Integrating GlusterFS & supporting storage array offloads in VDSM

Deepak C Shetty – deepakcs@linux.vnet.ibm.com
Harsh P Bora – harsh@linux.vnet.ibm.com

(IBM Linux Technology Center, Bangalore, India)

Oct 2012

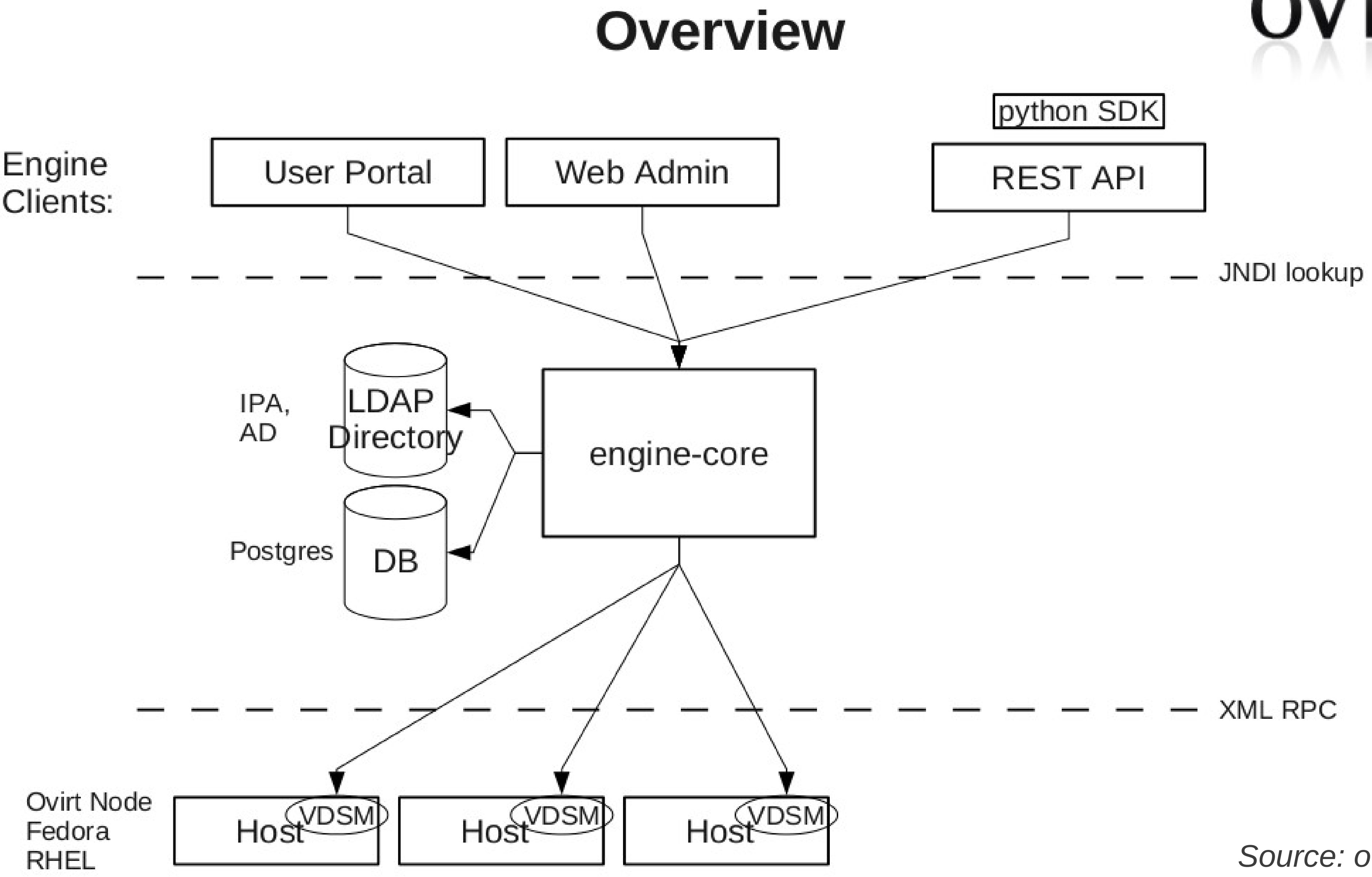
oVirt workshop, Bangalore

Integrating GlusterFS as a storage domain in VDSM

Agenda

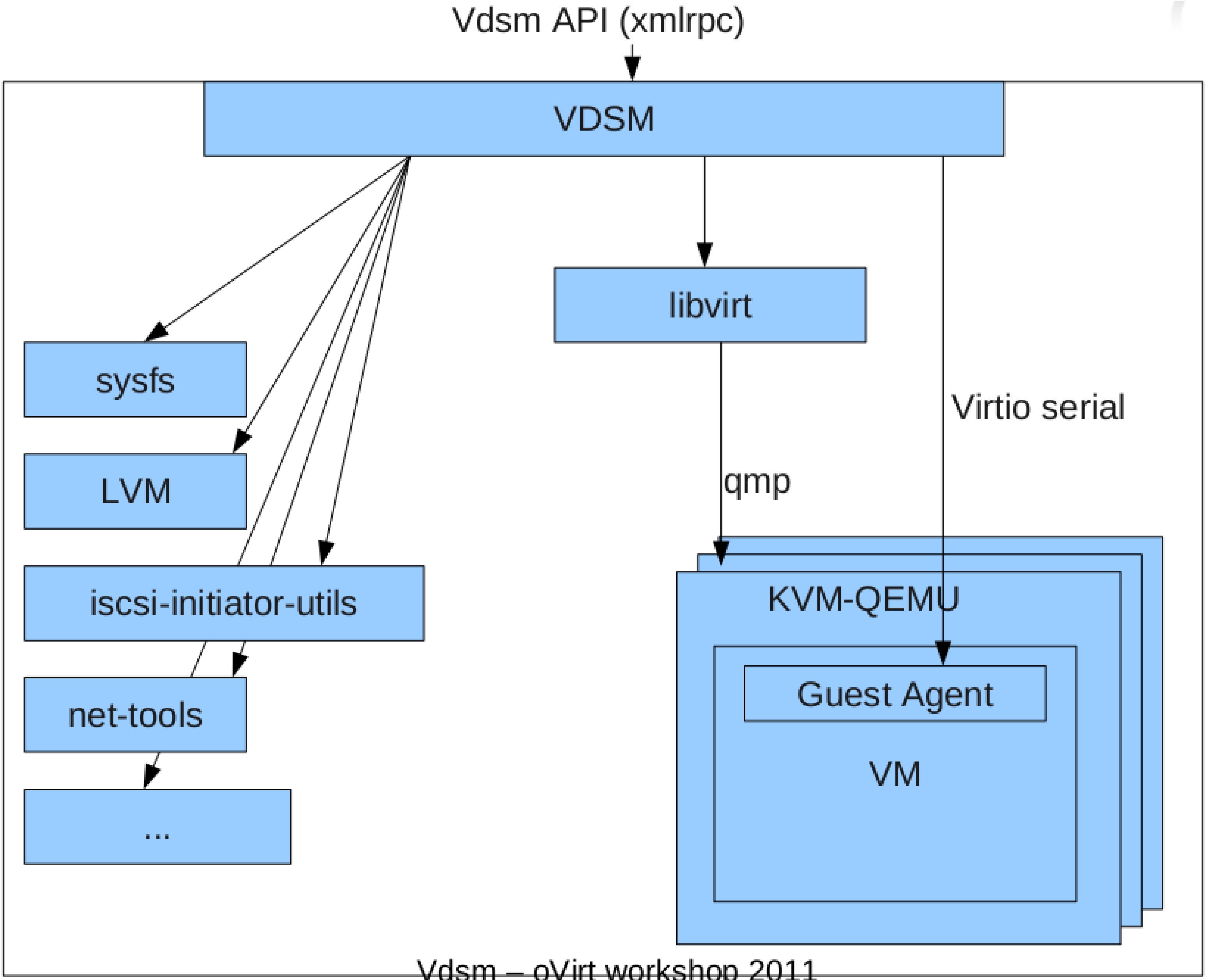
- oVirt architecture
- VDSM architecture
- VDSM storage concepts
- What is GlusterFS
- Enabling GlusterFS for Virtualization use
- QEMU-GlusterFS integration
- GlusterFS as a VDSM Storage Domain
  - Different approaches
- GlusterFS support in libvirt
- Future work
- References

oVirt architecture



Source: ovirt.org

VDSM architecture



Vdsm – oVirt workshop 2011

Source: ovirt.org

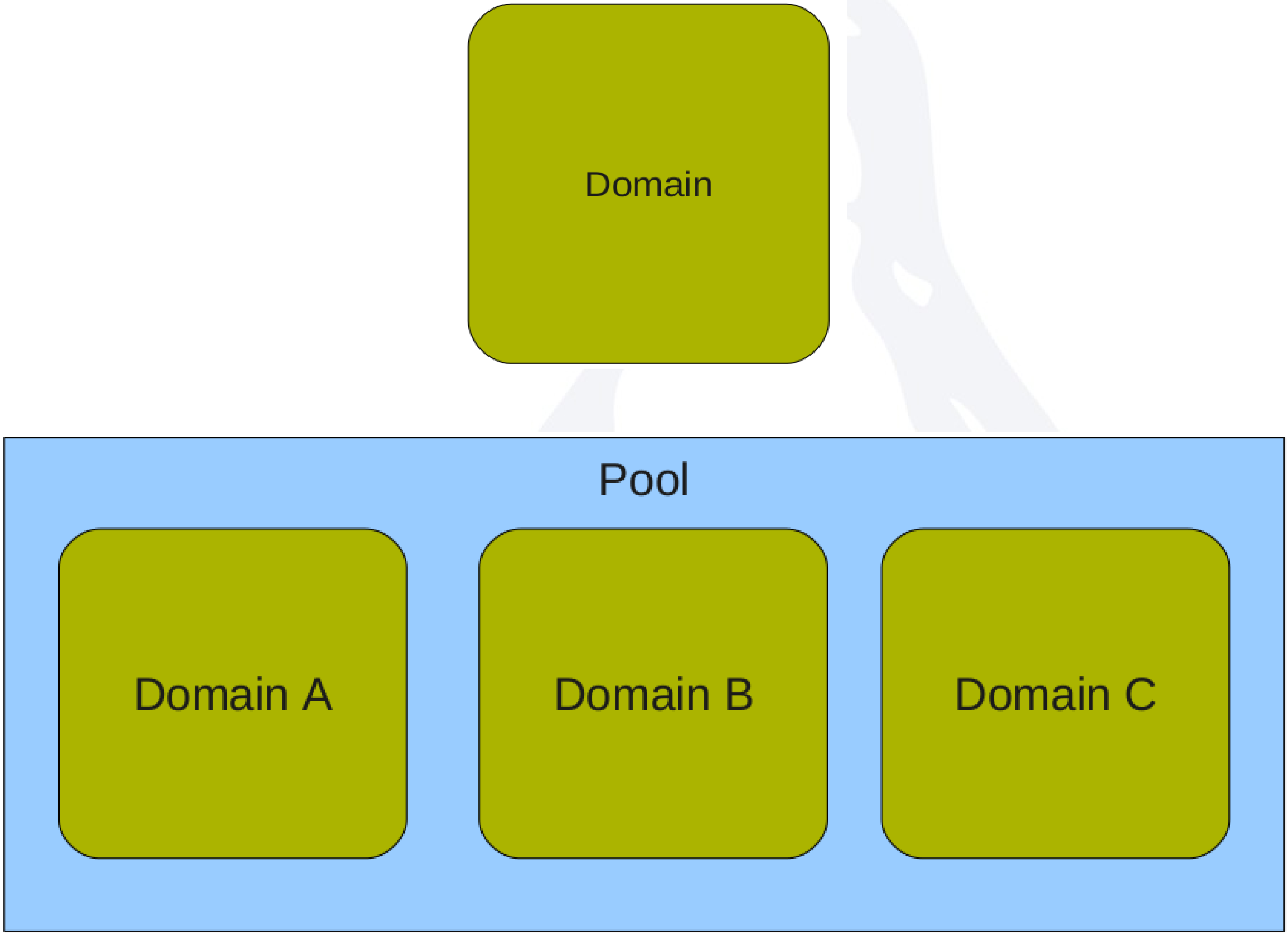
So what exactly is VDSM?

- **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?

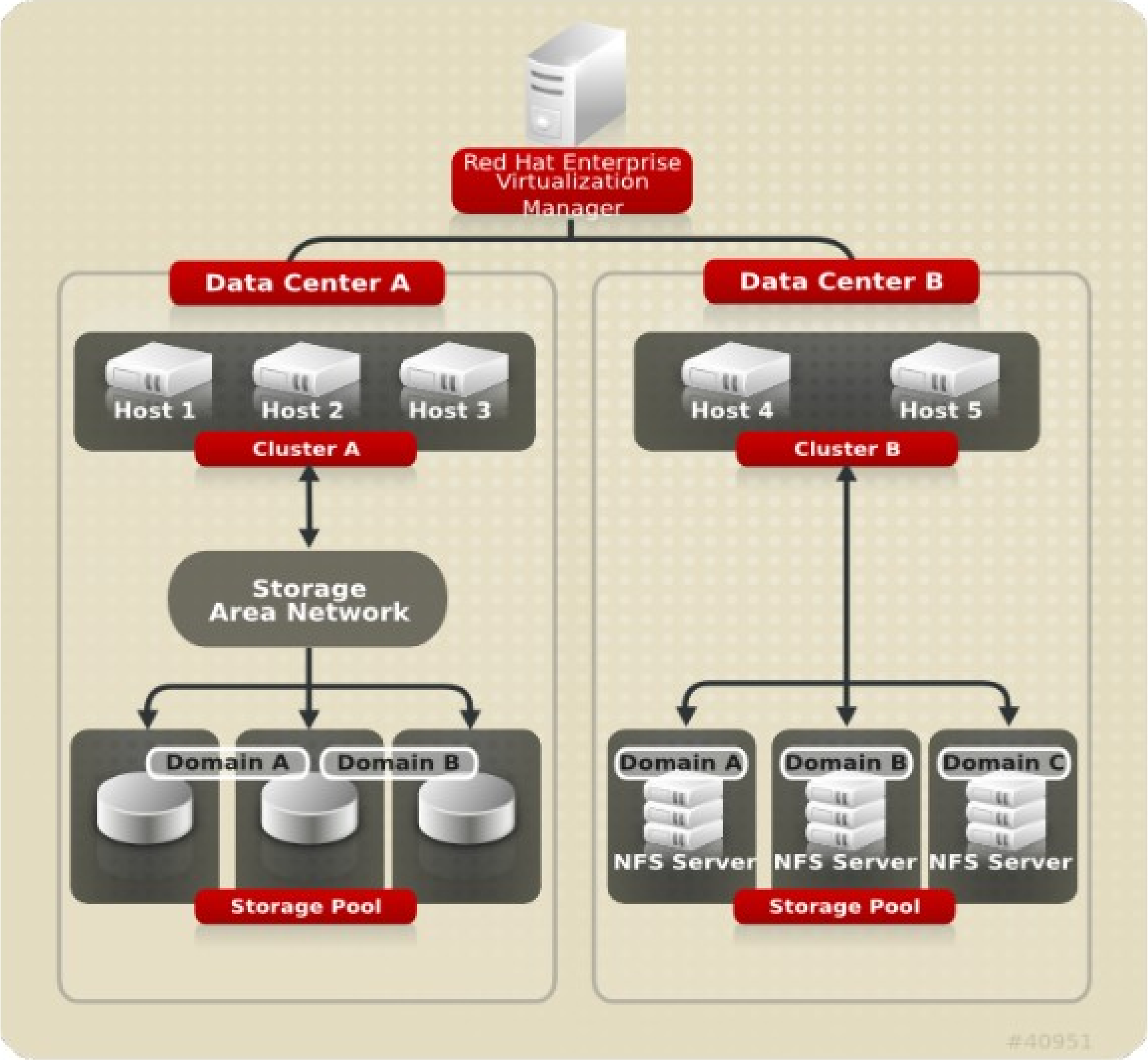
- Host bootstrap and registration
- VM life cycle (via libvirt)
- Guest interaction (sso, stop, etc)
- **Storage management**
- Network configuration
- Monitoring host and VMs
- Policy management
- Fencing proxy

VDSM storage concepts



Source: ovirt.org

Example



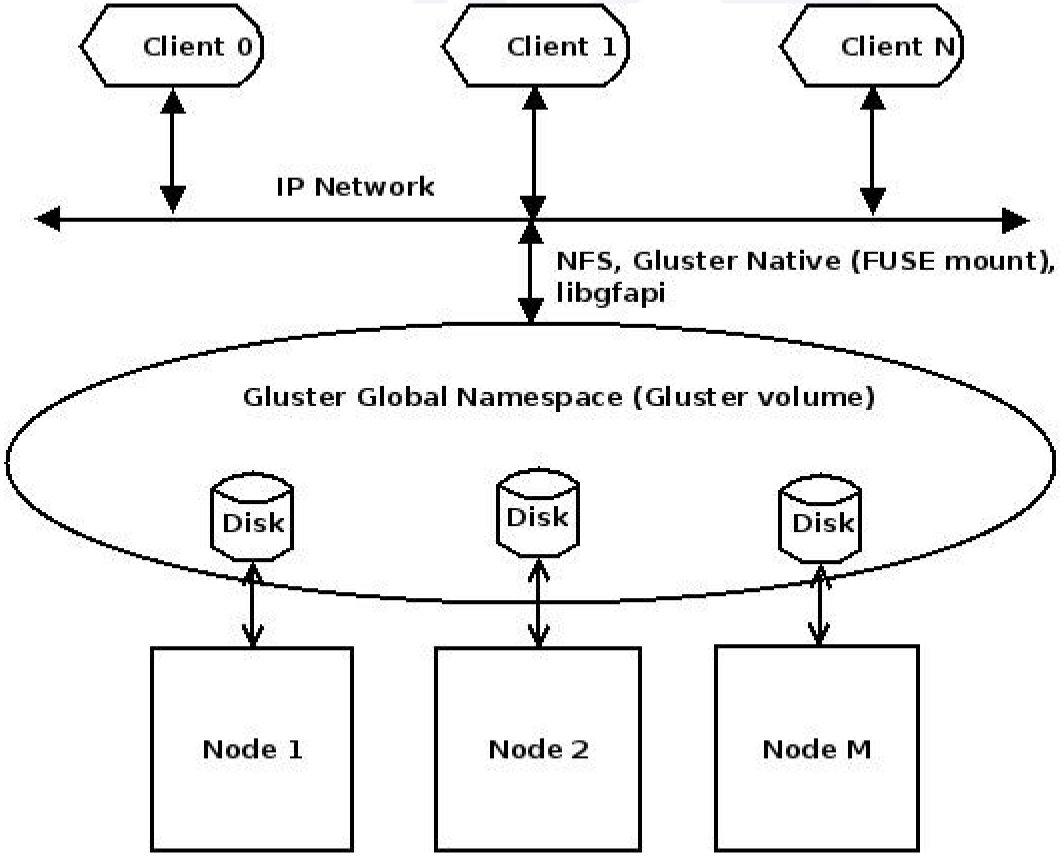
Source: ovirt.org

VDSM storage concepts contd.

- Storage domain
  - Fundamental storage entity
  - Standalone entity.
  - It acts as the image repository
- Domain types
  - File based (NFS, PosixFS, local)
    - **Extended to support GlusterFS**
  - 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.

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).


Client 0
Client 1
Client N
IP Network
NFS, Gluster Native (FUSE mount), libgfapi
Gluster Global Namespace (Gluster volume)
Disk
Disk
Disk
Node 1
Node 2
Node M

Source: LPC 2012

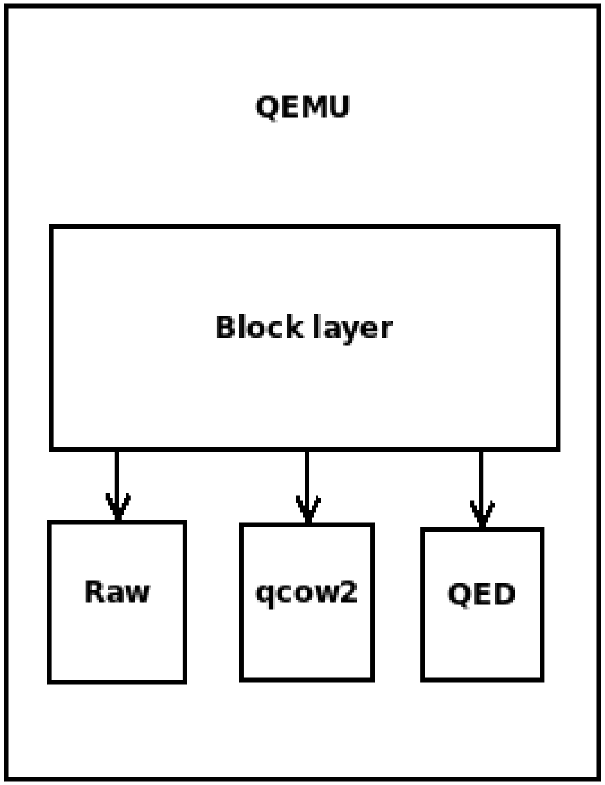
Enabling GlusterFS for Virtualization use

- QEMU-GlusterFS integration
  - Native integration, no FUSE mount
    - **New VDSM storage domain is required to exploit this**
  - 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

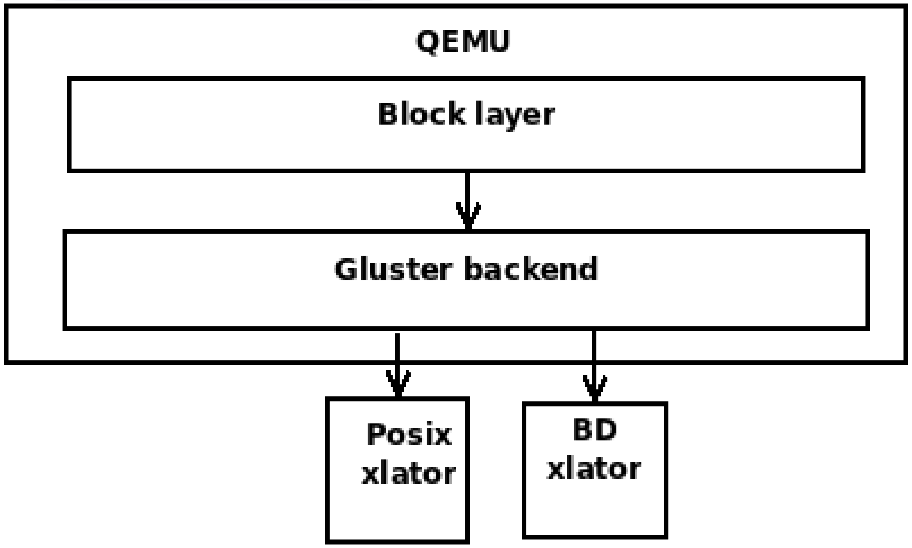
Source: LPC 2012

QEMU – 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

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**
  - Patches under review
  - Introduces GlusterStorageDomain class
  - Introduces GlusterVolume class
  - Introduces GlusterFSConnection class
  - Adds support for network disk type in libvirtvm.py

GlusterFS as a new VDSM Storage Domain - Details

- GLUSTERFS_DOMAIN as a new Storage Domain
  - 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.

        - 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.
- Reference
    - http://gerrit.ovirt.org/#/c/6856/

GlusterFS support in Libvirt

- Abstraction layer above Hypervisors
- Define Virtual Machines using XML
- VDSM uses Libvirt to talk to QEMU-KVM in oVirt stack
- 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 to support GlusterFS (Work in progress)
  - http://www.redhat.com/archives/libvir-list/2012-October/msg00085.html

GlusterFS support in Libvirt

- Proposed Libvirt XML format to specify disk images on glusterfs is:

```
<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>
```

Future work

- Work towards getting GLUSTERFS_DOMAIN patch upstream
- Work towards getting GlusterFS libvirt patches upstream
- UI/Engine side enablement
- GlusterFS as a repository engine in VDSM

References

- GLUSTERFS_DOMAIN gerrit patchseries
  - http://gerrit.ovirt.org/#/c/6856/
  - See gluster_domain_support topic branch
- 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

Supporting storage array offloads from VDSM

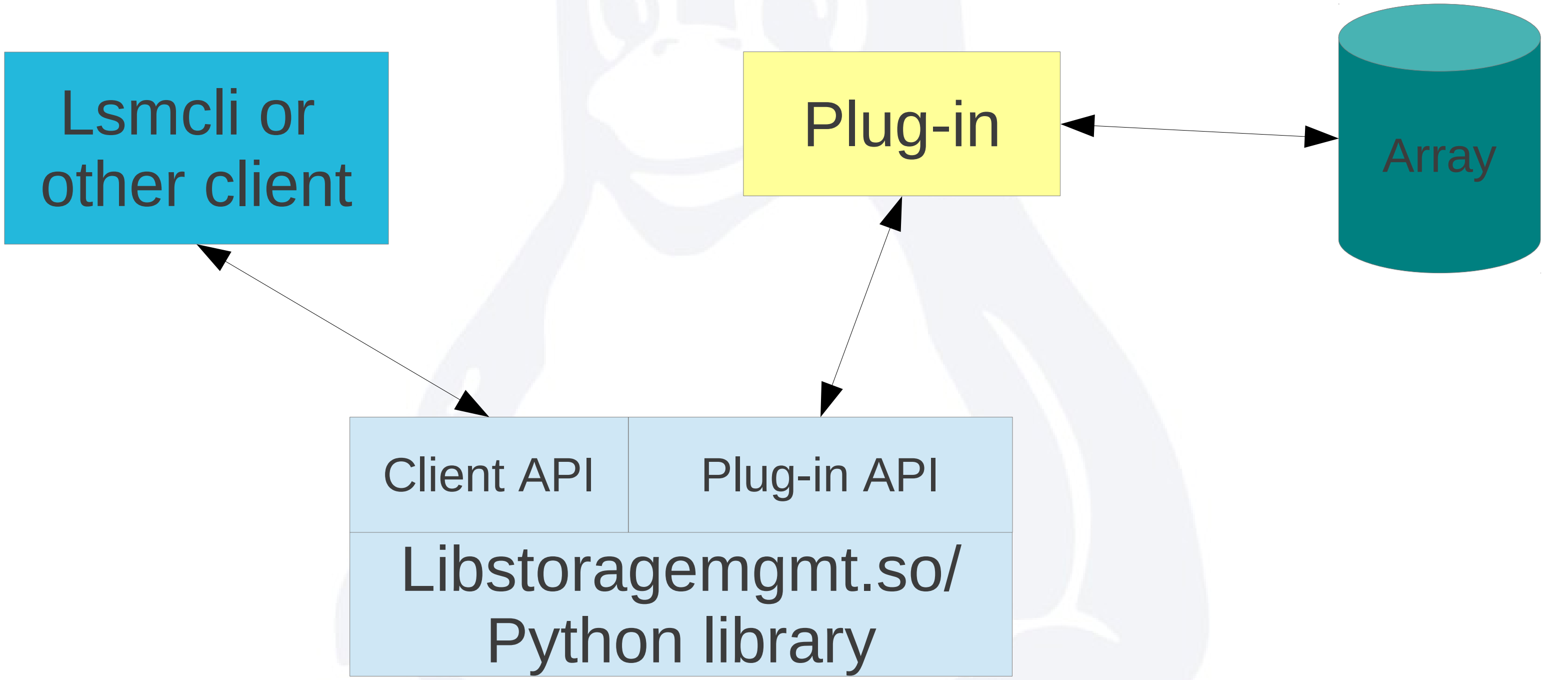
Agenda

- 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

libSM architecture



Source: LPC 2012

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

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 flows
  - 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)
  - Copy/Clone vmdisk flow
    - Same as above

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.