Integrating GlusterFS as a storage domain in VDSM & supporting storage array offloads from VDSM

Deepak C Shetty – deepakcs@linux.vnet.ibm.com

(IBM Linux Technology Center, Bangalore, India)

Nov. 2012

oVirt workshop, Barcelona

Part 1: Integrating GlusterFS as a storage domain in VDSM

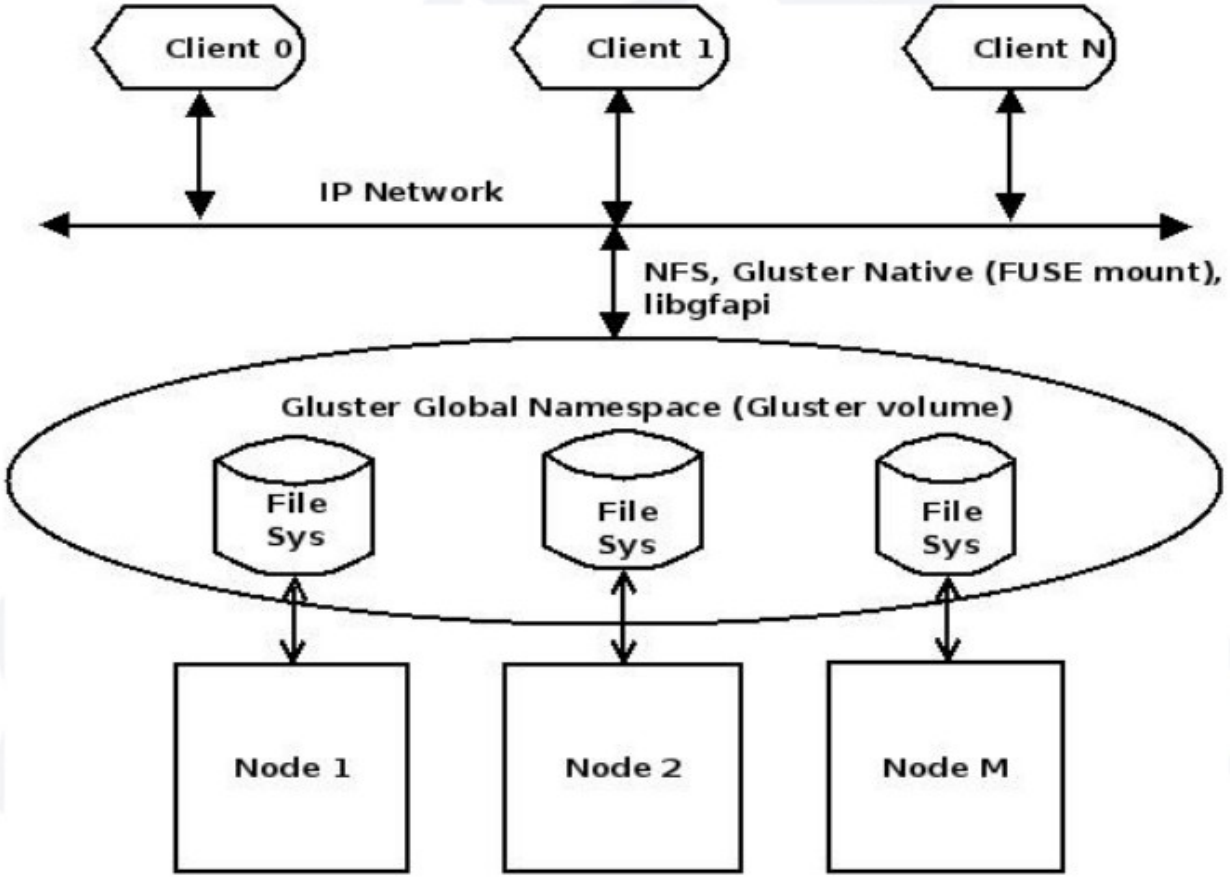
Agenda

- What is GlusterFS
- Enabling GlusterFS for virtualization use
- oVirt/VDSM architecture
- VDSM storage concepts
- GlusterFS as a VDSM Storage Domain
  - Different approaches
- GlusterFS domain support in oVirt engine
- Future work
- References

GlusterFS

What is GlusterFS

- User-space distributed file system
- Capable of scaling to several peta-bytes
- Aggregates storage resources from multiple nodes and presents a unified file system namespace
- Storage resources (aka bricks) can be made of any commodity hardware, for eg: x86-64 server(s)



Source: LPC 2012

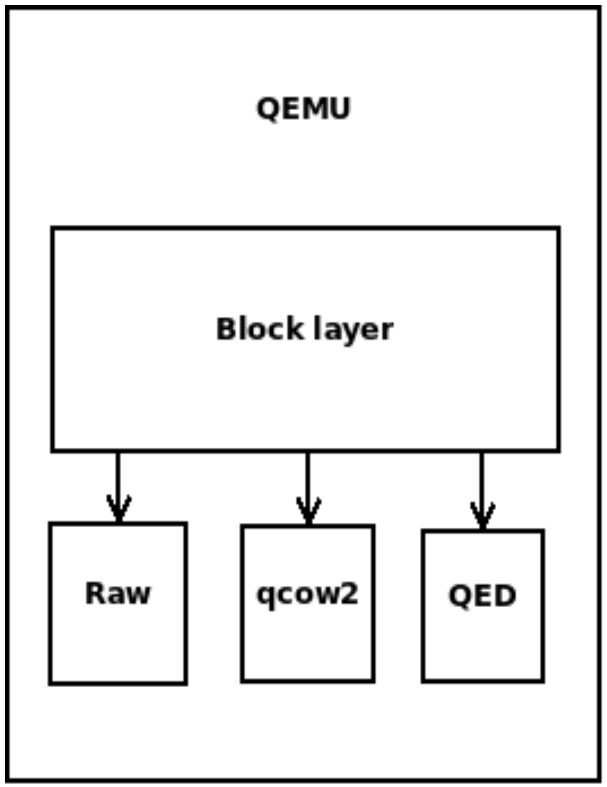
GlusterFS - features

- Replication
  - HA
- Striping
  - Store large files across multiple storage servers
- Distribution
  - Distribute files evenly across storage servers
- Combinations of the above
- Geo-replication/sync
  - Disaster recovery
- No metadata
  - Elastic hash algorithm (on-the-fly)
  - Eliminates need for metadata lookup
- Online addition and removal of nodes
  - Volume expansion and shrinking
- Stackable user space design
  - Translators

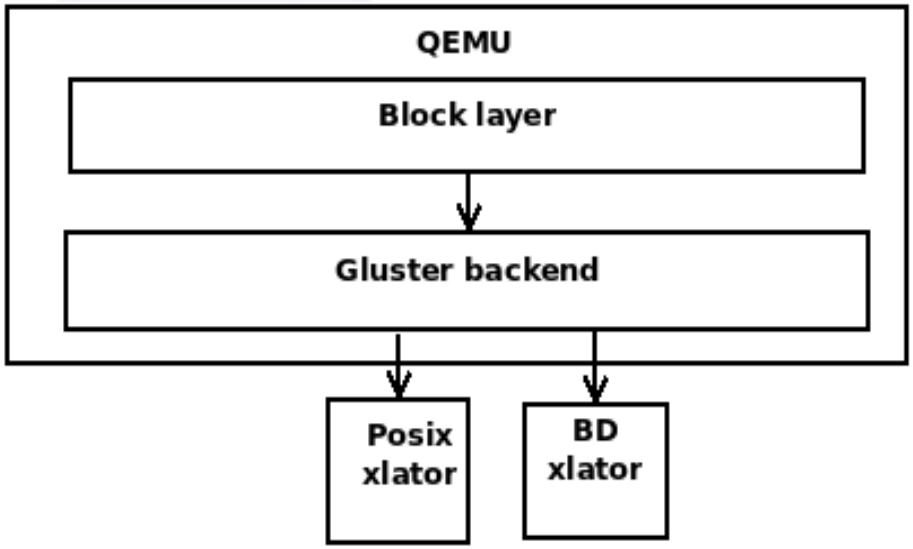
Enabling Gluster for virtualization use

QEMU – GlusterFS Integration

Before



After



- Before
  - **`-drive file=<path/to/gluster/mount>/<path/to/image>`**
  - Maps to `<disk type=file..>...</disk>` in libvirt xml
  - FUSE overhead
- After
  - **`-drive file=gluster[+transport]://[server[:port]]/volname/image[?socket=...]`**
  - Maps to `<disk type=network..>...</disk>` in libvirt xml
  - No FUSE overhead

Source: LPC 2012

Enabling GlusterFS for virtualization use

Heads up: A **new VDSM storage domain** is required to exploit this

- QEMU-GlusterFS integration
  - Native integration, no FUSE mount
  - Gluster as QEMU block back end
  - QEMU talks to gluster and hides different file systems and storage types underneath
  - Available in QEMU upstream
- Single solution for local, SAN and NAS
  - GlusterFS fits well into local and NAS scenarios
  - Block device support in GlusterFS via BD xlator (SAN)
- Making GlusterFS virt-ready
  - Gluster CLI enhancements for snapshots and clones

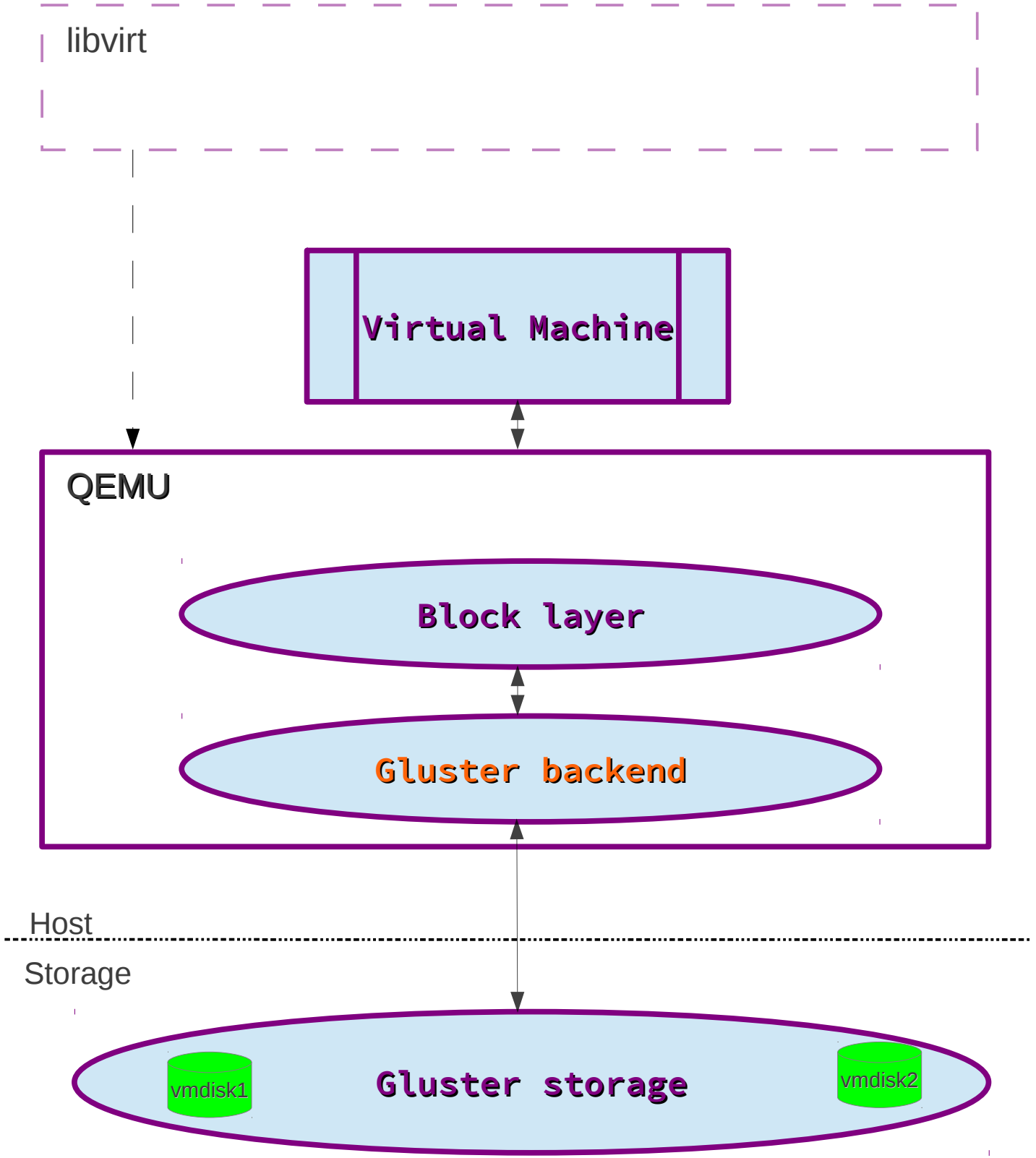
Source: LPC 2012

QEMU gluster specification

- **gluster[+transport]://[server[:port]]/volname/image[?socket=...]**
  - 'gluster' is the protocol
  - 'transport' specifies the transport type used to connect to gluster management daemon (glusterd). Valid transport types are tcp, unix and rdma. If a transport type isn't specified, then tcp type is assumed
  - 'server' specifies the server where the volume file specification for the given volume resides. This can be either hostname, ipv4 address or ipv6 address. ipv6 address needs to be within square brackets [ ]
  - If transport type is 'unix', then 'server' field should not be specifed. Instead the 'socket' field needs to be populated with the path to unix domain socket
  - 'port' is the port number on which glusterd is listening. This is optional and if not specified, QEMU will send 0 which will make gluster to use the default port. If the transport type is unix, then 'port' should not be specified
  - 'volname' is the name of the gluster volume which contains the VM image
  - 'image' is the path to the actual VM image that resides on gluster volume

QEMU gluster specification (contd.)

- **Examples**
  - gluster://1.2.3.4/testvol/a.img
  - gluster+tcp://1.2.3.4/testvol/a.img
  - gluster+tcp://1.2.3.4:24007/testvol/dir/a.img
  - gluster+tcp://[1:2:3:4:5:6:7:8]/testvol/dir/a.img
  - gluster+tcp://[1:2:3:4:5:6:7:8]:24007/testvol/dir/a.img
  - gluster+tcp://server.domain.com:24007/testvol/dir/a.img
  - gluster+unix:///testvol/dir/a.img?socket=/tmp/glusterd.socket
  - gluster+rdma://1.2.3.4:24007/testvol/a.img


libvirt
Virtual Machine
QEMU
Block layer
Gluster backend
Host
Storage
vmdisk1
Gluster storage
vmdisk2

GlusterFS support in libvirt

Heads up: VDSM uses libvirt to talk to QEMU-KVM in oVirt stack

- libvirt
  - A toolkit to interact with the virtualization capabilities Linux
  - Abstraction layer above Hypervisors
  - Helps define & manage Virtual Machines using XML
- Since QEMU now supports GlusterFS as a storage backend, libvirt can enable the upper layer of virtualization management software to make use of this feature
- libvirt support for GlusterFS (WIP)
  - Gluster fits as a new network block device

GlusterFS support in libvirt (contd.)

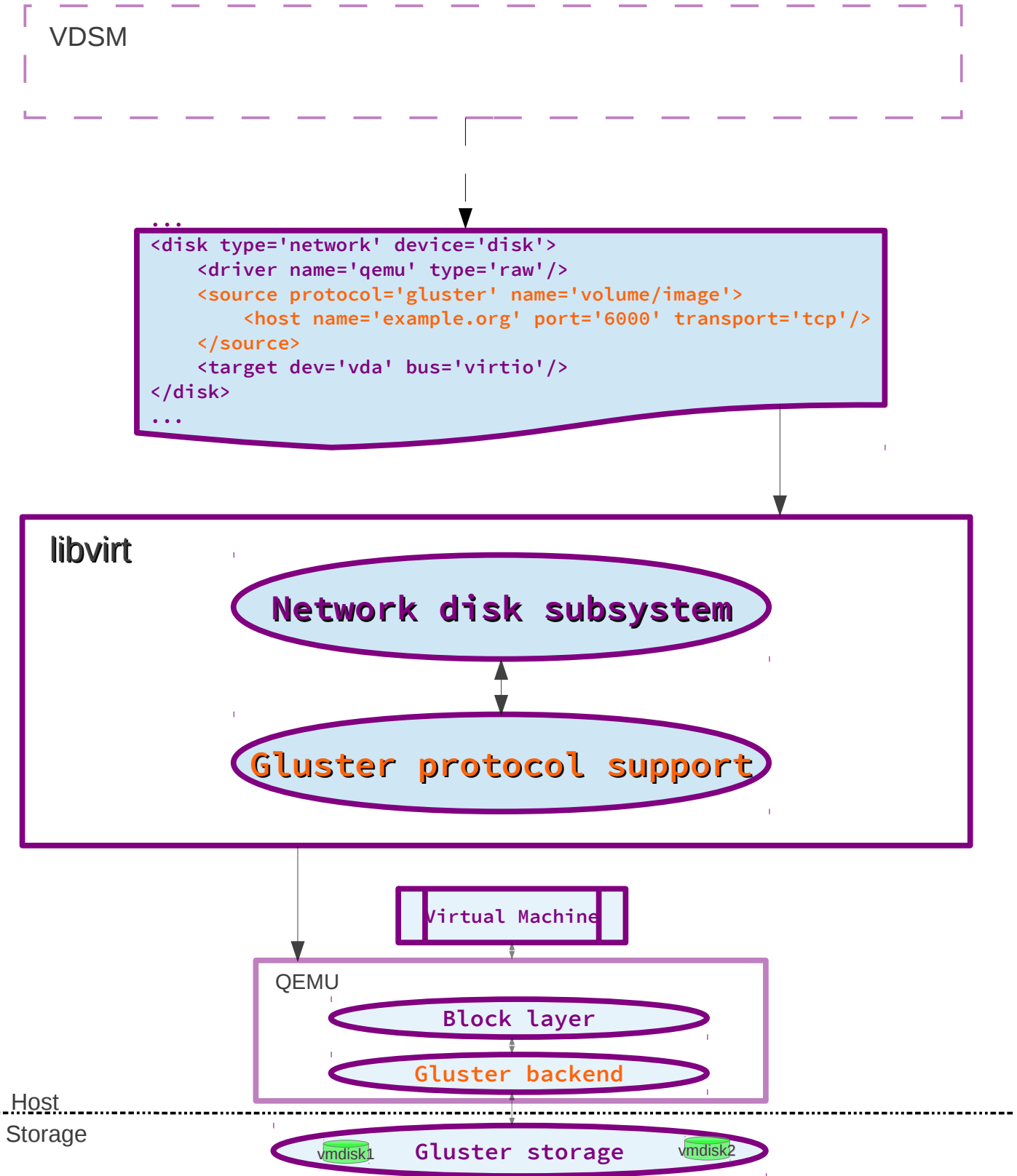
- Proposed libvirt XML format to specify disk images on glusterfs

```
<disk type='network' device='disk'>
  <driver name='qemu' type='raw'/>
  <source protocol='gluster' name='volume/image'>
    <host name='example.org' port='6000' transport='tcp'/>
  </source>
</disk>
```

Note: In the **<host>** element above, **transport** is an optional attribute
Valid **transport** values are **tcp**, **unix** or **rdma**. If none specified, **tcp** is assumed

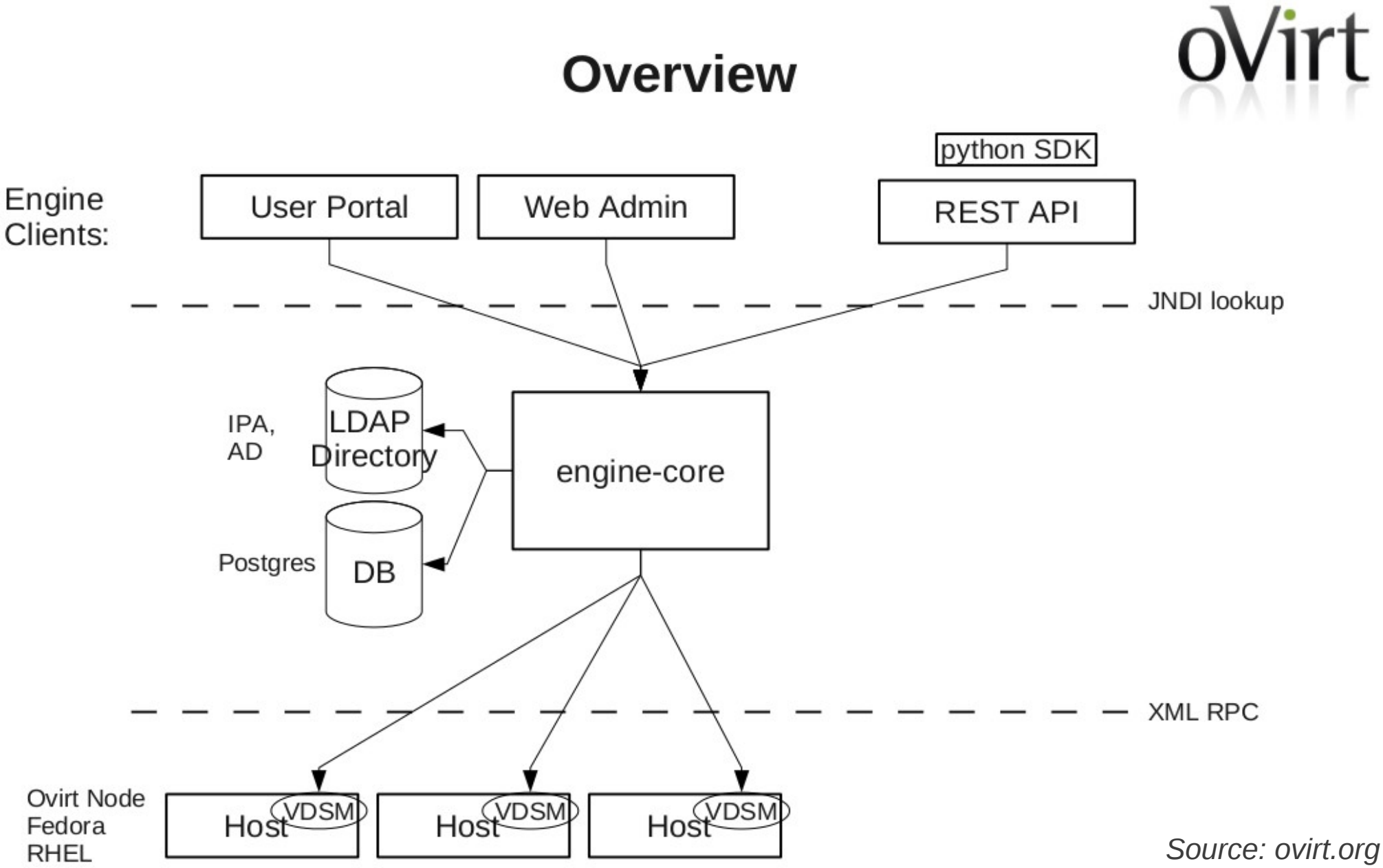
If **transport** type is **unix**, **socket** attribute specifies path to unix socket

```
<disk type='network' device='disk'>
  <driver name='qemu' type='raw'/>
  <source protocol='gluster' name='volume/image'>
    <host name='localhost' port='0' transport='unix' socket='/path/to/sock'/>
  </source>
</disk>
```



VDSM
...
<disk type='network' device='disk'>
<driver name='qemu' type='raw'/>
<source protocol='gluster' name='volume/image'>
<host name='example.org' port='6000' transport='tcp'/>
</source>
<target dev='vda' bus='virtio'/>
</disk>
...
libvirt
Network disk subsystem
Gluster protocol support
Virtual Machine
QEMU
Block layer
Gluster backend
Host
Storage
vmdisk1
Gluster storage
vmdisk2

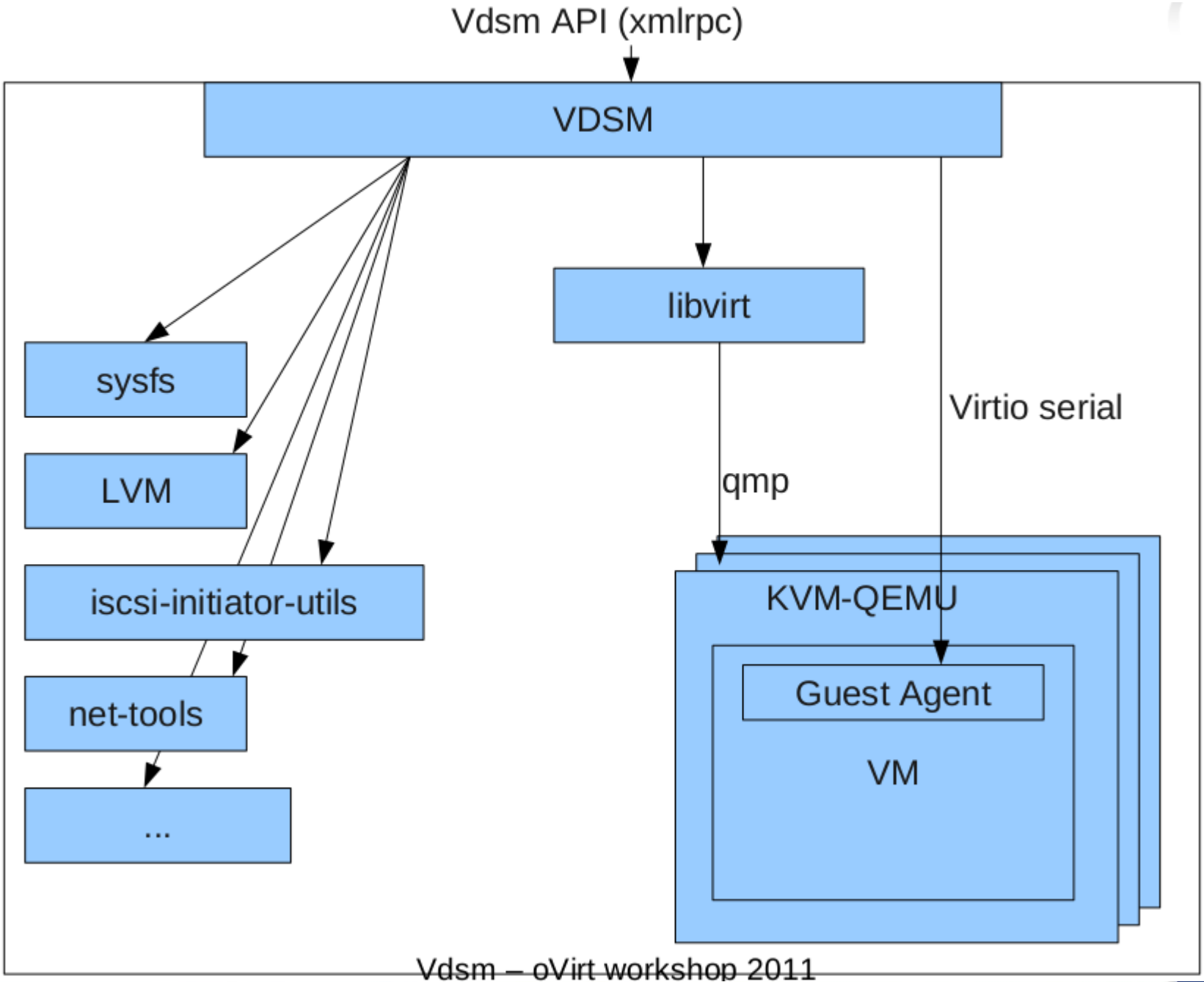
oVirt/VDSM architecture

oVirt architecture



Source: ovirt.org

VDSM architecture



Source: ovirt.org

So what exactly is VDSM?

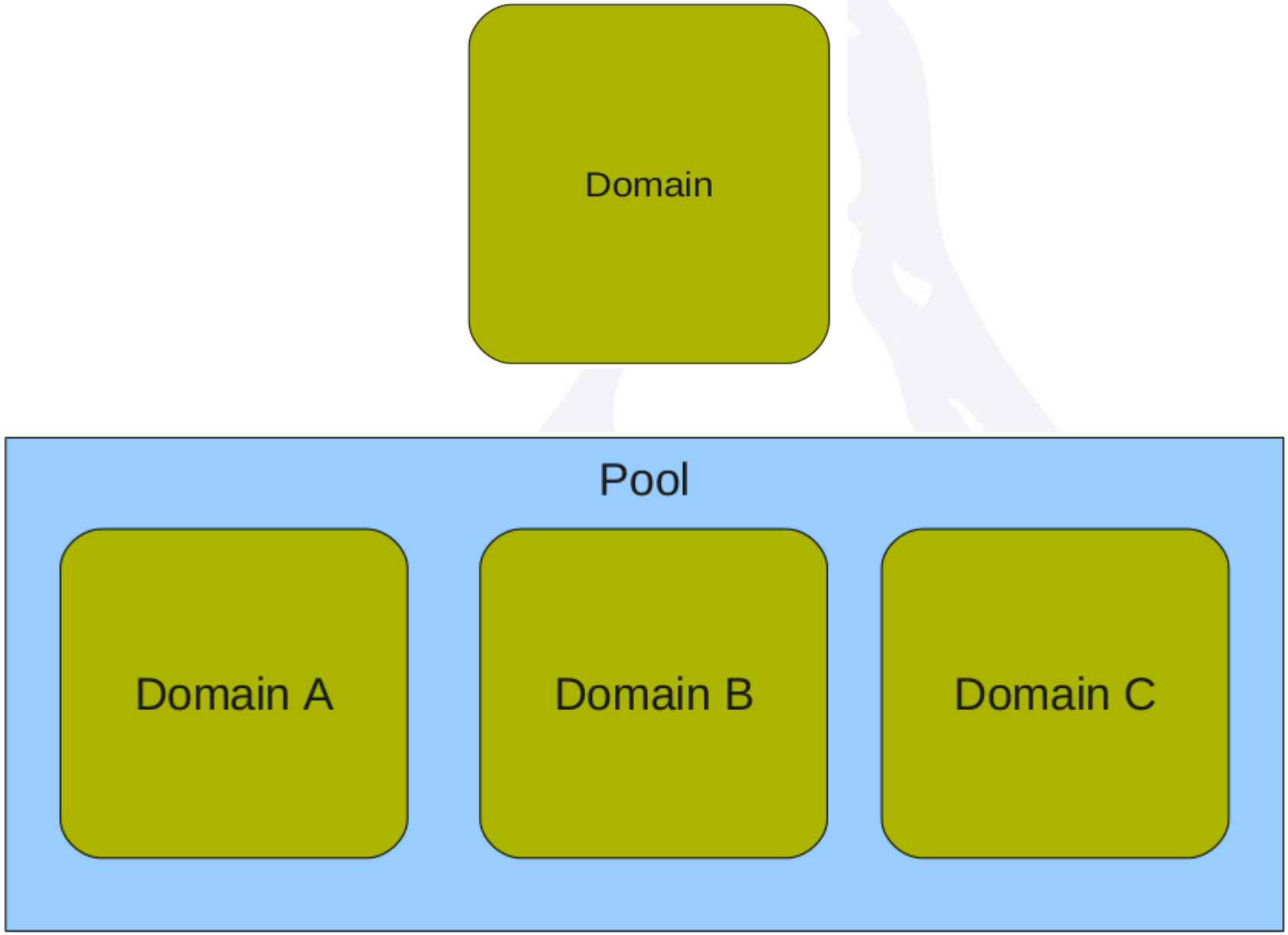
- **V**irtual **D**esktop **S**erver **M**anager
- Host agent for oVirt engine
- Node virtualization management API
- Abstracts low level details of the underlying linux environments
- Exposes high level API consumed by oVirt engine
  - Covers all functionality needed by oVirt engine

And what does VDSM do?

- Host bootstrap and registration
- VM life cycle (via libvirt)
- Guest interaction (sso, stop, etc)
- **Storage management**
- Network configuration
- Monitoring host and VMs
- Policy management
- Fencing proxy

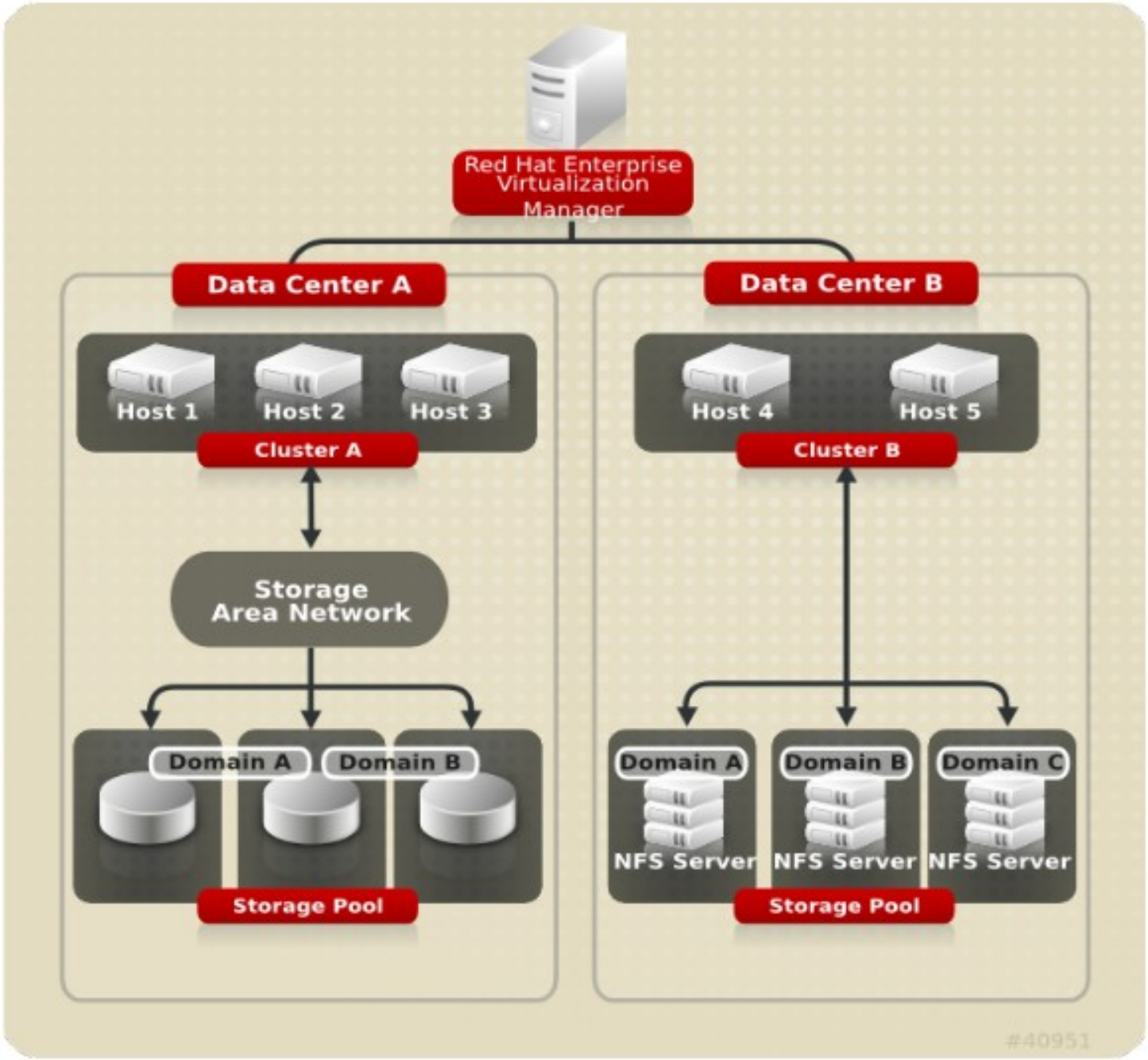
VDSM storage concepts

VDSM storage concepts



Source: ovirt.org

Example



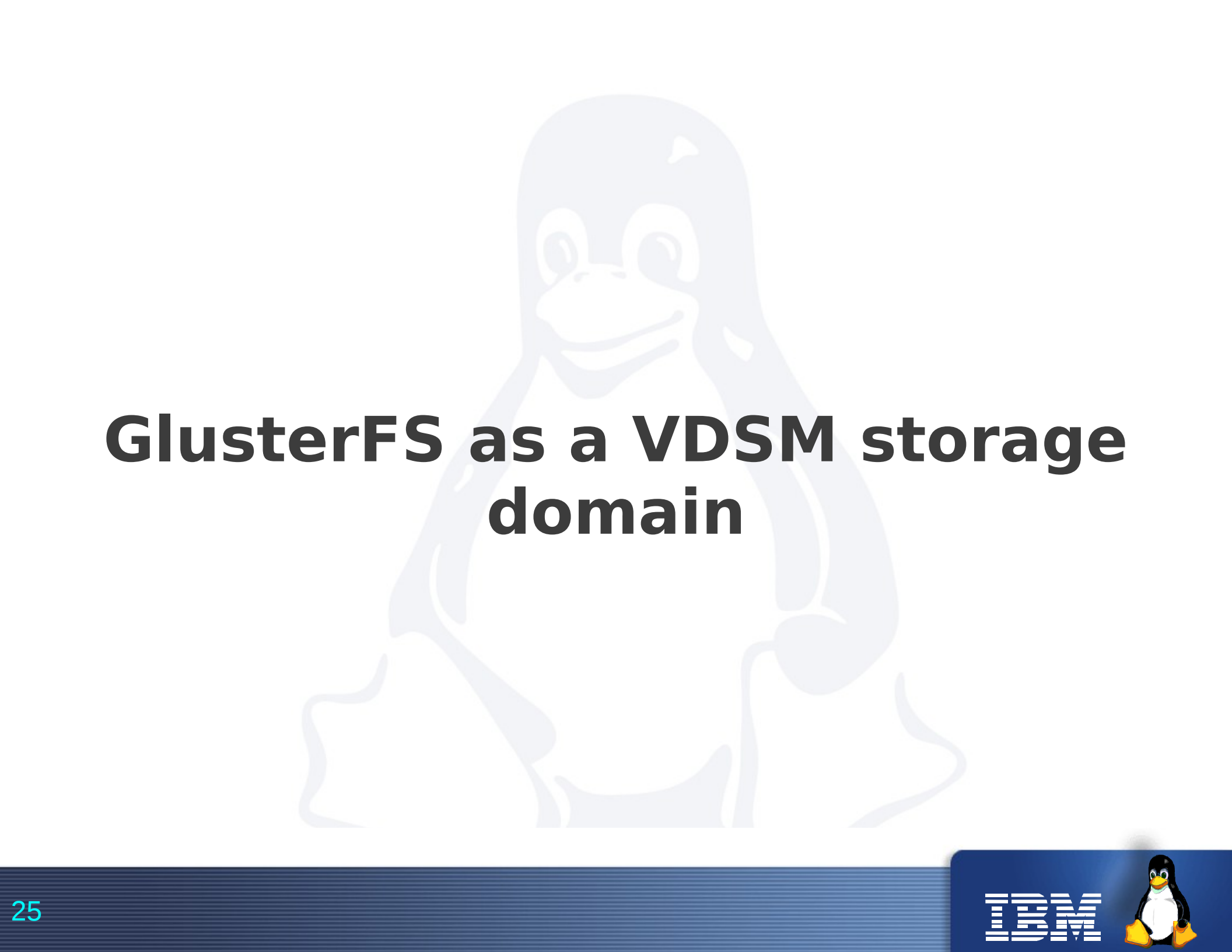
Source: ovirt.org

VDSM storage concepts contd.

- Storage domain
  - Fundamental standalone storage entity
  - It acts as the image repository
- Domain types
  - File based (NFS, PosixFS, local)
  - Block based (iSCSI, FCP)
- Storage pool
  - Group of storage domains
  - Implemented as a managed cluster
  - VM repository that contains meta data about storage domains, storage tasks, VMs, locks, etc

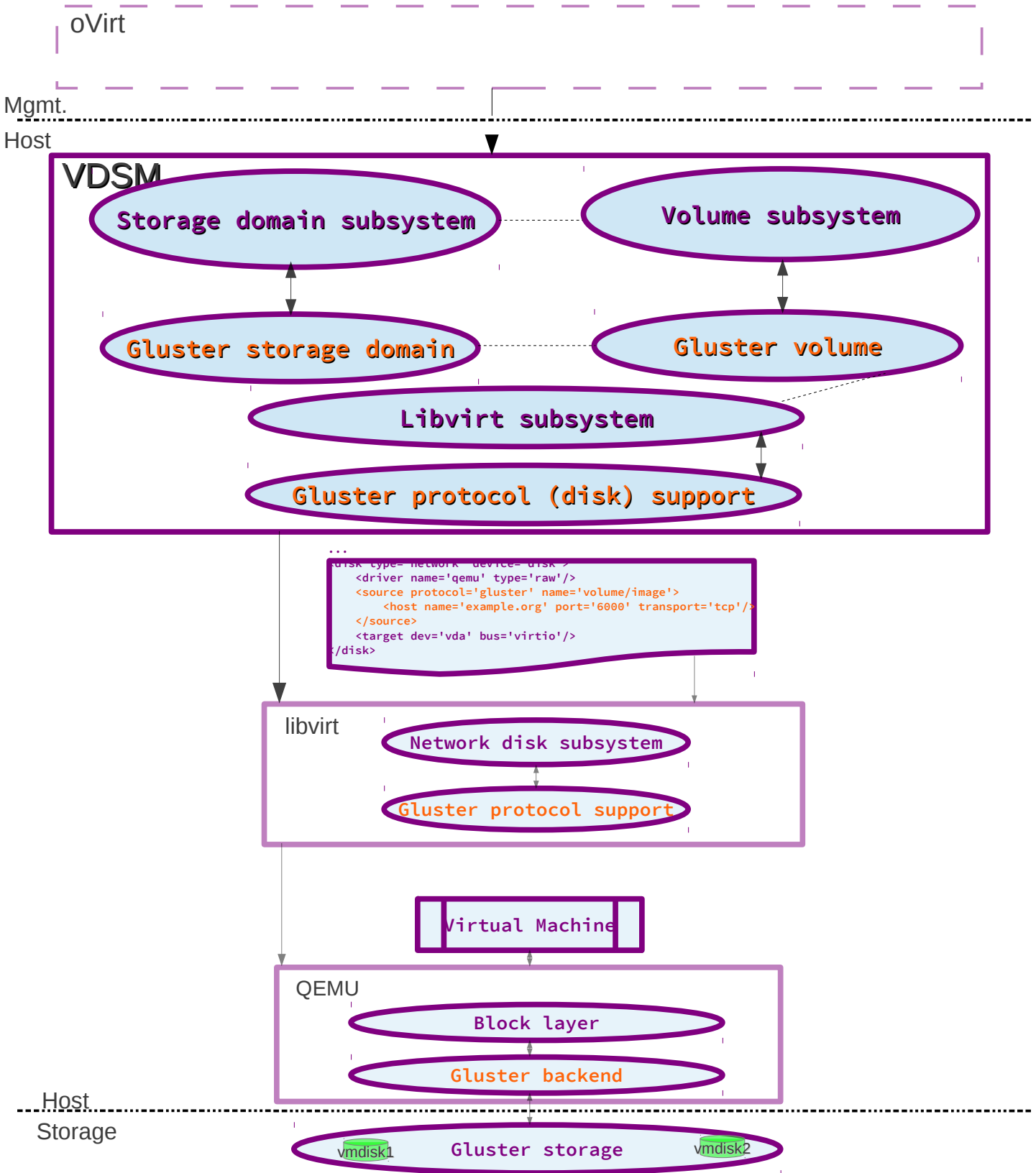
Extended to support GlusterFS

GlusterFS as a VDSM storage domain



GlusterFS as a VDSM Storage Domain – Approaches

- As a PosixFs storage domain
- As a PosixFs storage domain + VDSM hooks
- As a enhanced PosixFs storage domain (with network disk support)
- As a nfsSD
  - Re-uses nfsSD to support GlusterFS as well
- **As a brand new storage domain** (WIP)
  - Patches under review
  - Introduces GlusterStorageDomain class
  - Introduces GlusterVolume class
  - Introduces GlusterFSConnection class
  - Adds support for network disk type in libvirtvm.py


oVirt
Mgmt.
Host
VDSM
Storage domain subsystem
Volume subsystem
Gluster storage domain
Gluster volume
Libvirt subsystem
Gluster protocol (disk) support
...
<driver name='qemu' type='raw'/>
<source protocol='gluster' name='volume/image'>
<host name='example.org' port='6000' transport='tcp'/>
</source>
<target dev='vda' bus='virtio'/>
</disk>
libvirt
Network disk subsystem
Gluster protocol support
Virtual Machine
QEMU
Block layer
Gluster backend
Host
Storage
vmdisk1
Gluster storage
vmdisk2

GlusterFS as a new VDSM Storage Domain - Details

- GLUSTERFS_DOMAIN as a new Storage Domain (WIP)
  - Introduces glusterSD.py on the domain side
    - Implements class GlusterStorageDomain(nfsSD) and its associated baggage
      - `findDomain, findDomainPath, getMountPoint, getVolumeClass`
    - Re-uses nfsSD, but in a more acceptable way
    - Support for the new domain class in SD Cache logic (sdc.py)
- GlusterFSConnection for connecting to GlusterFS volume
  - New class GlusterFSConnection, which mounts the domain in a unique mount point path
- GlusterVolume for working with GlusterFS volume type
  - Introduces ability to return custom volume info in volume.py
    - getVmVolumeInfo() added
  - Introduces glusterVolume.py on the volume side
    - Implements class GlusterVolume(fileVolume)

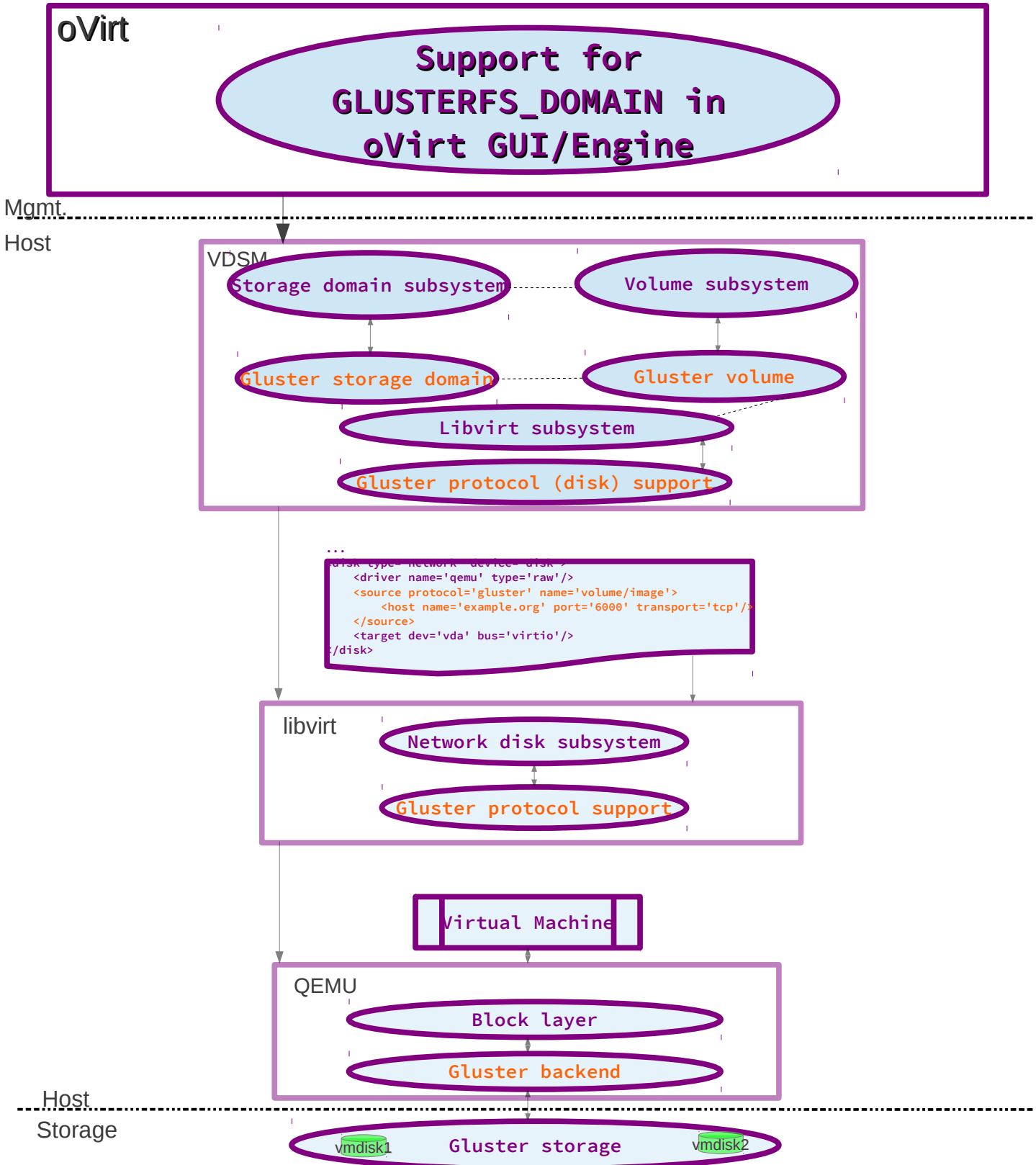
GlusterFS as a new VDSM Storage Domain – Details (contd.)

- GlusterVolume for working with GlusterFS volume type (contd.)
  - Overrides getVmVolumeInfo() to return gluster specific volume info.
  - `{'volType':VmVolumeInfo.TYPE_NETWORK, 'path':glusterPath, 'protocol':'gluster', 'volPort':volPort, 'volTransport':volTrans, 'volfileServer': volfileServer}`
- prepareVolumePath / prepareImage flows modified to take vmVolInfo into account
- Support for network disk type in libvirtvm.py
  - Uses vmVolInfo passed to generate `<disk type='network'...> ... </disk>`
- Pros
  - Exploits QEMU-GlusterFS native integration
  - Fits in the VDSM storage domain idealogy
  - Got +ve response from the community
- Cons
  - None :) Patches under review

GlusterFS domain support in oVirt engine

GlusterFS domain support in oVirt engine

- Support being added to ovirt-engine (OE) to list GLUSTERFS_DOMAIN as a new storage domain (WIP)
  - Similar to POSIXFS_DOMAIN in the OE
- The same params as specified by user for PosixFs domain will be applicable to GlusterFS as well (spec, vfsType, options)
  - **`spec`** : volfileserver:volname
  - **`vfsType`** : glusterfs
  - **`options`** : if any, will be passed as-is to the mount cmdline
- Single pane of glass solution for creating, managing & consuming GlusterFS for storage and virtualization use cases
  - oVirt 3.1 already supports managing GlusterFS based storage cluster – *storage admin perspective*
  - This support will allow oVirt to consume GlusterFS storage cluster as a storage domain / image repository and run VMs off it - *virtualization admin perspective*


oVirt
Support for GLUSTERFS_DOMAIN in oVirt GUI/Engine
Mgmt.
Host
VDSM
Storage domain subsystem
Volume subsystem
Gluster storage domain
Gluster volume
Libvirt subsystem
Gluster protocol (disk) support
...
<driver name='qemu' type='raw'/>
<source protocol='gluster' name='volume/image'>
<host name='example.org' port='6000' transport='tcp'/>
</source>
<target dev='vda' bus='virtio'/>
</disk>
libvirt
Network disk subsystem
Gluster protocol support
Virtual Machine
QEMU
Block layer
Gluster backend
Host
Storage
vmdisk1
Gluster storage
vmdisk2

Future work

- Work towards getting GLUSTERFS_DOMAIN vdsm patches upstream
- Work towards getting GLUSTERFS_DOMAIN oVirt engine patches upstream
- Work towards getting GlusterFS libvirt patches upstream
- GlusterFS as a repository engine in VDSM

References

- GLUSTERFS_DOMAIN support in vdsm - gerrit patchseries
  - http://gerrit.ovirt.org/#/q/status:open+project:vdsm+branch:master+topic:gluster_domain_support,n,z
- GLUSTERFS_DOMAIN feature page on oVirt wiki
  - http://wiki.ovirt.org/wiki/Features/GlusterFS_Storage_Domain
- GLUSTERFS_DOMAIN support in oVirt engine - gerrit patchseries
  - http://gerrit.ovirt.org/#/q/status:open+project:ovirt-engine+branch:master+topic:glusterfs,n,z
- ovirt.org
  - oVirt workshop slides
- VDSM Repository Engines
  - http://gerrit.ovirt.org/#/q/status:open+project:vdsm+branch:master+topic:repo_engine,n,z
- QEMU-GlusterFS native integration
  - Available in upstream QEMU
- Support for Gluster based disks in libvirt XML
  - https://www.redhat.com/archives/libvir-list/2012-October/msg00085.html
- Storage virtualization for KVM – Putting the pieces together
  - http://www.linuxplumbersconf.org/2012/wp-content/uploads/2012/09/2012-lpc-virt-storage-virt-kvm-rao.pdf

Part 2: Supporting storage array offloads from VDSM

Agenda

- Storage arrays overview
- What is libStorageMgmt (libSM)
- libSM architecture
- VDSM – libSM integration
- Future work
- References

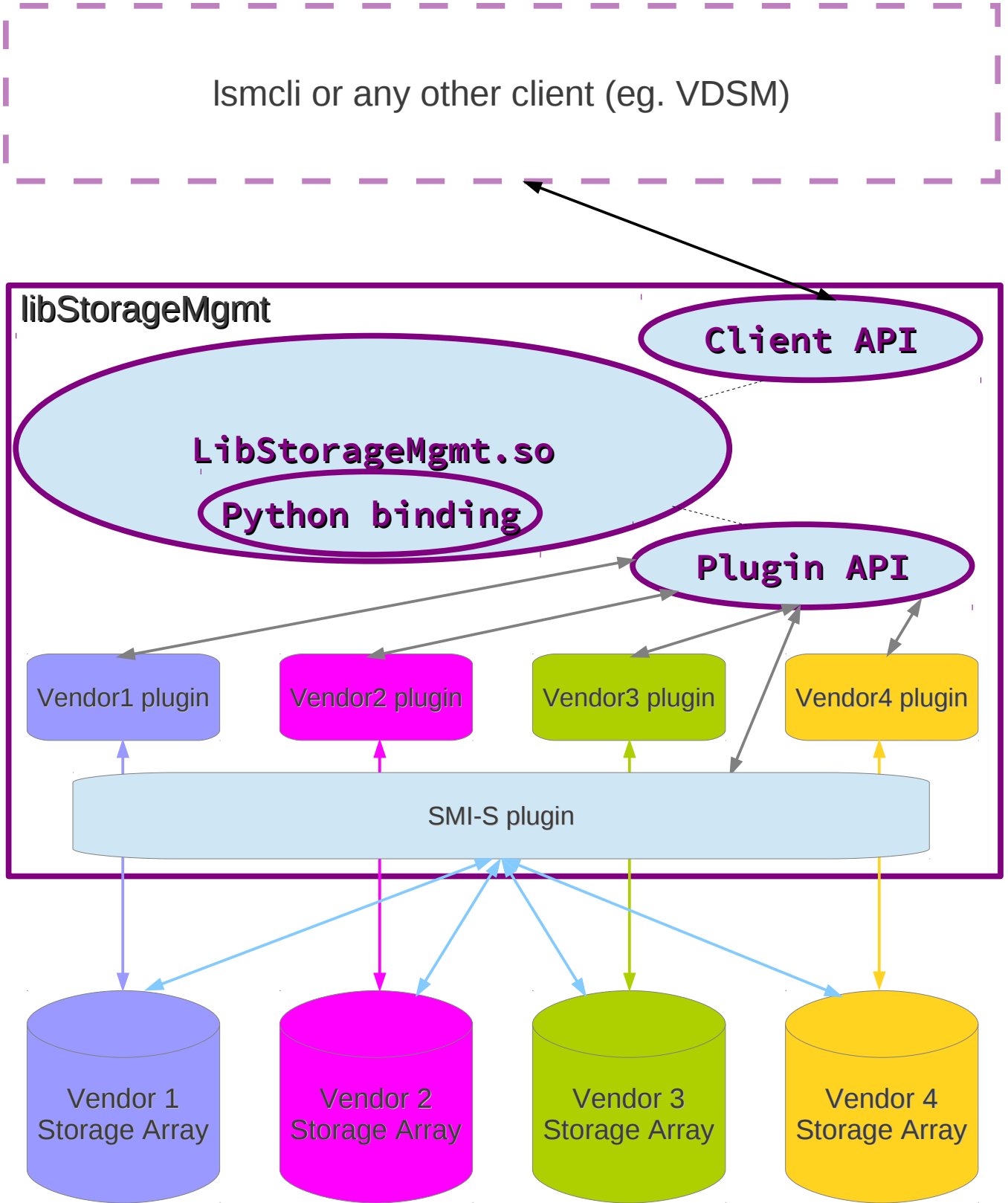
Storage arrays overview

- External storage array types
  - NAS (File based)
    - Export filesystem
    - Work at file granularity
    - Eg: IBM SoNAS
  - SAN (Block based)
    - Export storage pool and volumes
    - Work at LUN granularity
    - Eg: IBM V7K, SVC etc.
- Interfaces
  - SMI-S
    - Basic management operations
    - Eg: LUN creation, deletion, listing etc.
  - CLI
    - Enhanced/specialised services
    - Eg: IBM Flashcopy, global mirror etc
  - T10 extensions
    - Eg: XCOPY, WRITESAME, ATS

What is libStorageMgmt?

- Open source, vendor agnostic library which provides an API for managing external storage arrays
- Licensed under GNU Lesser General Public License
- Command line interface, lsmcli
- Daemon for executing plugins in separate process
- Current array support (varying levels of functionality)
  - NetApp
  - Linux software target
  - SMI-I compliant arrays

Source: LPC 2012


lsmcli or any other client (eg. VDSM)
libStorageMgmt
Client API
LibStorageMgmt.so
Python binding
Plugin API
Vendor1 plugin
Vendor2 plugin
Vendor3 plugin
Vendor4 plugin
SMI-S plugin
Vendor 1
Storage Array
Vendor 2
Storage Array
Vendor 3
Storage Array
Vendor 4
Storage Array

VDSM – libSM integration

- Goals
  - Ability to plugin external storage array into oVirt/VDSM virtualization stack, in a vendor neutral way
  - Ability to list features/capabilities of the array
    - Eg: copy offload, thinp etc.
  - Ability to utilize the storage offload capabilities from oVirt/VDSM
- **Work in progress**
  - Initial proposal discussed on the ML
  - https://lists.fedorahosted.org/pipermail/vdsm-devel/2012-May/001011.html

VDSM – libSM integration contd.

- VDSM can use libSM to
  - Provision storage
    - Done by storage admin
    - Manage LUNs (create, delete, resize)
    - LUN mapping/zoning for hosts to see LUNs
  - Consume storage
    - Done by virtualization admin
    - Refresh hosts for them to see LUNs
    - Create new storage domain on the array using the LUN provisioned
    - Discover the array capabilities
    - Specify whether to use storage offload or not

VDSM – libSM integration contd.

- Potential flow
  - Create snapshot
    - VDSM will check the snapshot offload capability in the domain metadata
    - If available, and override is not configured, it will use libSM to offload LUN/File snapshot
    - If override is configured or capability is not available, it will use its internal logic to create snapshot (qcow2)
- Flow for other tasks (eg: copy/clone vmdisk) will be similar

VDSM – libSM integration contd.

- Potential approach
  - libSM as a repository engine in VDSM
    - Repo engine not mainstream yet
    - Initial patches are available, needs investigation on whether they are sufficient to integrate libSM
  - libSM interface level changes (if needed) to suit the needs of repository engine integration

Challenges

- What does a VMdisk map to ?
  - LUN Vs LV
  - Arrays provision LUNs
  - VDSM uses VG/LV combination today
  - Storage offloads are typically at LUN level
  - Too many LUNs – not preferred by storage admins
  - Sub-LUN offloads not available in all arrays
- How will libSM co-exist with VDSM in a cluster of hosts ?
  - On one of the VDSM node ?
  - On all VDSM nodes
    - Some arrays impose limitation on the number of clients accessing the array at the same time.
- Lots more
  - See the ML discussion thread for more details

Future work

- Follow up on the ML discussion thread
  - Reach to conclusion
  - Request more community participation
- Follow up on repository engine work
- Work towards a PoC and full implementation

References

- Proposal discussed on the ML
  - https://lists.fedorahosted.org/pipermail/vdsm-devel/2012-May/001011.html
- libSM home page
  - https://sourceforge.net/projects/libstoragemgmt/
- LPC 2012 presentation on libSM
  - http://sourceforge.net/projects/libstoragemgmt/files/documentation/LPC_lsm_2012.odp/download
- Repository engine
  - http://gerrit.ovirt.org/#/q/status:open+project:vdsm+branch:master+topic:repo_engine,n,z

Legal Statement

- This work represents the view of the authors and does not necessarily represent the view of IBM.
- IBM, IBM(logo) are trademarks or registered trademarks of International Business Machines Corporation in the United States and/or other countries.
- Linux is a registered trademark of Linus Torvalds.
- Other company, product, and service names may be trademark or service marks of others.
- There is no guarantee that the technical solutions provided in this presentation will work as-is in every situation.