Distributing connectivity management in Cloud-Edge infrastructures: challenges and approaches

David ESPINEL

Journées Cloud 2019 - Toulouse

September 12 2019

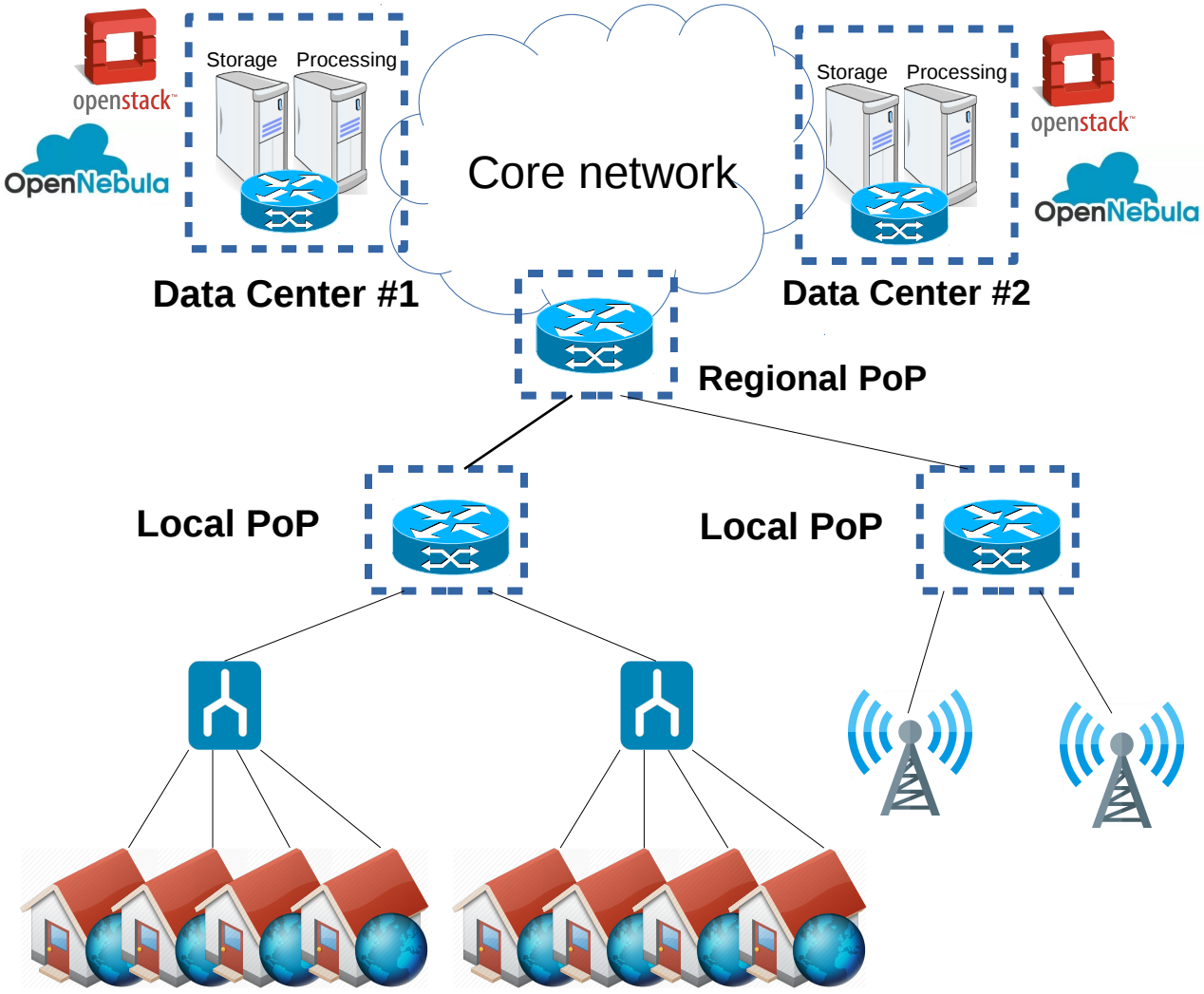
interne France Télécom - Orange

Presentation

- Context
- Required networking services
- Networking-related challenges
- Distributed inter-site services module

Context

Traditional IaaS architecture

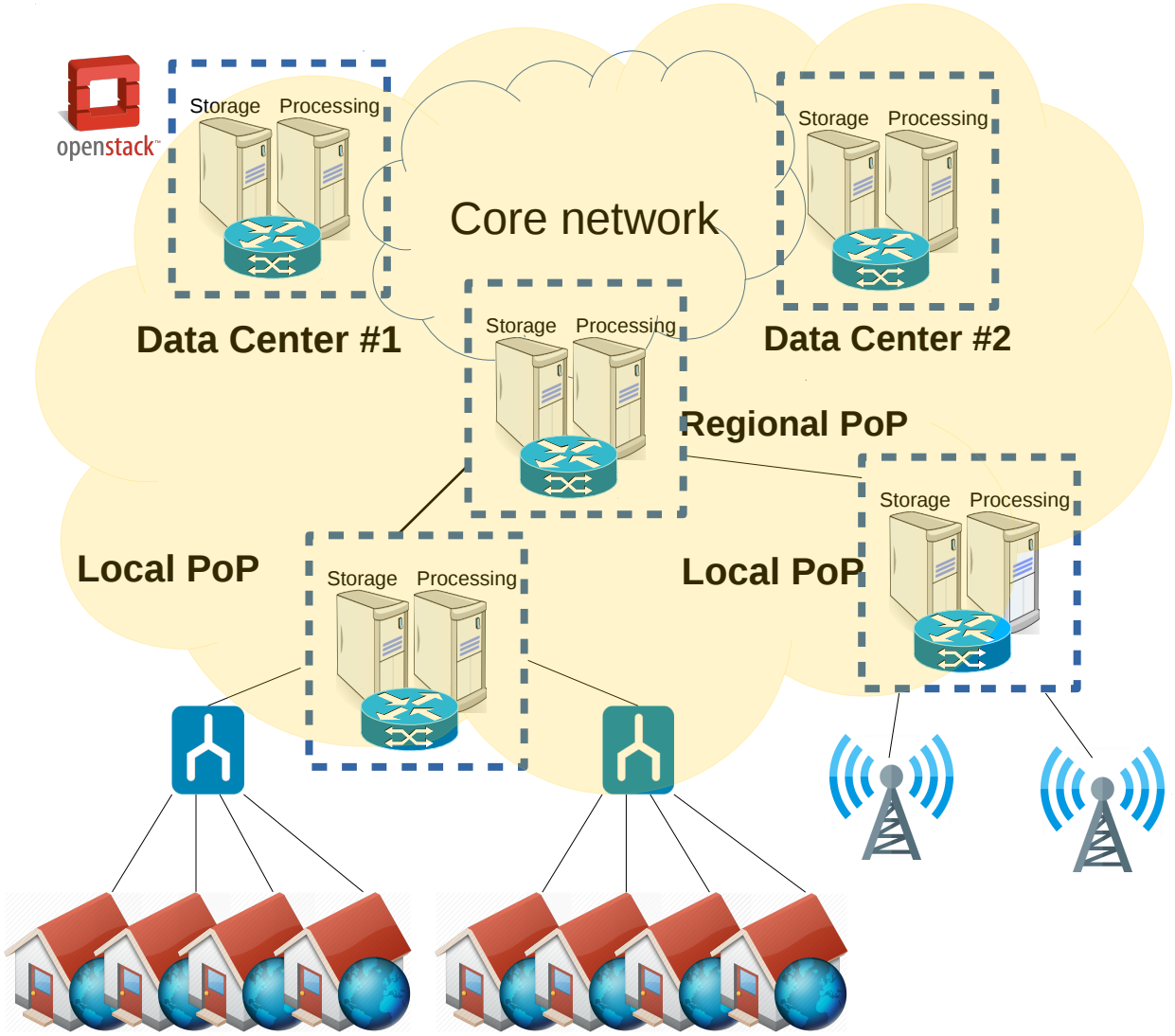


New emerging needs challenging the centralized approach:

- IoT
- NFV
- MEC

Context

Spanning the VIM to the entire infrastructure

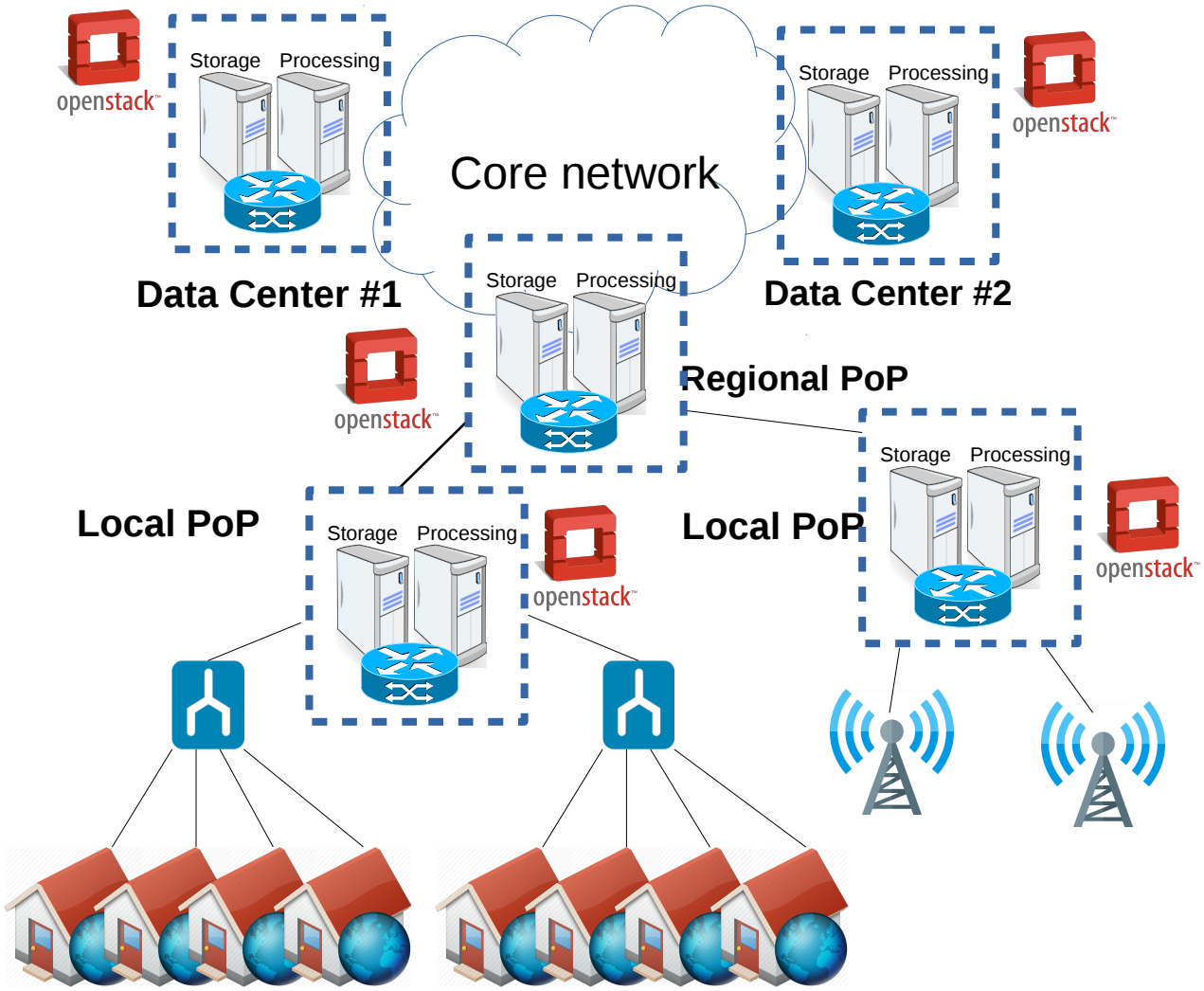


Single centralized model

- SPOF
- Bottleneck
- Network partitioning
- Implementation performance

Context

Leveraging a distributed IaaS architecture

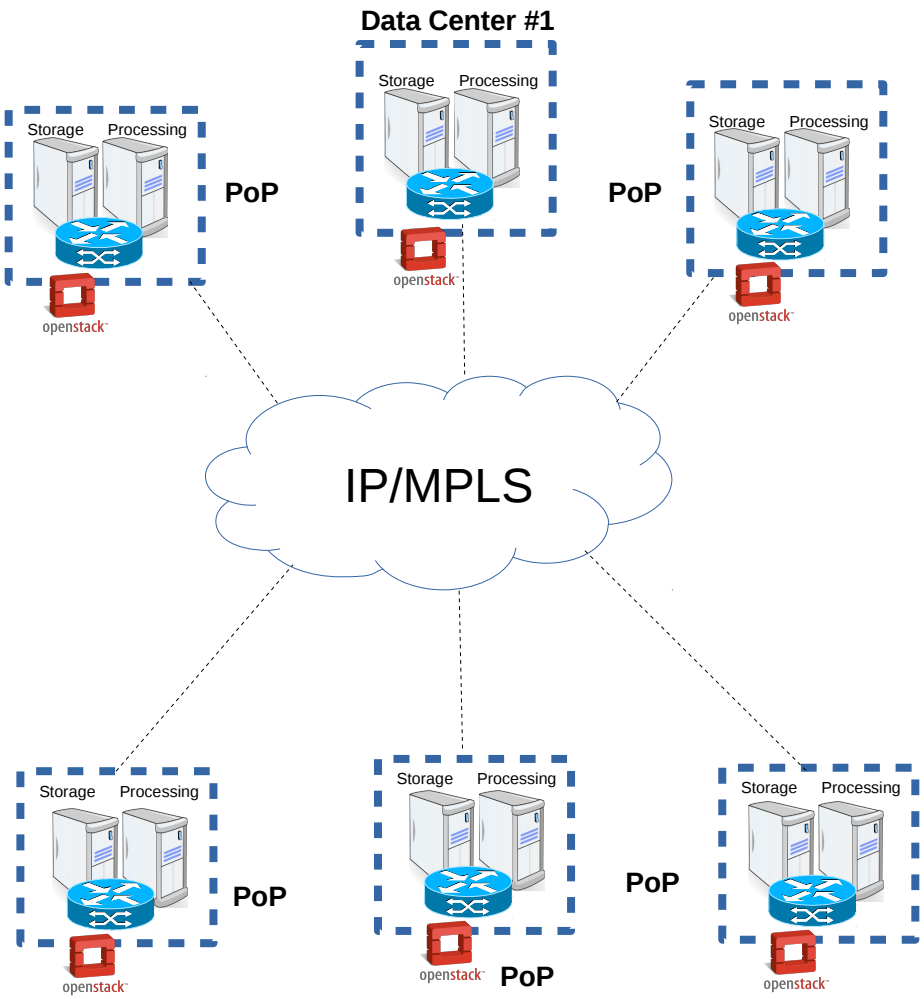


Needs to be closer to the user:

- Delay constraints
- Traffic overhead
- Legal requirements

Context

Leveraging a distributed IaaS architecture: Management questions



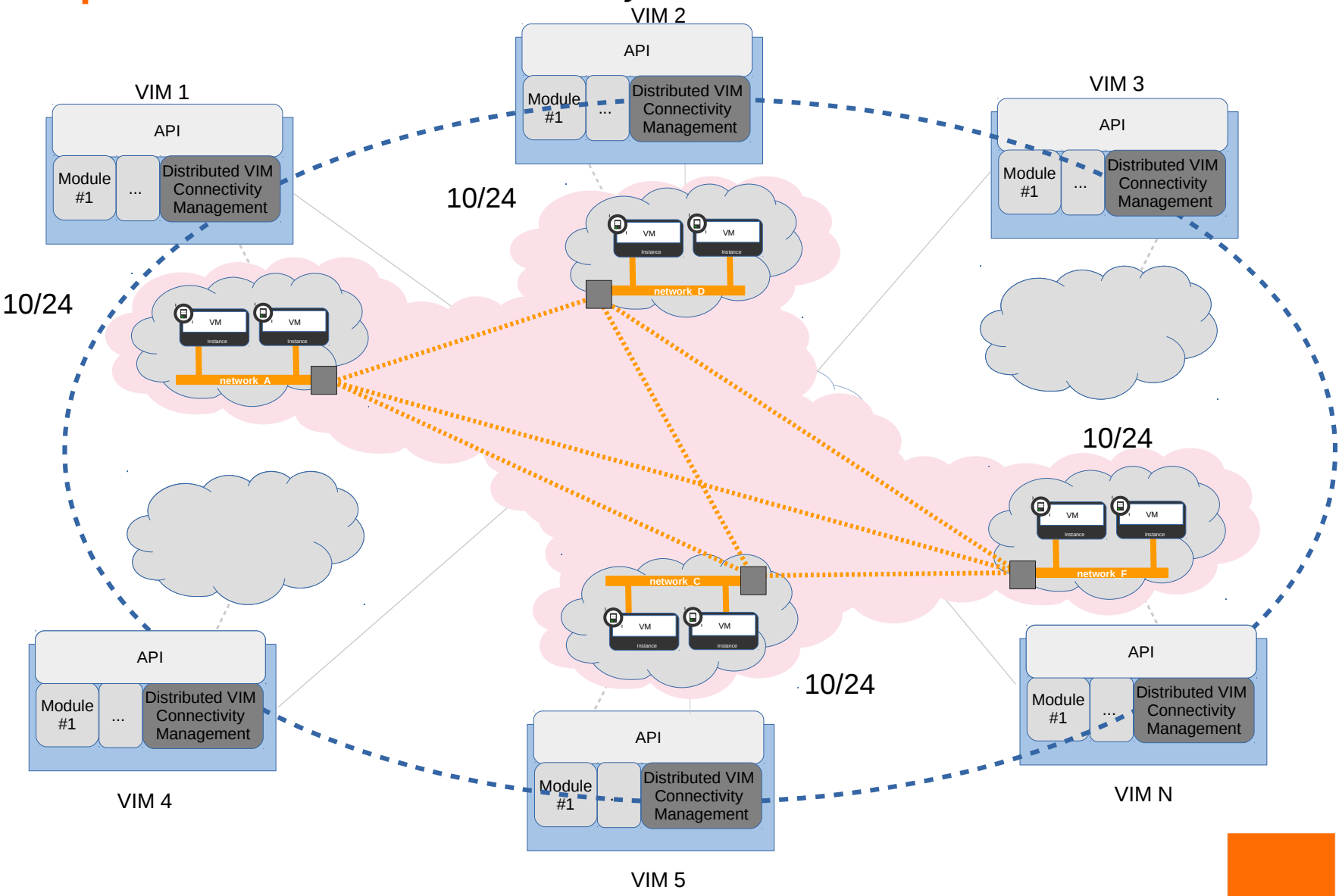
How to manage and interconnect networking constructions belonging to different VIMs in a distributed natively way?

While providing:

- *Scalability*
- *Resiliency*
- *Data locality awareness*
- *Abstraction & automation*

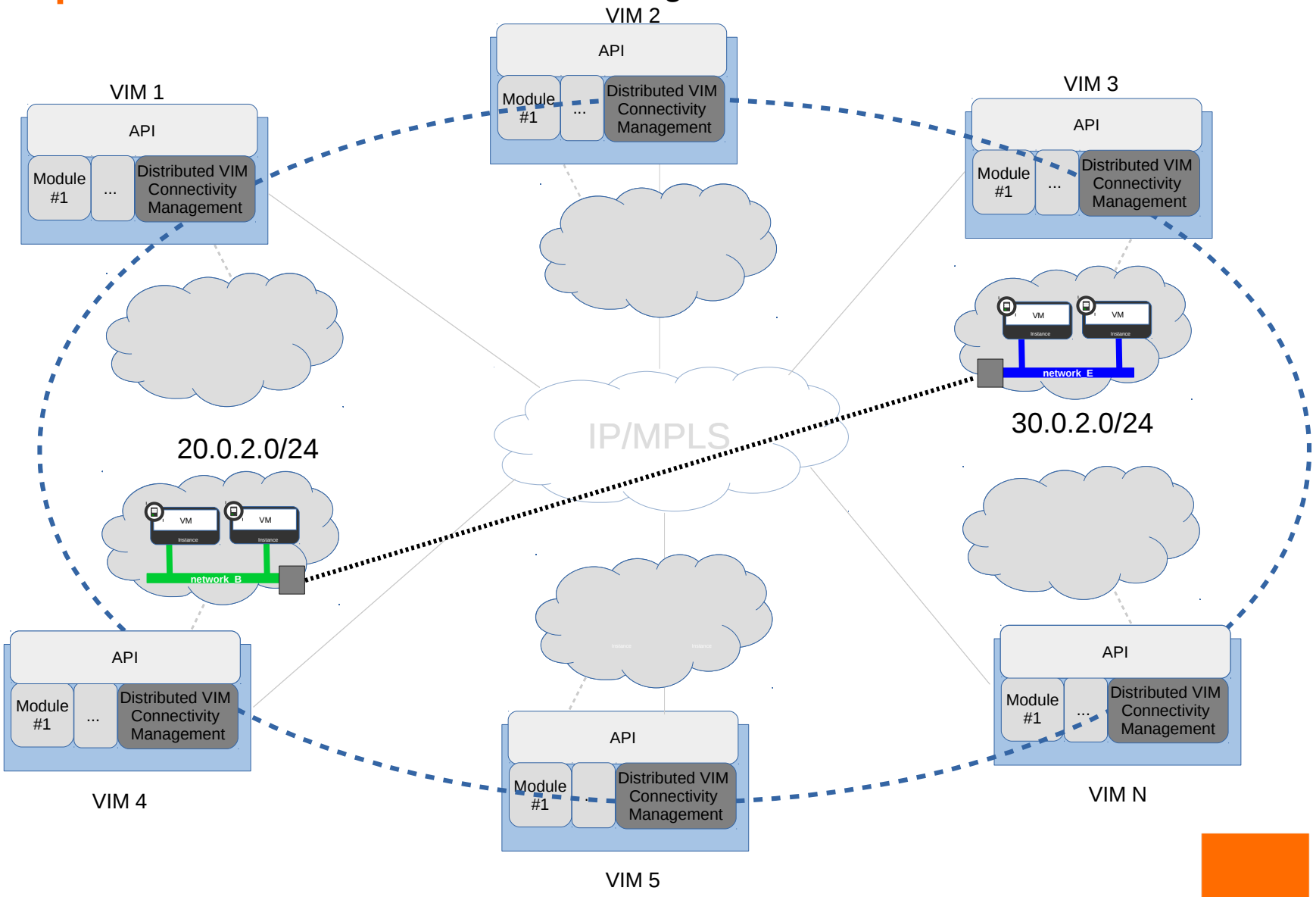
Required services

Layer 2 network extension



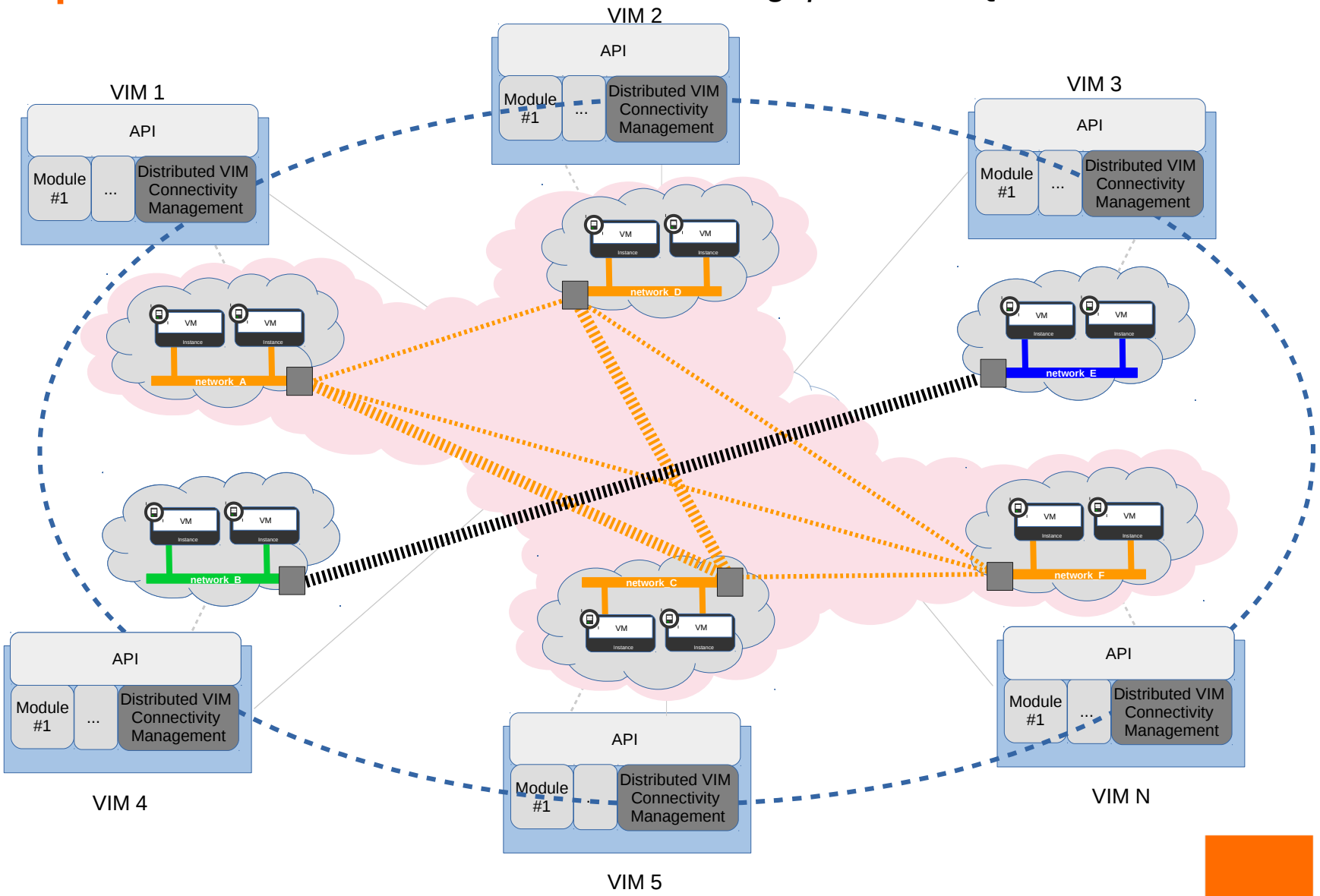
Required services

Routing function



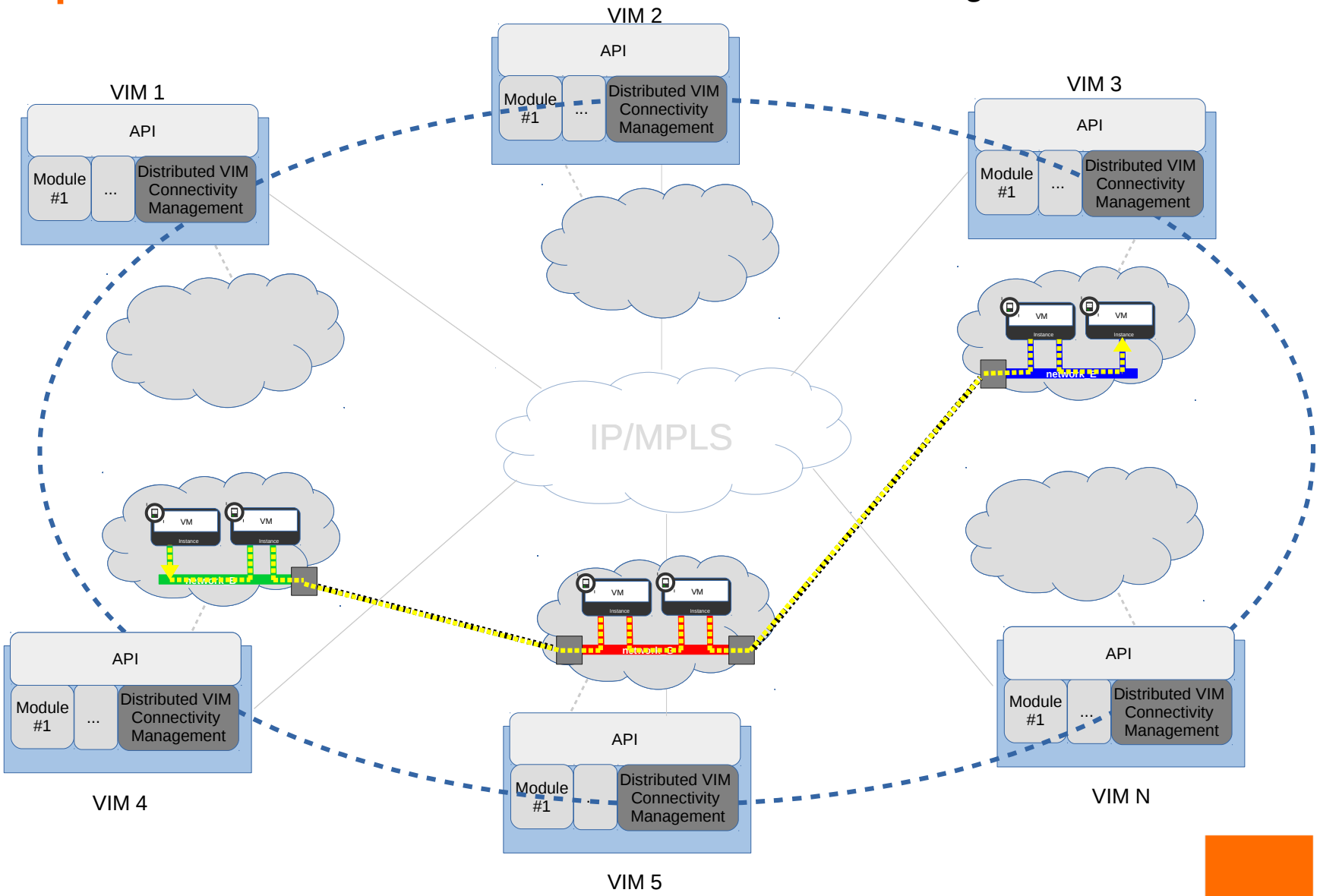
Required services

Traffic filtering, policies & QoS



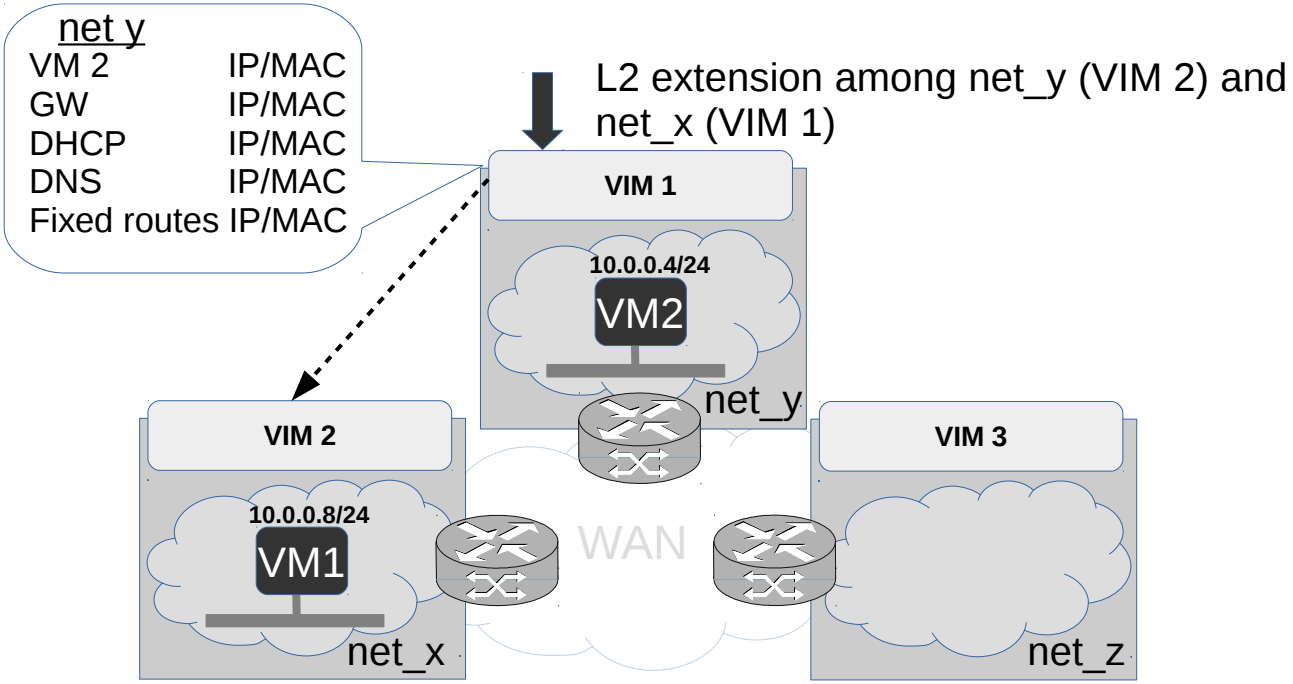
Required services

Service Function Chaining



Information granularity

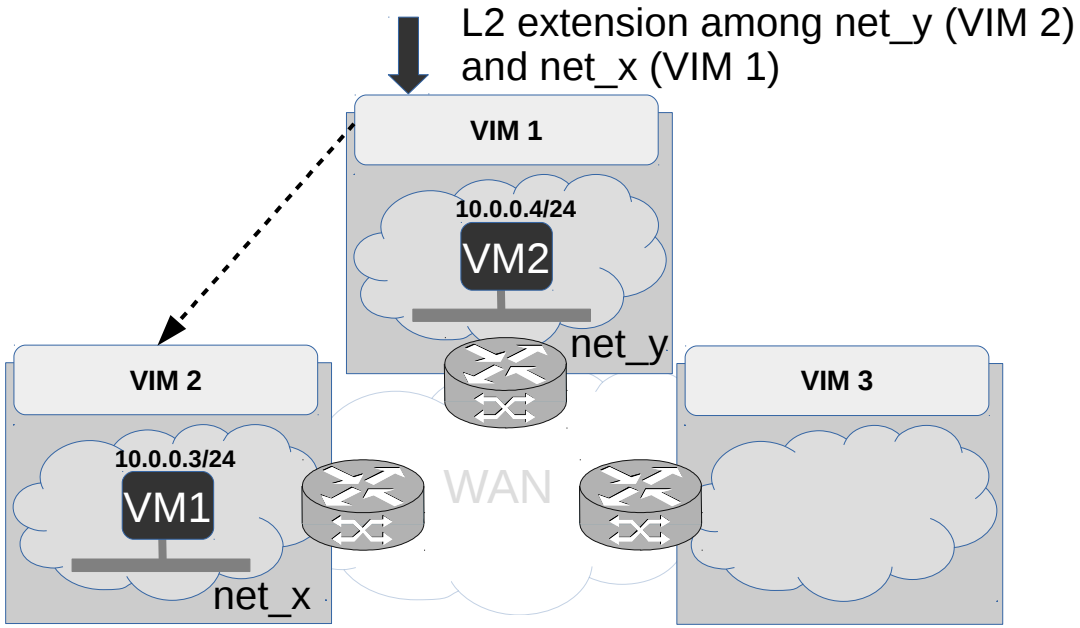
Challenge #1



- *Which information for which service?*
- *Identifying how information should be shared*
- *Different sharding strategies*

Information scope

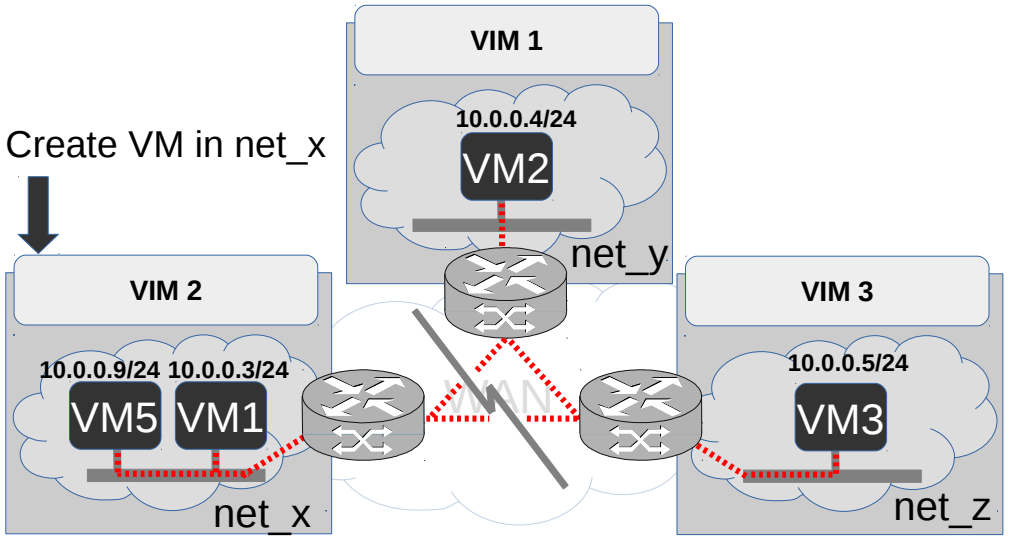
Challenge #2



- *Avoid unnecessary management traffic*
- *Share info only when requested*
- *Contact only the relevant sites*
- *Sharing networking information on-demand and in an efficient manner*

Information availability

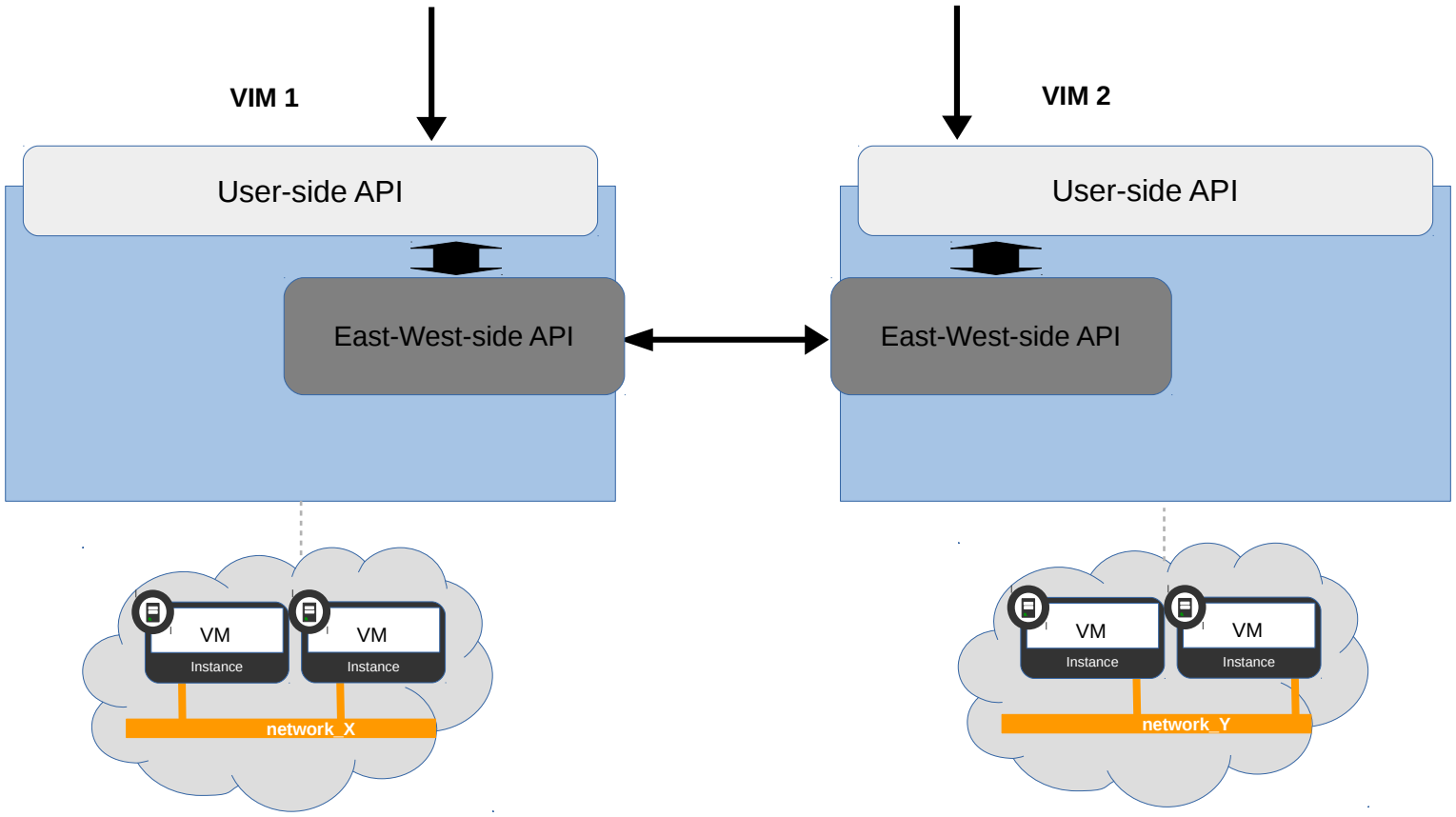
Challenge #3



- *Each VIM should be able to deliver network services*
- *Non-disconnected sites continue to provide inter-site services*
- *Inter-site traffic forwarding needs to be assured*

Automatized interfaces

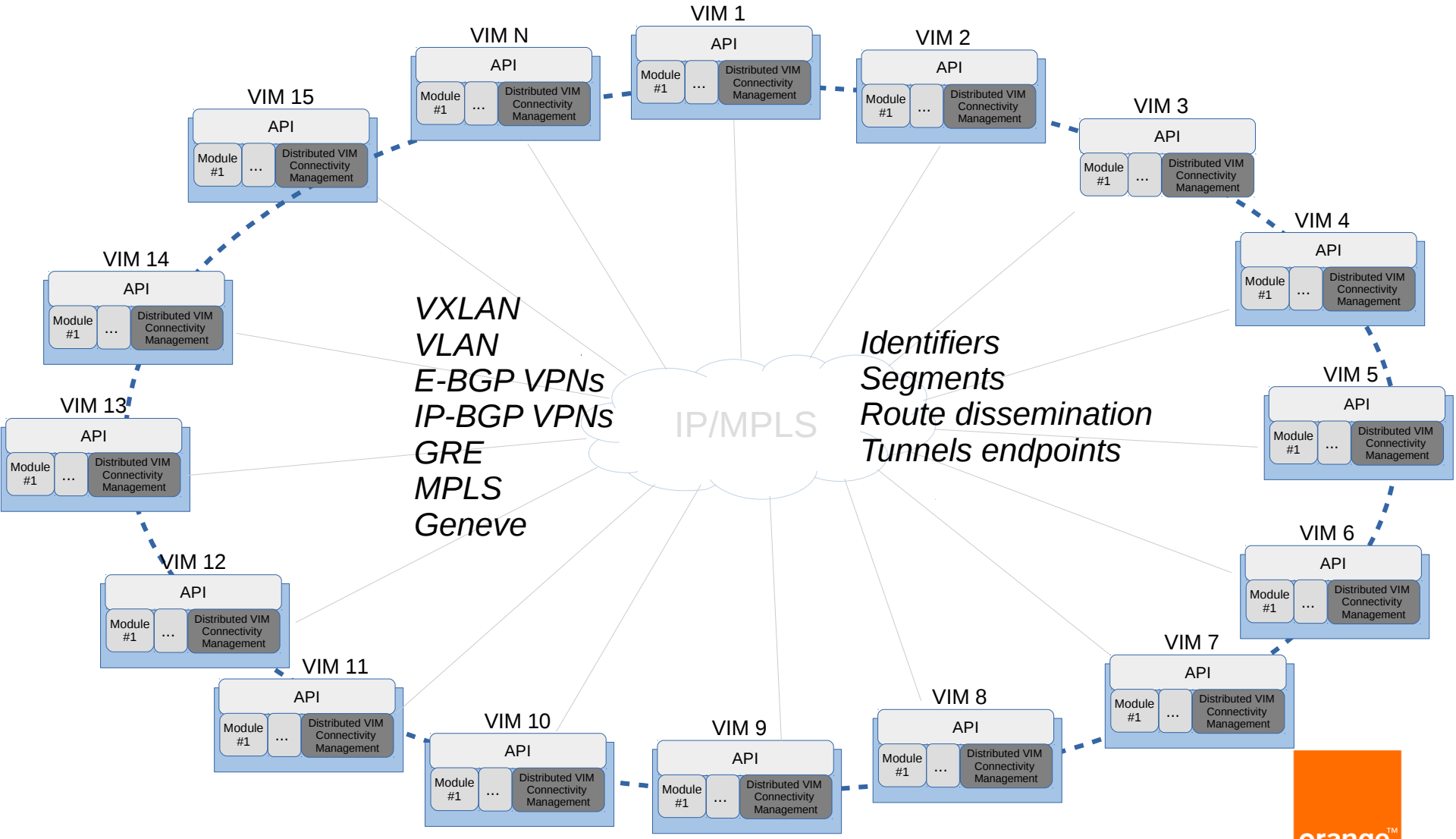
Challenge #4



- *Management integration challenge*
- *Smoothly bridge of the two interfaces*
- *Abstracted and automated process*

Networking technologies

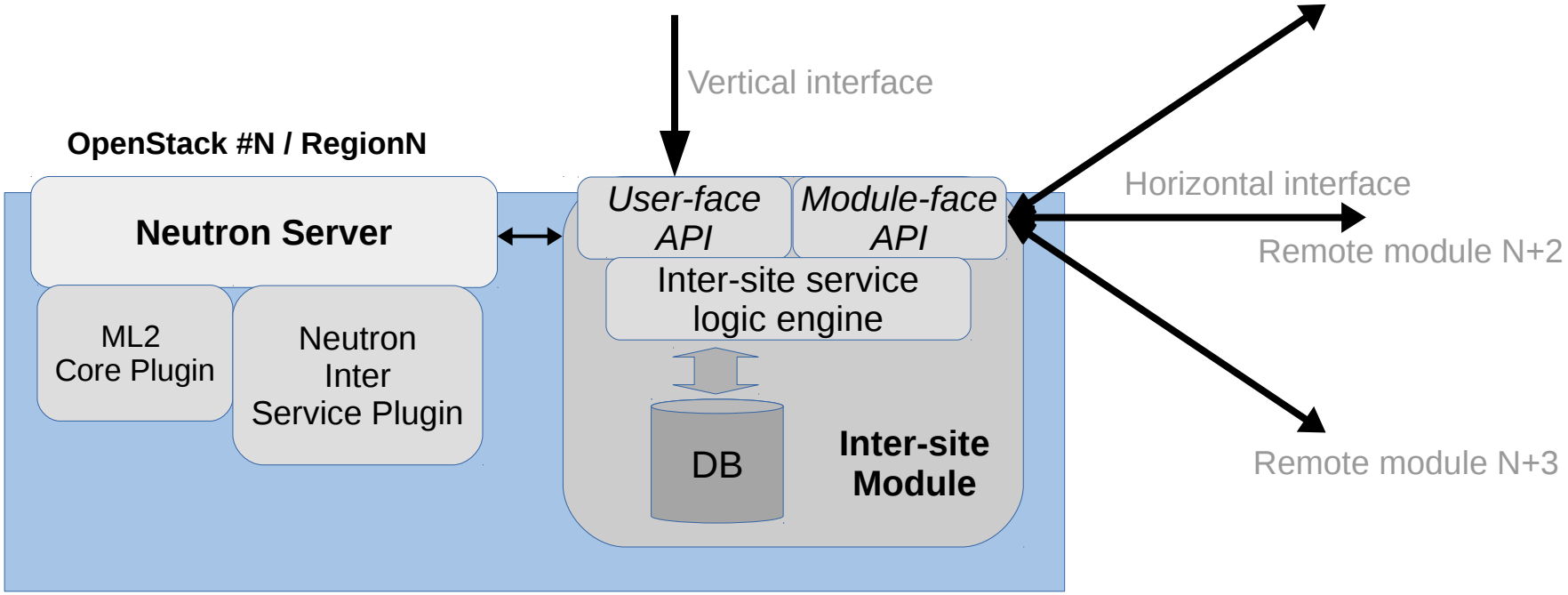
Challenge #5



Distributed inter-site services module

DISCO & ODL(Federation + NetVirt) inspired

- Type of service (L2 or L3/ QoS and SFC pending)
- List of N resources that will compose the service



Distributed inter-site services module

DISCO & ODL(Federation + NetVirt) inspired

Algorithm 4 Vertical inter-site networking services CREATE request

Require: N_i : *net id of network*$_i$ *in Site*$_i$

procedure

SEND: Vertical API CREATE inter-site $\{type : L3/L2, resources : \{N_i, N_(i+1), N_(i+2), ..., N_{n-1}, N_n\}_{i=1}^{n}\}$ to $Site_1$

function VERTICAL API CREATE INTER-SITE($type, resources$):

```
local ← resources.findLocal()
resources_subnets ← { }
resources_subnets ← get(N_i.getSubnet()_{i=1}^n)
subnet_CIDRs ← { }
subnet_CIDRs ← get(resources_subnets_i.getCIDR()_{i=1}^n)
if type ==' L3' then
    overlappingAddresses ← ⋂_{i=1}^n subnet_CIDRs_i
    if overlappingAddresses! = { } then
        error : overlapping addresses
if type ==' L2' then
    main_CIDR ← subnet_CIDRs[0]
    available_ips ← 2^(32−main_CIDR.hosts_bits) − 3
    total_sites ← len(resources)
    distributed_ranges ← divide_ip_range(main_CIDR, available_ips, total_sites)
    local_params ← distributed_ranges[0]
for object in remotes do
    if object! = local then
        interconnections.add(interconnection_create(type, local, object))
id ← GENERATE_ID()
inter−site_service ← new_Service(id, type, local_params, resources, interconnections)
inter − site_service.db.commit()
if type ==' L2' then
    update_local_subnet(local_params)
    update_vms_ips(local)
for object in remotes do
    if object! = local then
        if type ==' L2' then
            params ← distributed_ranges[object.index]
        SEND: Horizontal API CREATE inter-site Site_(object.index)
{id : id, type : type, remotes : resources, params : params}
return 201, Created inter − site services
```

Collecting information from the resources

Per service type sharding strategy

Doing local changes

Send service request to remote modules

Distributed inter-site services module

DISCO & ODL(Federation + NetVirt) inspired

Algorithm 5 Horizontal inter-site networking services CREATE request

Require: $N_i : net\,id\,of\,network_i\,in\,Site_i$

procedure

SEND: Horizontal API CREATE inter-site $\{global_id : id, type : L3, params : params, resources : \{N_i, N_(i+1), N_(i+2), ..., N_{n-1}, N_n\}_{i=1}^{n}\}$ to $Site_2$:

function HORIZONTAL API CREATE INTER-SITE($id, type, params, resources$):

```
    local ← resources.findLocal()
    local_params ← params
    for object in remotes do
        if object! = local then
            interconnections.add(interconnection_create(type, local, object))
    if type ==' L2' then
        update_local_subnet(local_params)
        update_vms_ips(local)
    inter−site_service ← new_Service(id, type, local_params, resources, interconnections)
    inter − site_service.db.commit()
return 201, Created inter − site service
```

Doing local changes

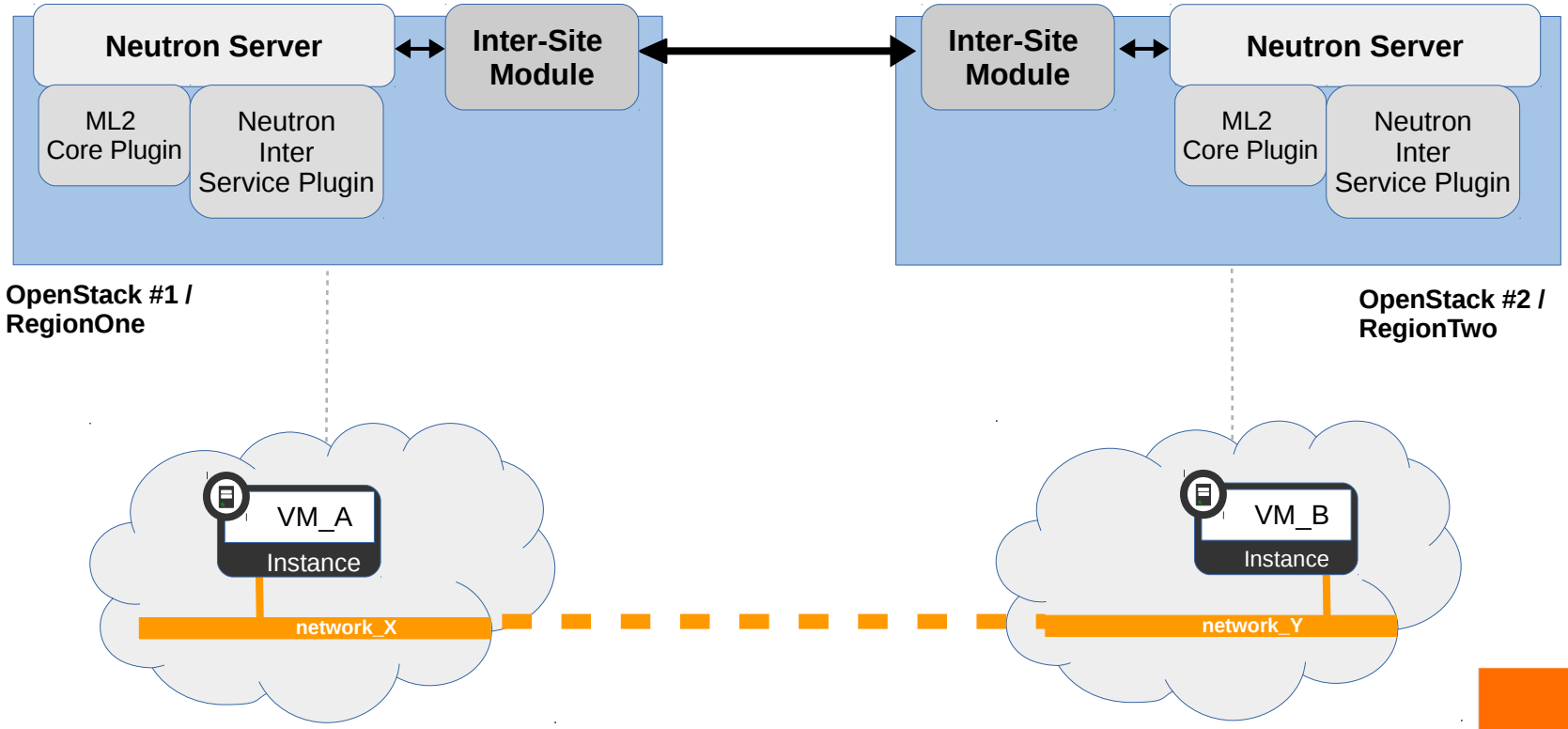
DEMO: Layer 2 network extension

net_x: (Initial)
Allocation-pool: 10.0.0.2-10.0.0.254
VM_A IP: 10.0.0.8

net_x: (Final)
Allocation-pool: 10.0.0.3-10.0.0.128
VM_A IP: 10.0.0.8

net_y: (Initial)
Allocation-pool: 10.0.0.2-10.0.0.254
VM_B IP: 10.0.0.10

net_y: (Final)
Allocation-pool: 10.0.0.129-10.0.0.254
VM_B IP: 10.0.0.135



Next steps & open questions

- Improve the code for the create and update requests
- Add support for networking disconnections
- Test the performance (Grid’5000)
- (For the L2 strategy) Until which point can we shrink the allocation pool to add new sites to a service?
- (For the L2 strategy) CIDR division or global IP sharing?

Thanks !

References

Arora, S. (**2013**). *NFV Orchestration using OpenStack*, . Retrieved from http://dspace.library.uvic.ca/bitstream/handle/1828/8467/Arora_Simar_MSc_2017.pdf?sequence=3&isAllowed=y.
Berde, P.; Gerola, M.; Hart, J.; Higuchi, Y.; Kobayashi, M.; Koide, T.; Lantz, B.; O'Connor, B.; Radoslavov, P.; Snow, W. and Parulkar, G. (**2014**). *ONOS: Towards an Open, Distributed SDN OS*, HotSDN '14 Retrieved from http://zoo.cs.yale.edu/classes/cs434/cs434-2017-spring/readings/onos-hotsdn2014.pdf.
Blial, O.; Mamoun, M. B. and Benaini, R. (**2016**). *An Overview on SDN Architectures with Multiple Controllers*, Hindawi Retrieved from .
Bousselmi, A.; Peltier, J. F. and Chari, A. (**2016**). *Towards a Massively Distributed IaaS Operating System: Composition and Evaluation of OpenStack*, IEEE Conference on Standards for Communications and Networking Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7785190.
Chari, A.; Eleouet, F. and Rohon, M. (**2014**). *Exploring SDN control solutions for in-network distributed IaaS services.*
Chen, S.-C. and Hwang, R.-H. (**2016**). *A Scalable Integrated SDN and OpenStack Management System*, IEEE International Conference on Computer and Information Technology.
Denton, J.PACKT (Ed.), **2014**. *Learning OpenStack Networking (Neutron)*. PACKT.
Dixit, A.; Hao, F.; Mukherjee, S.; Lakshman and t null, R. K. (**2013**). *Towards an Elastic Distributed SDN Controller*, HotSDN Retrieved from http://delivery.acm.org/10.1145/2500000/2491193/p7-dixit.pdf?ip=161.106.4.5&id=2491193&acc=PUBLIC&key=3D7371D543F45B67%2E3D7371D543F45B67%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&__acm__=1519033673_cf005f7343070f5a54868aa92288601c.
Foundation (**2017a**). *OpenStack Docs*, . Retrieved from https://docs.openstack.org/ocata/fr/install-guide-ubuntu/overview.html.
Foundation, O. (**2015a**). *DragonFlow : Distributed implementation of Neutron within a large DC*, . Retrieved from https://wiki.openstack.org/wiki/Dragonflow.
Foundation, O. (**2015b**). *Massively Distributed RPCs in OpenStack*, . Retrieved from https://docs.openstack.org/performance-docs/latest/test_plans/massively_distribute_rpc/plan.html.
Foundation, O. (**2016a**). *Accelerating NFV Delivery with OpenStack*, . Retrieved from https://www.openstack.org/telecoms-and-nfv/.
Foundation, O. (**2016b**). *Fog Edge Massively Distributed Clouds*, . Retrieved from https://wiki.openstack.org/wiki/Fog_Edge_Massively_Distributed_Clouds.
Foundation, O. (**2017b**). *Medium Edge deployment scenario - Gap Analysis*, . Retrieved from https://etherpad.openstack.org/p/edge-gap-analysis.
Fu, Y.; Bi, J.; Gao, K.; Chen, Z.; Wu, J. and Hao, B. (**2014**). *Orion: A Hybrid Hierarchical Control Plane of Software-Defined Networking for Large-Scale Networks*, IEEE 22nd International Conference on Network Protocols Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6980430.
Gampel, E. (**2015**). *Centralized vs. Distributed SDN Control Path Paradigm*, . Retrieved from http://blog.gampel.net/2015/08/centralized-vs-distributed-sdn-control.html.
Hoai, L. (**2017**). *OpenStack and Software-Defined Networking The Enormous Potential of Open Source Software Collaboration*, . Retrieved from https://www.theseus.fi/bitstream/handle/10024/133923/Hoai_Le.pdf?sequence=1&isAllowed=y.
Hu, J.; Lin, C.; Li, X. and Huang, J. (**2014**). *Scalability of Control Planes for Software Defined Networks:Modeling and Evaluation*, IEEE Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6914314.
Juniper (**2015**). *Contrail Architecture*, . Retrieved from https://www.juniper.net/assets/uk/en/local/pdf/whitepapers/2000535-en.pdf.
Karakus, M. and Durresi, A. (**2016**). *A survey: Control plane scalability issues and approaches in Software-Defined Networking (SDN)*, ComNet .
Koponen, T.; Casado, M.; Gude, N.; Stribling, J.; Poutievski, L.; Zhu, M.; Ramanathan, R.; Iwata, Y.; Inoue, H.; Hama, T. and Shenker, S. (**2012**). *Onix: A Distributed Control Platform for Large-scale Production Networks*, OSDI Retrieved from http://static.usenix.org/events/osdi10/tech/full_papers/Koponen.pdf.
Lebre, A. and Pastor, J. (**2015**). *The DISCOVERY Initiative - Overcoming Major Limitations of Traditional Server-Centric Clouds by Operating Massively Distributed IaaS Facilities*, . Retrieved from https://hal.inria.fr/hal-01203648v2/document.
Li, Y. and Chen, M. (**2015**). *Software-Defined Network Function Virtualization: A Survey*, IEEE access Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7350211.
Marconett, D. and Yoo, S. (**2015**). *FlowBroker: A Software-Defined Network Controller Architecture for Multi-Domain Brokering and Reputation*, Journal of Network System Management.
Medeiros, B.; Jr., M. S.; Melo, T.; Torrez, M.; Redígolo, F.; Rodrigues, E. and Cristofoleti, D.33rd Brazilian Symposium on Computer Networks & Systems, D. (Ed.), **2015**. *Applying Software-defined Networks to Cloud Computing*. 33rd Brazilian Symposium on Computer Networks and Distributed Systems, . Retrieved from http://sbrc2015.ufes.br/wp-content/uploads/Ch1.pdf.
Phemius, K.; Bouet, M. and Leguay, J. (**2014**). *DISCO: Distributed Multi-domain SDN Controllers*, Network Operations and Management Symposium Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6838330.
Santos, M.; Nunes, B.; Obraczka, K. and Turletti, T. (**2014**). *Decentralizing SDN's Control Plane*, IEEE Conference on Local Computer Networks Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6925802.
Subramanian, S. and Voruganti, S.Packt (Ed.), **2016**. *Software-Defined Networking (SDN) with OpenStack*. Packt.
Taleb, T.; Samdanis, K.; Mada, B.; Flinck, H.; Dutta, S. and Sabella, D. (**2017**). *On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration*, IEEE COMMUNICATIONS SURVEYS & TUTORIALS : 25 Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7931566.
Tootoonchian, A. and Ganjali, Y. (**2010**). *HyperFlow: A Distributed Control Plane for OpenFlow*, IEEE Proceedings of the 2010 internet network management conference on Research on enterprise networkin Retrieved from https://www.usenix.org/legacy/event/inmwren10/tech/full_papers/Tootoonchian.pdf.
Yang, H.; Ivey, J. and Riley, G. F. (**2017**). *Scalability Comparison of SDN Control Plane Architectures Based on Simulations*, International Performance Computing and Communications Conference Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8280465.
Zhang, T.; Bianco, A.; De-Domenico, S. and Giaccone, P. (**2016**). *The Role of Inter-Controller Traffic in the Placement of Distributed SDN Controllers*, IEEE Conference on Network Function Virtualization and Software Defined Networks Retrieved from https://arxiv.org/pdf/1605.09268.pdf.