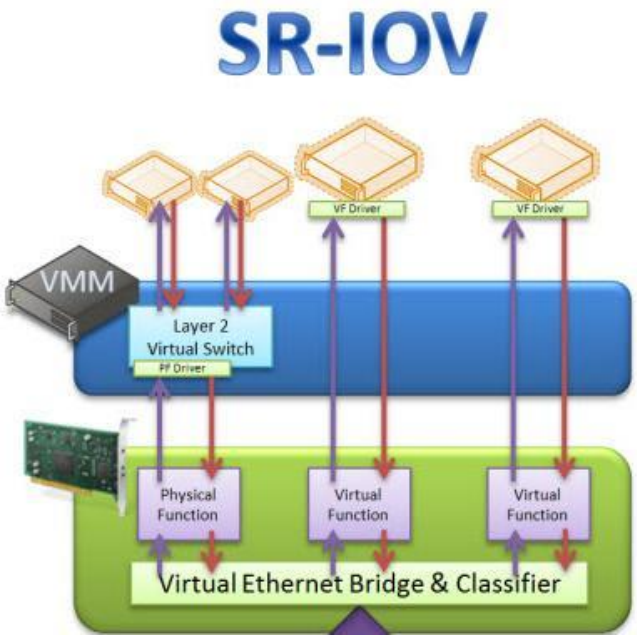
Memory Huge Pages allows the VNFs to request RAM memory from a special pool where page sizes are bigger, enabling better performance.

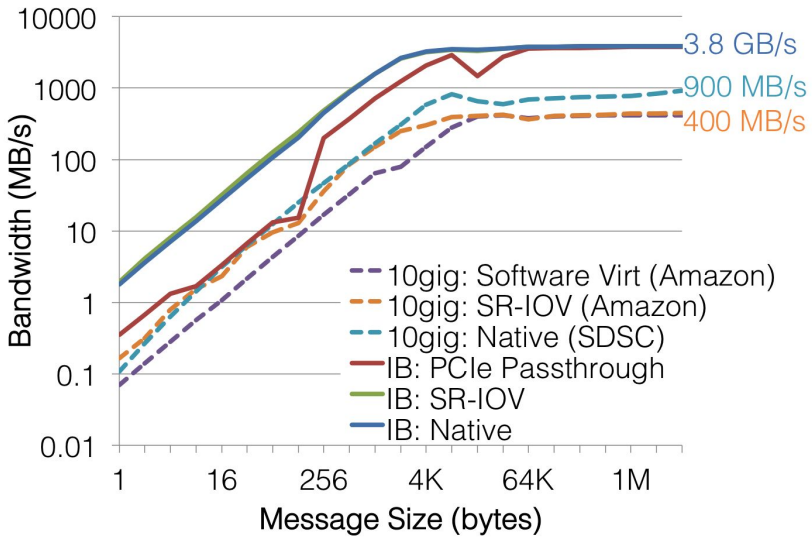


<u>Enabling/changing Huge Pages require a node reload</u>, and the NFVI servers to allocate a new memory pool with bigger pages, this will not allow VMs set with normal pages to use this new pool, so it should be limited.

SR-IOV

SR-IOV allows VNFs to have direct access to a virtualized PCI of a NIC, thus giving it better throughput.





SDSC Early SR-IOV results

Enabling SR-IOV requires a node reload for reconfiguration of the IOMMU virtualization mode. It also requires physical interfaces to be dedicated to this feature.

Comparing SR-IOV provisioning (per VM/NS)

Open Source MANO

Provisioning SR-IOV with OpenStack

1. Create a network that uses the SR-IOV physnet
   - `openstack network create --provider-network-type=vlan --provider-physical-network=physnet2 --provider-segment=110 sriov-vlan110`
2. Create the corresponding subnet
   - `openstack subnet create --no-dhcp --network=sriov-vlan110 --subnet-range=11.0.0.0/24 sriov-vlan110-subnet`
3. Create a port with the direct binding at the subnet
   - `openstack port create --network sriov-vlan110 --vnic-type=direct --binding-profile trusted=true sriov-vlan110-port01`
4. Launch the VM using that port
   - `openstack server create --flavor m1.large --image bionic --nic port-id=sriov-vlan110-port01 vm01`
5. Repeat as needed, then configure the switches

Automating SR-IOV connectivity for the complete NS with OSM

1. Add the VIM that includes the physnet (once)

```
osm vim-create --name VIMSRIOV --user user --password password --auth_url
http://172.21.7.5:5000/v3 --tenant tenant --account_type openstack --config
'{dataplane_physical_net: physnet2, microversion: 2.32}'
```

2. Model your VNF(s) to use SR-IOV instead of VIRTIO

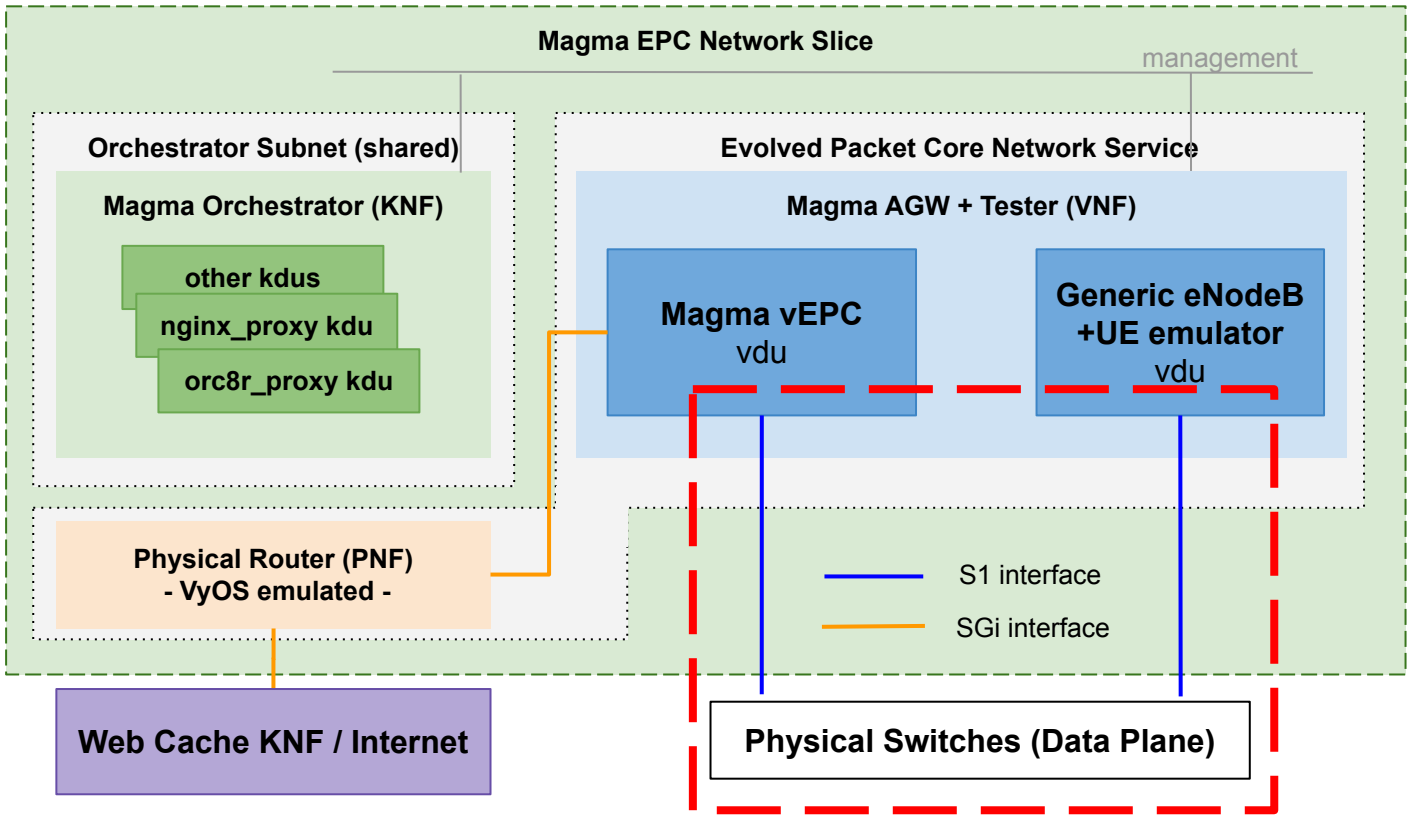
```
interface:
-   name: dataVDU
    type: EXTERNAL
    virtual-interface:
        type: SR-IOV
```

3. Launch your NS, OSM will take care of everything, including VLD interconnectivity if the fabric was pre-configured with SDN Assist.

Note: In all cases, the image you use should have the driver for supporting the physical NIC

EPA in our Network Service

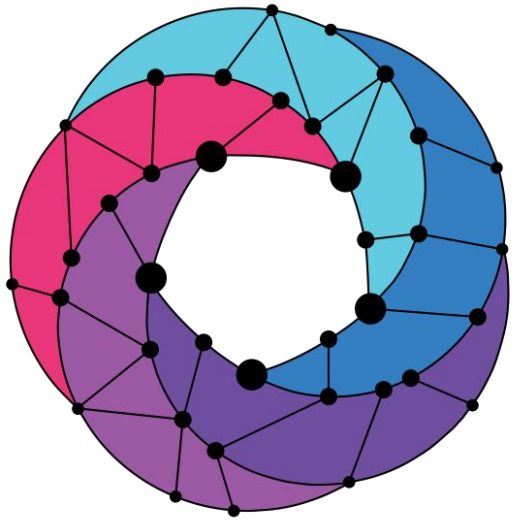
In our example, we can configure the S1 data interface, currently using VIRTIO drivers (OVS/VxLAN) to use SR-IOV instead. We can also set the descriptor to request CPU Pinning, memory Huge Pages, and stick the VDUs to a single NUMA node.
Today in OSM, all these optimizations are applied automatically when selecting SR-IOV in one of the interfaces, in order to match the packet processing capabilities that the direct connection to the NIC will allow for.



Example on how to model EPA and SR-IOV

Explore the Day-0 VNF Onboarding Guidelines for more details on the different options available

```
vnfd-catalog:
    vnfd:
    -   id: hackfest_magma-agw-enb_vnfd
        ...
        vdu:
        -   id: magma-agw-vdu
            guest-epa:
                cpu-pinning-policy: DEDICATED
                mempage-size: LARGE
            interface:
            -   name: eth0
                type: INTERNAL
                position: 1
                virtual-interface:
                    type: SR-IOV
        ...
        -   id: srsLTE-vdu
            ...
            interface:
            ...
            -   name: eth1
                type: INTERNAL
                virtual-interface:
                    type: SR-IOV
```



Open Source MANO

Find us at:

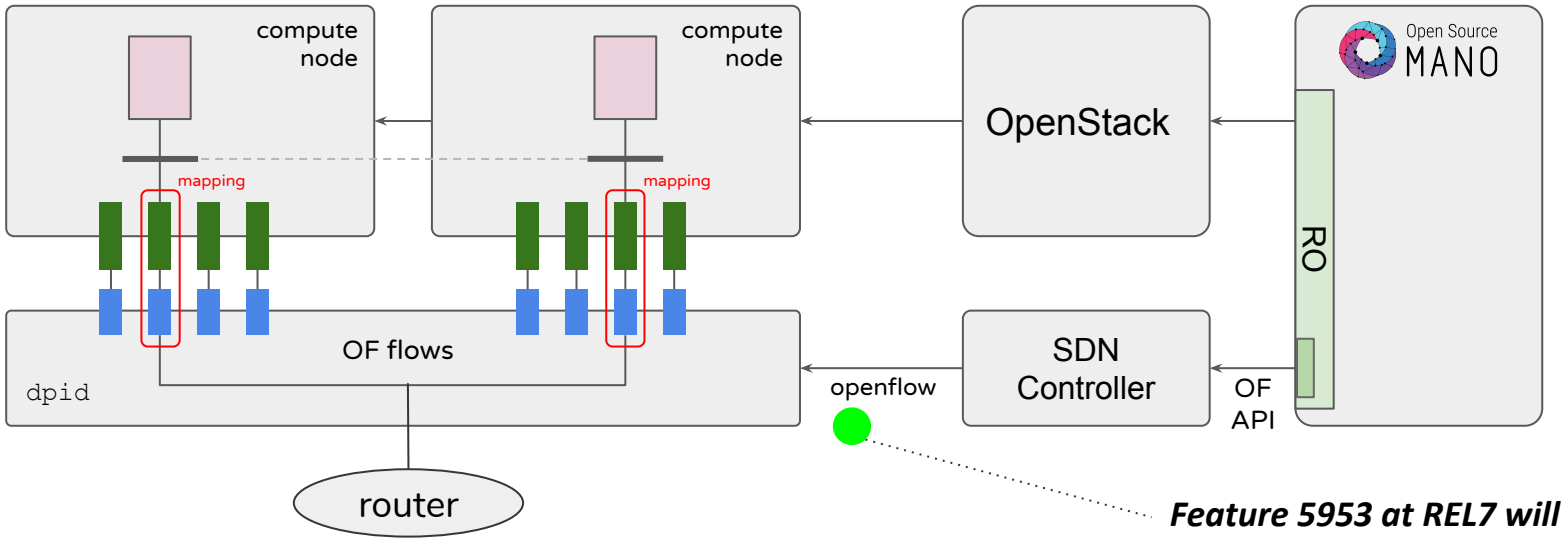
osm.etsi.org
osm.etsi.org/wikipub

SR-IOV + SDN Assist:
Enabling the chaining of **high performance** VNFs

1. OSM orquestrates SR-IOV

   → Proper assignment of I/O physical interfaces to the VM (VFs = Virtual Functions)

1. OSM SDN Assist gives the ability to create L2 connections between VFs
   - Interconnecting VMs
   - Attaching external traffic sources



Feature 5953 at REL7 will implement higher level protocols

DEMO: Deployment of two VNFs with EPA+Passthrough:
Underlay connectivity with ONOS SDN controller

1. OSM orchestrates two VNF: Gen & vBNG. Each one with:

   → EPA: 10 cores, 32 GB huge pages, 2 pci passthrough interfaces

1. VIM deploys and allocate physical interfaces to the VM
2. OSM SDN Assist creates requeted L2 connectivity