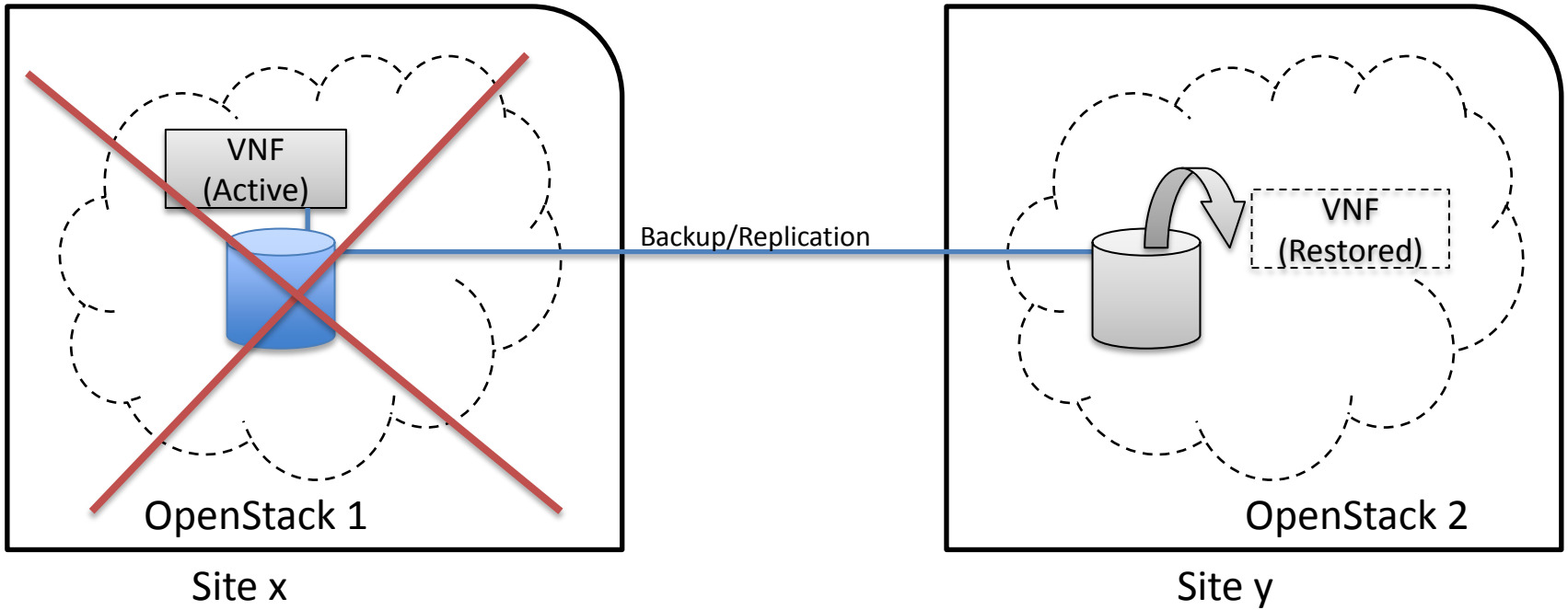
Multisite BP-Bug review

Use case 3: VNF Geo_site_redundancy

a VNF (telecom application) should, be able to restore in another site for catastrophic failures happened.



Save VNF if catastrophic failures like flood, earthquake, power-off happened

Use case 3: VNF Geo_site_redundancy

a VNF (telecom application) should, be able to restore in another site for catastrophic failures happened.

1. Nova Quiesce + Cinder Consistency volume snapshot+ Cinder backup

GR software get the attached volumes for the VMs in a VNF from Nova
GR software add these attached volumes to the consistency group in Cinder (NOTE: make sure that the block storage driver supports CG technology)
GR software call Nova API Quiesce to freeze VM and flush buffer
GR software make cgsnapshots of these volumes in Cinder
GR software create volumes from the cgsnapshots in Cinder
GR software create backup (incremental) for these volumes to backup storage (swift or ceph, or..) in Cinder
if this site failed,
GR software restore these backup volumes to Cinder in the backup site.
GR software boot vm from bootable volume rom Cinder in the backup site and attach the data volumes.

Pros: 1) atomic quiesce / unquiesce api from Nova, make transactional snapshot of a group of VMs is possible, for example, quiesce VM1, quiesce VM2, quiesce VM3, snapshot VM1's volumes, snapshot VM2's volumes, snapshot VM3's volumes, unquiesce VM3, unquiesce VM2, unquiesce VM1. For some telecom application, the order is very important for a group of VMs with strong relationship. 2) leverage the Cinder consistency group functionality.
Cons: Need Nova to expose the quiesce / unquiesce, fortunately it's alreay there in Nova-compute, just to add API layer to expose the functionality.

Requirement to OpenStack: Nova needs to expose quiesce / unquiesce api, which is lack in Nova now.
BP registered: https://blueprints.launchpad.net/nova/+spec/expose-quiesce-unquiesce-api

Use case 3: VNF Geo_site_redundancy

a VNF (telecom application) should, be able to restore in another site for catastrophic failures happened.

2. Cinder Volume Replication

GR software create volume with replication enabled volume type
Cinder volume driver create volume with replication in backend storage
Cinder volume driver record the reference of block device in the remote site storage into the Cinder volume meta data
GR software get the reference of block device in the remote site storage
if this site failed,
GR software manage the volume from the reference of the block device in the backup site.
GR software boot vm from bootabl volume rom Cinder in the backup site and attach the data volumes.

Pros: 1) Replication will be done in the storage level automaticly, no need to create backup regularly, for example, daily.
Cons: 1) No consistency guarantee. Application should be aware of the replication and guarantee the consistency 2) Only few storage backend support to expose reference of the block device.

Requirement to OpenStack: save the real ref to volume admin_metadata after it has been managed by the driver
BP registered: https://review.openstack.org/#/c/182150/.

Use case 3: VNF Geo_site_redundancy

a VNF (telecom application) should, be able to restore in another site for catastrophic failures happened.

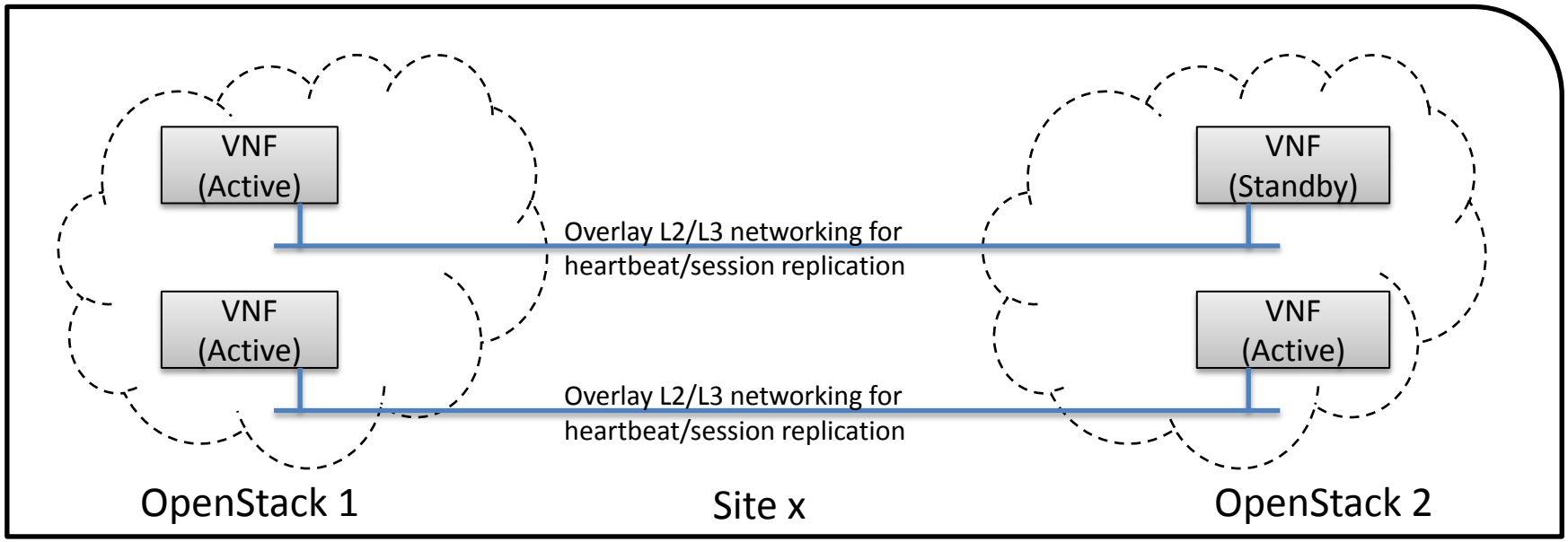
3. Nova Snapshot + Glance Image + Cinder Snapshot + Cinder Backup

GR software create VM snapshot in Nova
Nova quiece the VM internally
Nova create image in Glance
Nova create a snapshot of the VM, including volumes
If the VM is volume backed VM, then create volume snapshot in Cinder
No image uploaded to glance, but add the snapshot in the meta data of the image in Glance
GR software to get the snapshot information from the Glance
GR software create volumes from these snapshots
GR software create backup (incremental) for these volumes to backup storage (swift or ceph, or..) in Cinder
if this site failed,
GR software restore these backup volumes to Cinder in the backup site.
GR software boot vm from bootabl volume rom Cinder in the backup site and attach the data volumes.

Pros: 1) Automaticly quiesce/unquiesce, and snapshot of volumes of one VM. It's suitable for single VM backup/restore
Cons: 1) Impossible to form a transactional group of VMs backup. for example, quiesce VM1, quiesce VM2, quiesce VM3, snapshot VM1, snapshot VM2, snapshot VM3, unquiesce VM3, unquiesce VM2, unquiesce VM1. This is quite important in telecom application in some scenario 2) not leverage the Cinder consistency group. 3) One more service Glance involved in the backup. Not only to manage the increased snapshot in Cinder, but also need to manage the regarding tempary image in Glance.

Requirement to OpenStack: None.

Use case 2: VNF high availability across VIM



Use case 2: VNF high availability across VIM

Overlay L2/L3 networking cross Neutron for heartbeat/session replication

Requirement to OpenStack:

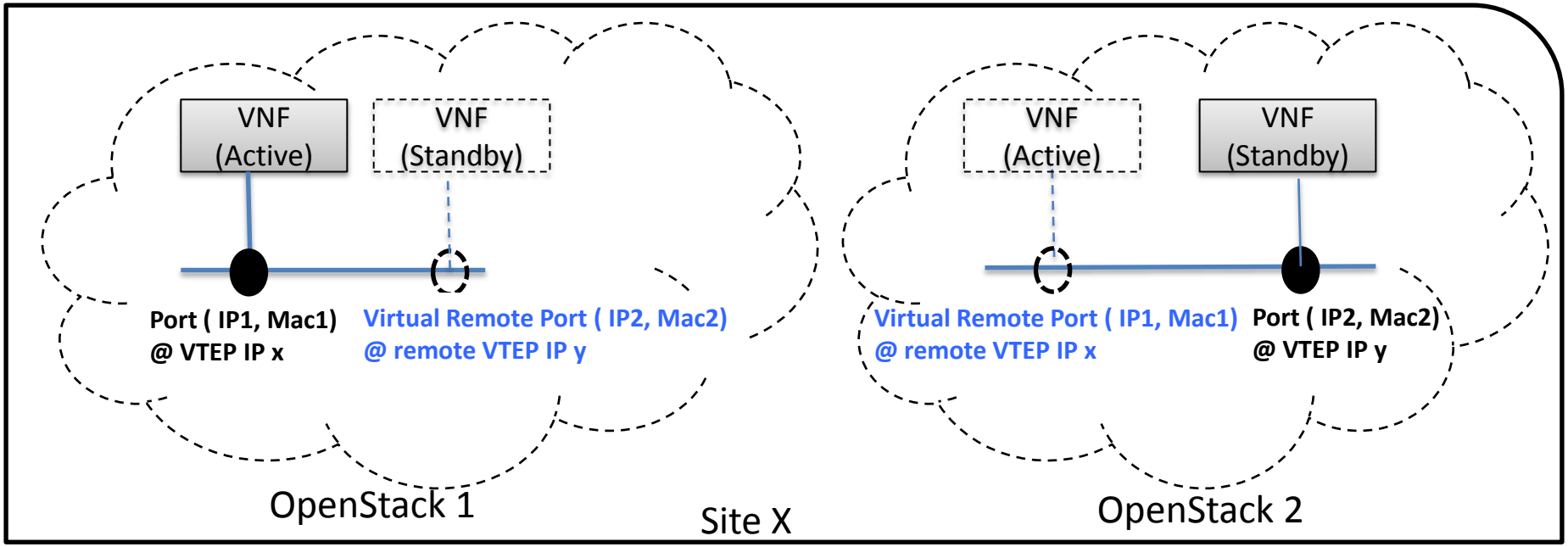
1) Overlay L2 networks or shared L2 provider networks, for cross OpenStack instance networking for heartbeat or state replication. Overlay L2 network is preferred:

- Legacy compatibility: Some telecom app with built-in internal L2 network, for easy to move these app to VNF, it would be better to provide L2 network
- IP overlapping: multiple VNFs may have overlapping IP address for cross OpenStack instance networking

2) L3 networking cross OpenStack instance for heartbeat or state replication

3) The IP address used for VNF to connect with other VNFs should be able to be floating cross OpenStack instance. For example, if the master failed, the IP address should be used in the standby which is running in another OpenStack instance.

Use Case 2: Virtual remote port for overlay L2 networking



OpenStack1:

1. Create Network (net)
2. Boot VM1 in net -> Port (IP 1, Mac 1) @ VTEP IP x
3. Create virtual remote port for VM2 in net (IP 2, Mac2 , VTEP = remote VTEP IP y).
4. L2population inside neutron

OpenStack2:

1. Create Network (net)
2. Boot VM 2 in net -> Port (IP 2, Mac 2) @ VTEP IP y
3. Create virtual remote port for VM1 in net (IP 1, Mac1 , VTEP = remote VTEP IP x)
4. L2population inside neutron

Use case 2: BP to Neutron

1. ML2: Port Cross Backbends Extension

   https://review.openstack.org/#/c/215409/2/specs/liberty/cross-backends-extension.rst

   The BP is to make port not binding to Host IP as VTEP IP, but using tunneling_ip as the VTEP (can be host IP or any other tunneling endpoint IP, for example, external Host IP or L2GW IP

   - Extend port with tunneling_ip attribute
   - Enhance L2Population support tunneling_ip

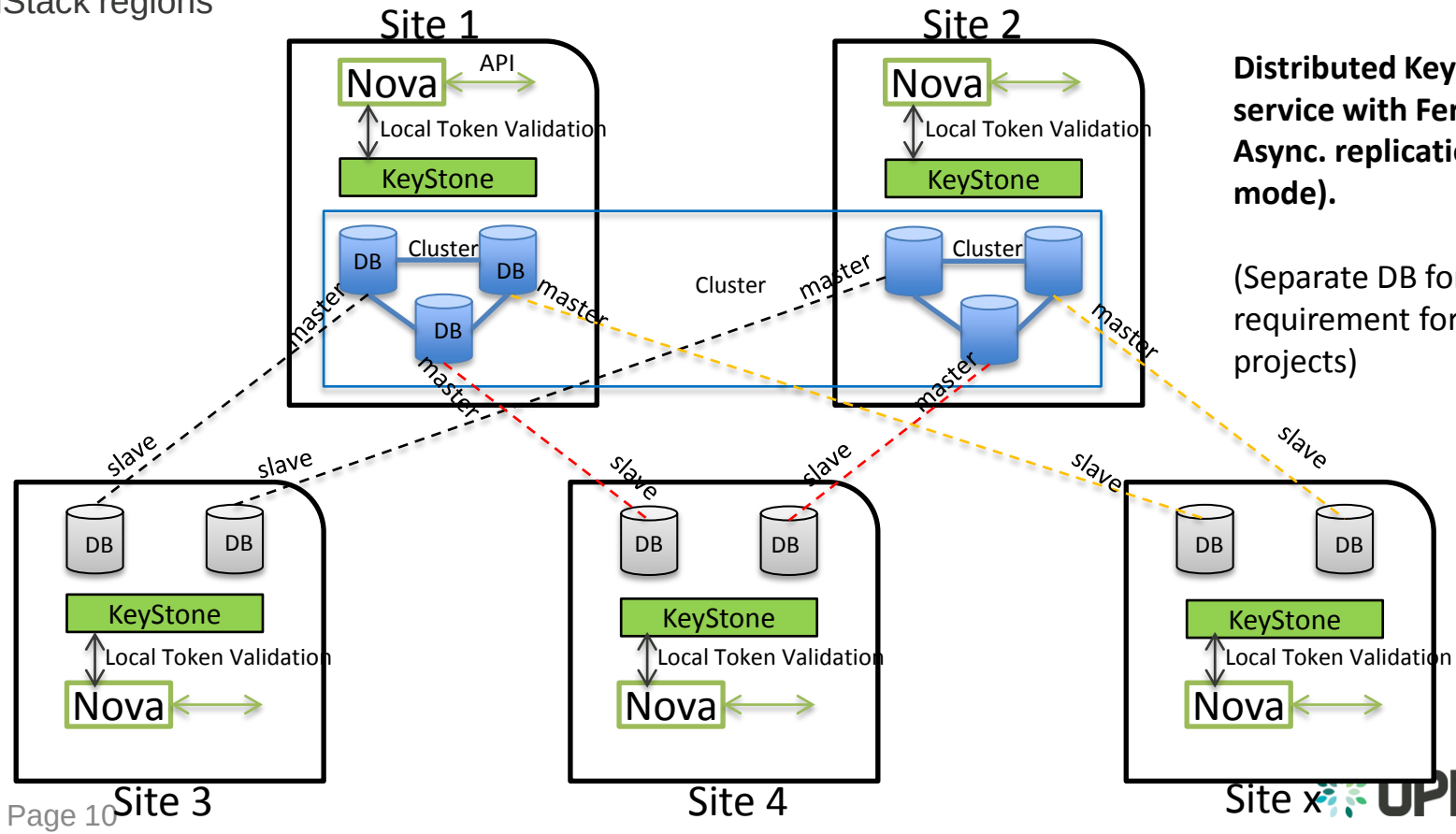
2. Remote port for L2 communicate

   https://bugs.launchpad.net/neutron/+bug/1484005

   This BP is to add a new ML2 mechanism driver in Neutron to handle the virtual remote port, and activate L2population inside Neutron.

Use case 1: multisite identity service management

a user should, using a single authentication point be able to manage virtual resources spread over multiple OpenStack regions



Distributed KeyStone service with Fernet token + Async. replication (star-mode).

(Separate DB for KeyStone requirement for installation projects)

Use case 1: multisite identity service management

a user should, using a single authentication point be able to manage virtual resources spread over multiple OpenStack regions

Proposed solution:

KeyStone service(Distributed) with Fernet token + Async replication (star-mode).

one master KeyStone cluster with Fernet token in two sites (for site level high availability purpose), other sites will be installed with at least 2 slave nodes where the node is configured with DB async replication from the master cluster members, and one slave's mater node in site1, another slave's master node in site 2.

Only the master cluster nodes are allowed to write, other slave nodes waiting for replication from the master cluster (very little delay) member. But the chanllenge of key distribution and rotation for Fernet token should be settled, you can refer to these two blogs: http://lbragstad.com/?p=133, http://lbragstad.com/?p=156

Pros.

Why cluster in the master sites? There are lots of master nodes in the cluster, in order to provide more slaves could be done async. replication in parallel.

Why two sites for the master cluster? to provide higher reliability (site level) for writing request.

Why using multi-slaves in other sites. Slave has no knowledge of other slaves, so easy to manage multi-slaves in one site than a cluster, and multi-slaves work independently but provide multi-instance redundancy(like a cluster, but independent).

Cons. The distribution/rotation of key management.

Use case 1: multisite identity service management

BUG:

Can't specify identity endpoint for token validation among several keystone servers in keystonemiddleware: https://bugs.launchpad.net/keystone/+bug/1488347. The method we provide works.

After discussion, community proposed using region-name to specify the keystone server for token validation. https://review.openstack.org/#/c/216579/. Not succeed in verification.

Use case 4: Centralized service for resources management and/or replication (sync tenant resources like images, ssh-keys, security groups, etc)

Image replication: the simplest way:

Using Glance V1 interface copy-from to call another Glance V2 interface (image download), that means the copy-from data is coming from download interface from another Glance.

https://review.openstack.org/#/c/210231/

Pros. minimum modification. Just let the copy-from with the token to the image source service

Cons. 1)Metadata not copied at the same time 2) sync. Operation. 3) no progress report

Use case 4: Centralized service for resources management and/or replication (sync tenant resources like images, ssh-keys, security groups, etc)

Image replication: async. clone task

Image replication across glance (there is already one BP, but need to work on this BP): https://blueprints.launchpad.net/glance/+spec/clone-image-across-regions

Thank you