VDSM

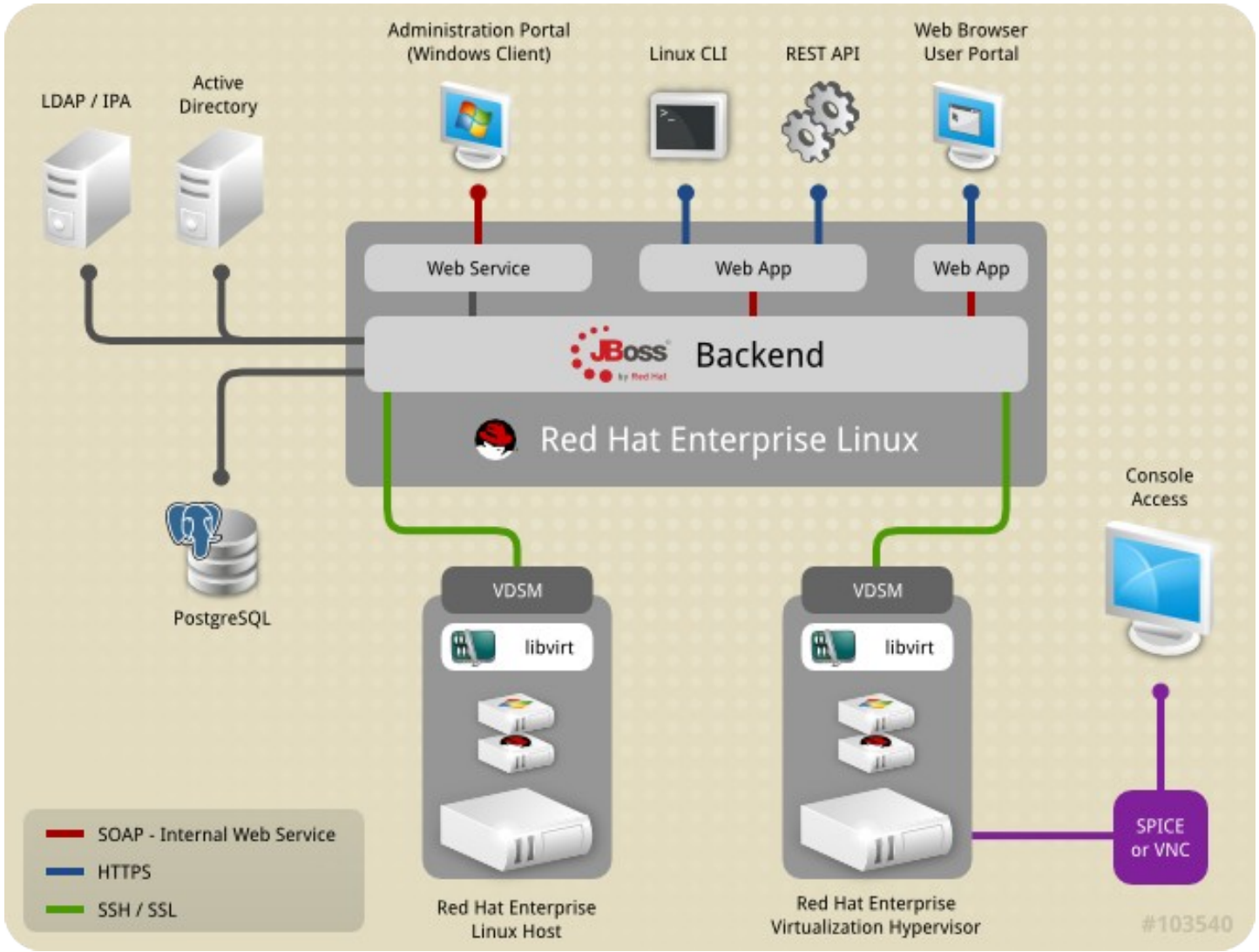
The oVirt Node Management Agent

Federico Simoncelli
Senior Software Engineer, Red Hat
February 2012

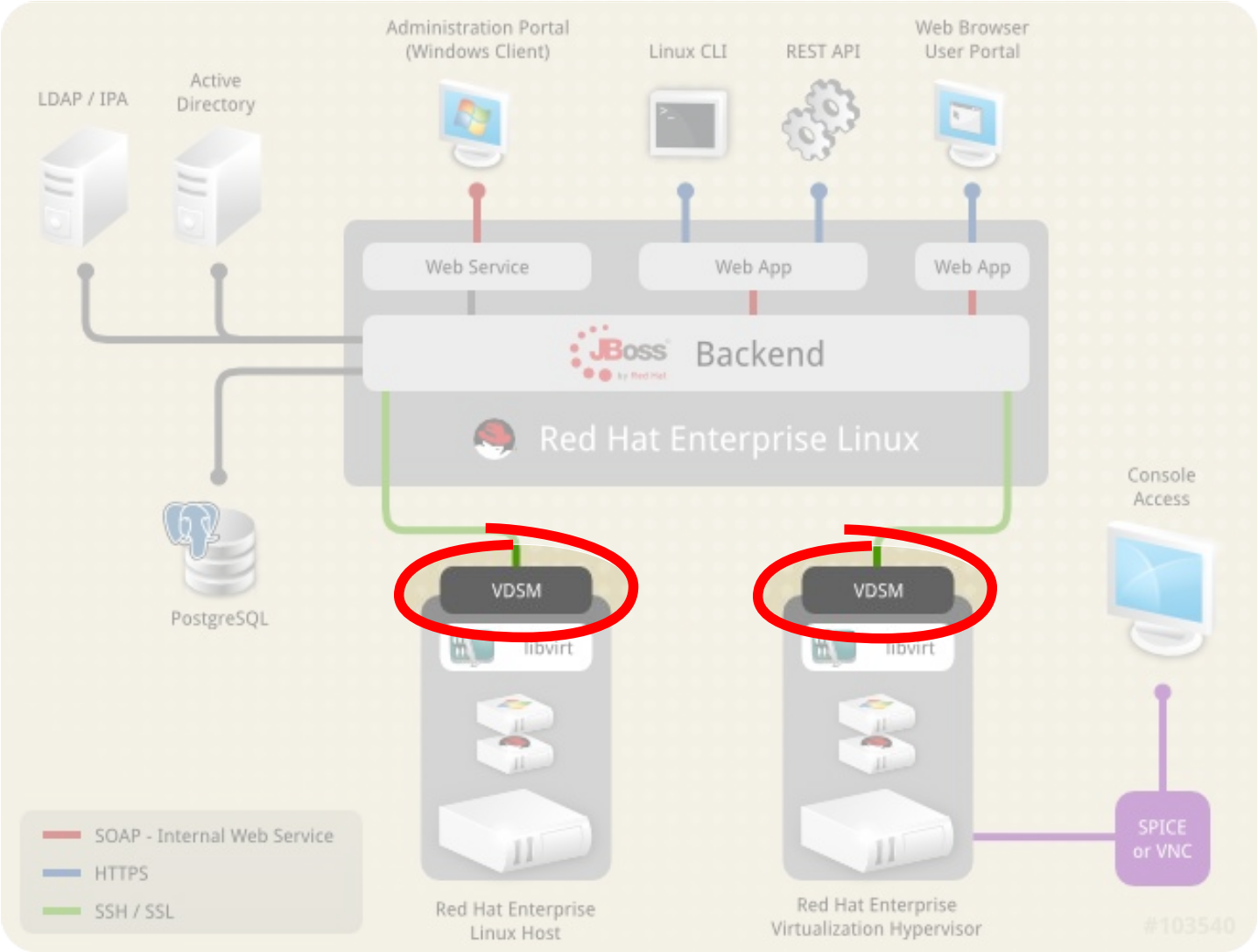
Agenda

- The oVirt Virtualization Architecture
- The oVirt Agent Requirements
- What is VDSM?
- Why use VDSM?
- Storage Architecture
- Storage Pool Manager (SPM)
- Thin Provisioning
- API Examples
- Roadmap
- How To Contribute

The oVirt Virtualization Architecture



The oVirt Virtualization Architecture



The oVirt Agent Requirements

- **Virtualization**
  - Take advantage of the latest Virtualization Technologies
  - Present a wide range of virtual devices
    (CPU, memory, buses and controllers as: PCI, IDE, SCSI, USB, etc...)
  - Provide additional operations
    (Pause, hibernate, migrate, snapshot, etc...)
- **Storage**
  - Manage tens of thousands of virtual disk images (cluster aware)
  - Each image potentially accessible from hundreds of nodes
  - Prepare, monitor and manage different types of storages
    (File based as the local filesystem and NFS, and block based as ISCSI and FCP)
- **Tools**
  - Configure the node
    (Install the required packages, configure the network, optimize configuration...)

What is VDSM?

- oVirt Node Agent
  (A daemon tailored for oVirt needs but it can be used by any other management platform)
- High level API for managing the cluster nodes
  (Abstracts low level details of underlying Linux environments)
- It manages transient VMs using libvirt and qemu-kvm
  (The VMs definition is stored centrally by oVirt)
- Written in Python
- Multi-threaded and multi-process
- Highly reliable: robustness as a design goal
  (No single point of failure, continues working in the absence of the manager)
- Multihost system, one concurrent metadata writer (SPM)
  (Scales linearly in data writers)
- Speaks with its guest agent via virtio-serial
- Platforms: RHEL5, RHEL6, Fedora, Debian and Ubuntu (in progress)

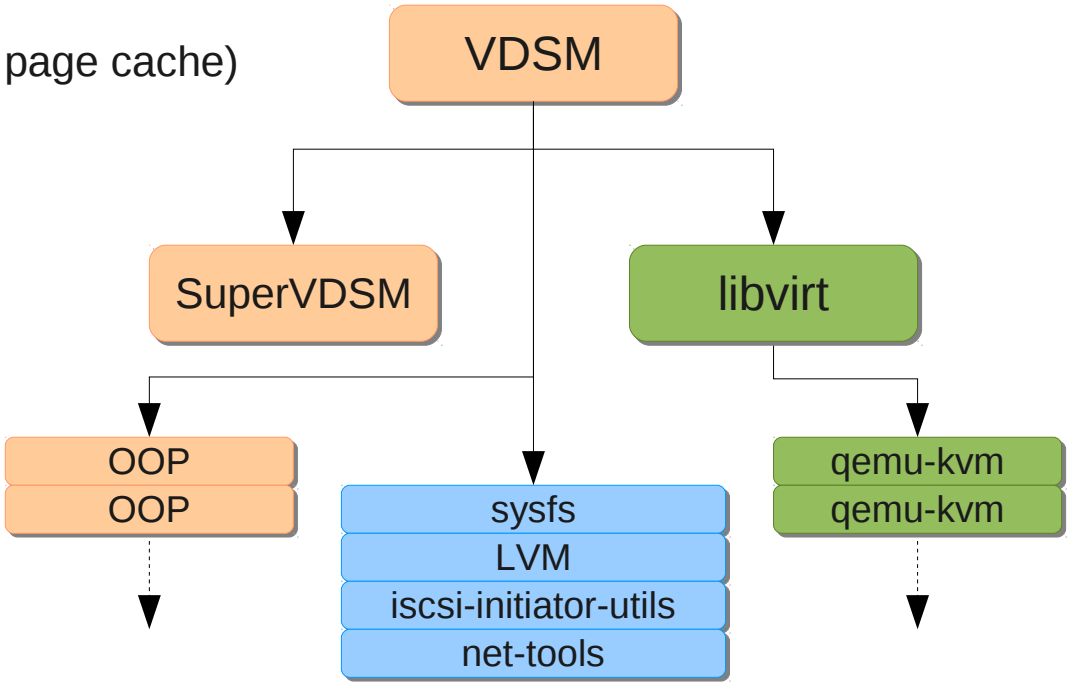
What is VDSM?

Responsibilities

- Host bootstrap and registration
- VM life cycle
- Storage and network management
- Host and VMs monitoring
- Policy management
  (Scheduler, KSM, thin provisioning, page cache)

Infrastructure

- SuperVDSM
- Out of process – process pool
- Asynchronous tasks



Why VDSM?

- $ qemu-kvm & Voila! We have a virtual machine, but read the fine print:

```
/usr/libexec/qemu-kvm -S -M rhel6.0.0 -cpu Conroe -enable-kvm -m 2048 -smp 1,sockets=1,cores=1,threads=1 -name z-win7x86-1 -uuid e3e19b36-f6b7-4ab9-b604-1f8b5c471bda -smbios type=1,manufacturer=Red
Hat,product=RHEL,version=6Server-6.1.0.2.el6_1,serial=50C1C6F0-B18B-11DE-ADF1-00215EC7FC0C_00:1A:64:E7:0E:E0,uuid=e3e19b36-f6b7-4ab9-b604-1f8b5c471bda -nodefconfig -nodefaults -chardev
socket,id=charmonitor,path=/var/lib/libvirt/qemu/z-win7x86-1.monitor,server,nowait -mon chardev=charmonitor,id=monitor,mode=control -rtc base=2011-08-04T06:17:36 -boot cdn -device virtio-serial-pci,id=virtio-
serial0,max_ports=16,bus=pci.0,addr=0x6 -drive file=/rhev/data-center/6927f974-c6f6-482f-aca9-907c4acc71a9/50027e48-6cb9-4345-9c7a-c22b41ad84d2/images/5ada0ef6-5f4a-40b8-ad92-cb6758de8536/c22f4e68-439b-4a87-8e22-
bc7d8e2391f1,if=none,id=drive-ide0-0-0,format=qcow2,serial=b8-ad92-cb6758de8536,cache=none,werror=stop,rerror=stop,aio=native -device ide-drive,bus=ide.0,unit=0,drive=drive-ide0-0-0,id=ide0-0-0 -drive file=/rhev/data-
center/6927f974-c6f6-482f-aca9-907c4acc71a9/e0acfcc8-c020-413e-84cd-a93cb0ab9b2d/images/11111111-1111-1111-1111-111111111111/RHEV-toolsSetup_3.0_12.iso,if=none,media=cdrom,id=drive-ide0-1-
0,readonly=on,format=raw -device ide-drive,bus=ide.1,unit=0,drive=drive-ide0-1-0,id=ide0-1-0 -drive file=/rhev/data-center/6927f974-c6f6-482f-aca9-907c4acc71a9/50027e48-6cb9-4345-9c7a-c22b41ad84d2/images/f52621e0-8b1e-
47af-809c-45de2aa697fc/f77b5dd2-3141-4ea7-84fa-e8cfffe9cff9,if=none,id=drive-virtio-disk0,format=qcow2,serial=af-809c-45de2aa697fc,cache=none,werror=stop,rerror=stop,aio=native -device virtio-blk-
pci,bus=pci.0,addr=0x7,drive=drive-virtio-disk0,id=virtio-disk0 -netdev tap,fd=27,id=hostnet0 -device rtl8139,netdev=hostnet0,id=net0,mac=00:1a:4a:23:11:0b,bus=pci.0,addr=0x3 -netdev tap,fd=29,id=hostnet1,vhost=on,vhostfd=30
-device virtio-net-pci,netdev=hostnet1,id=net1,mac=00:1a:4a:23:11:0c,bus=pci.0,addr=0x4 -chardev socket,id=charchannel0,path=/var/lib/libvirt/qemu/channels/z-win7x86-1.com.redhat.rhevm.vdsm,server,nowait -device
virtserialport,bus=virtio-serial0.0,nr=1,chardev=charchannel0,id=channel0,name=com.redhat.rhevm.vdsm -chardev spicevmc,id=charchannel1,name=vdagent -device virtserialport,bus=virtio-
serial0.0,nr=2,chardev=charchannel1,id=channel1,name=com.redhat.spice.0 -usb -spice port=5902,tls-port=5903,addr=0,x509-dir=/etc/pki/vdsm/libvirt-spice,tls-channel=main,tls-channel=inputs -k en-us -vga qxl -global qxl-
vga.vram_size=67108864 -device intel-hda,id=sound0,bus=pci.0,addr=0x5 -device hda-duplex,id=sound0-codec0,bus=sound0.0,cad=0
```

- To manage multiple virtual machines you would need libvirt: virsh, virt-manager
- To dynamically manage anything from a few VMs on a single host up to thousands of VMs on a cluster of hundreds of hosts using multiple storage targets: VDSM
- Robustness as a design goal
  - Evaporated NFS exports or faulty multipath
  - Node crashes and self-fencing of metadata writer
  - Live-locked qemu processes and internal Python exceptions

Storage Architecture

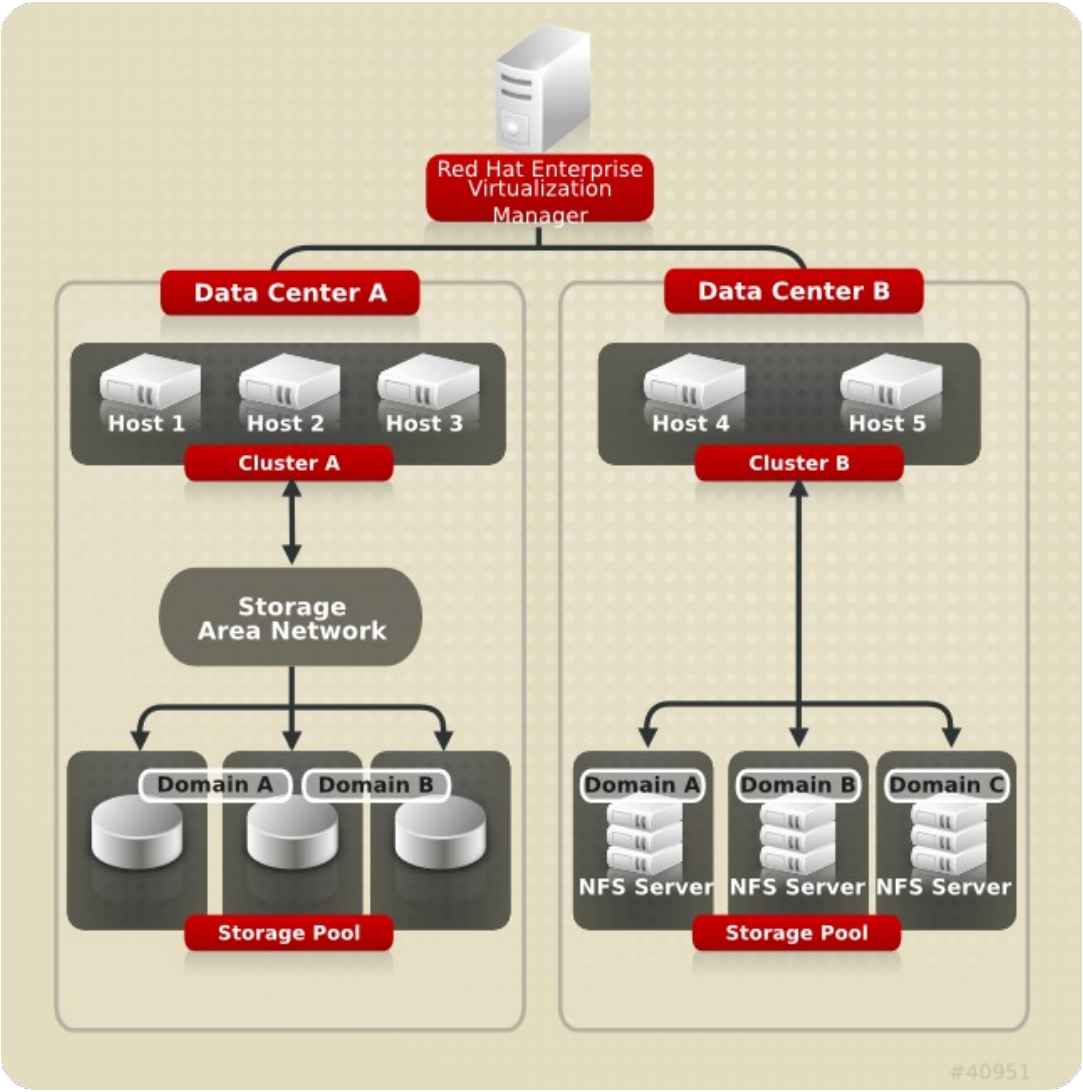
Centralized storage system
(disk images, templates, etc...)

Storage Domain

- A standalone storage entity
  (implemented with NFS, FCP, iSCSI, FCoE, and SAS)
- Stores the images and associated metadata
- Only true persistent storage for VDSM

Storage Pool

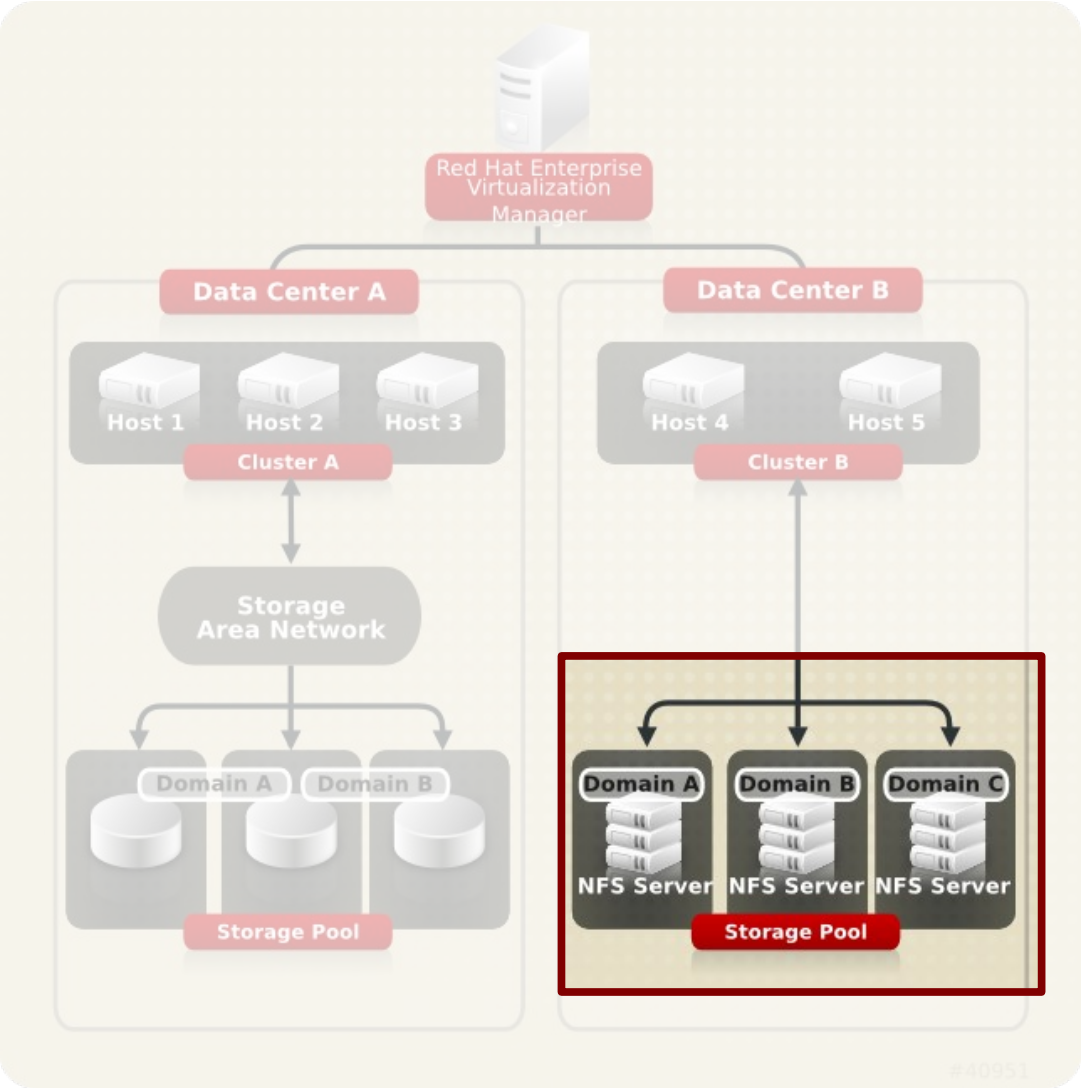
- Aggregates several Storage Domains
  (it will be deprecated in the future)
- Supposed to simplify cross domain operations



Storage Architecture

File Storage Domains

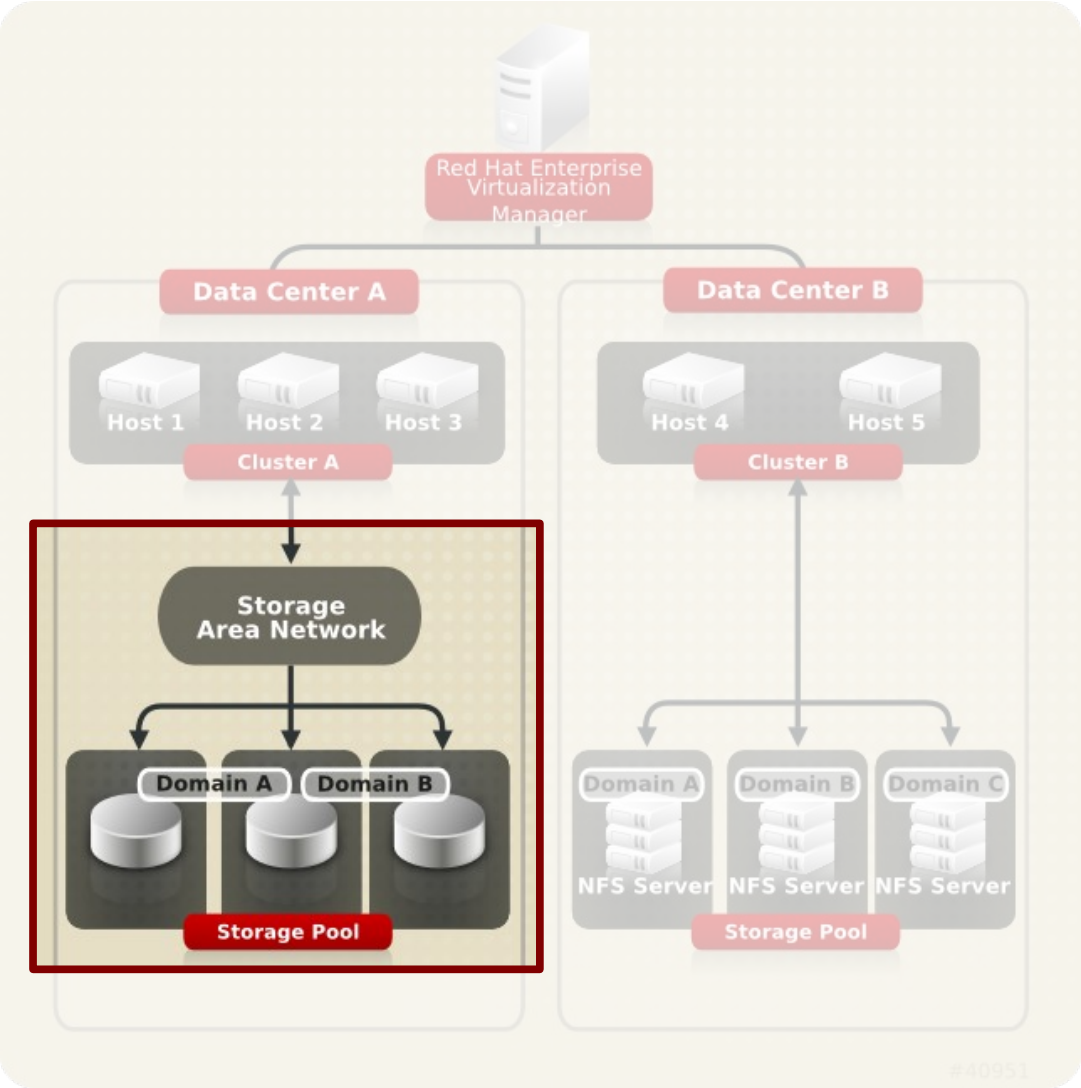
- Use file system features for segmentation
- Use file system for synchronizing access
- Sparse files
- Better image manipulation capabilities
- Volumes and metadata are files
- 1:1 Mapping between domain and NFS export



Storage Architecture

Block Storage Domains

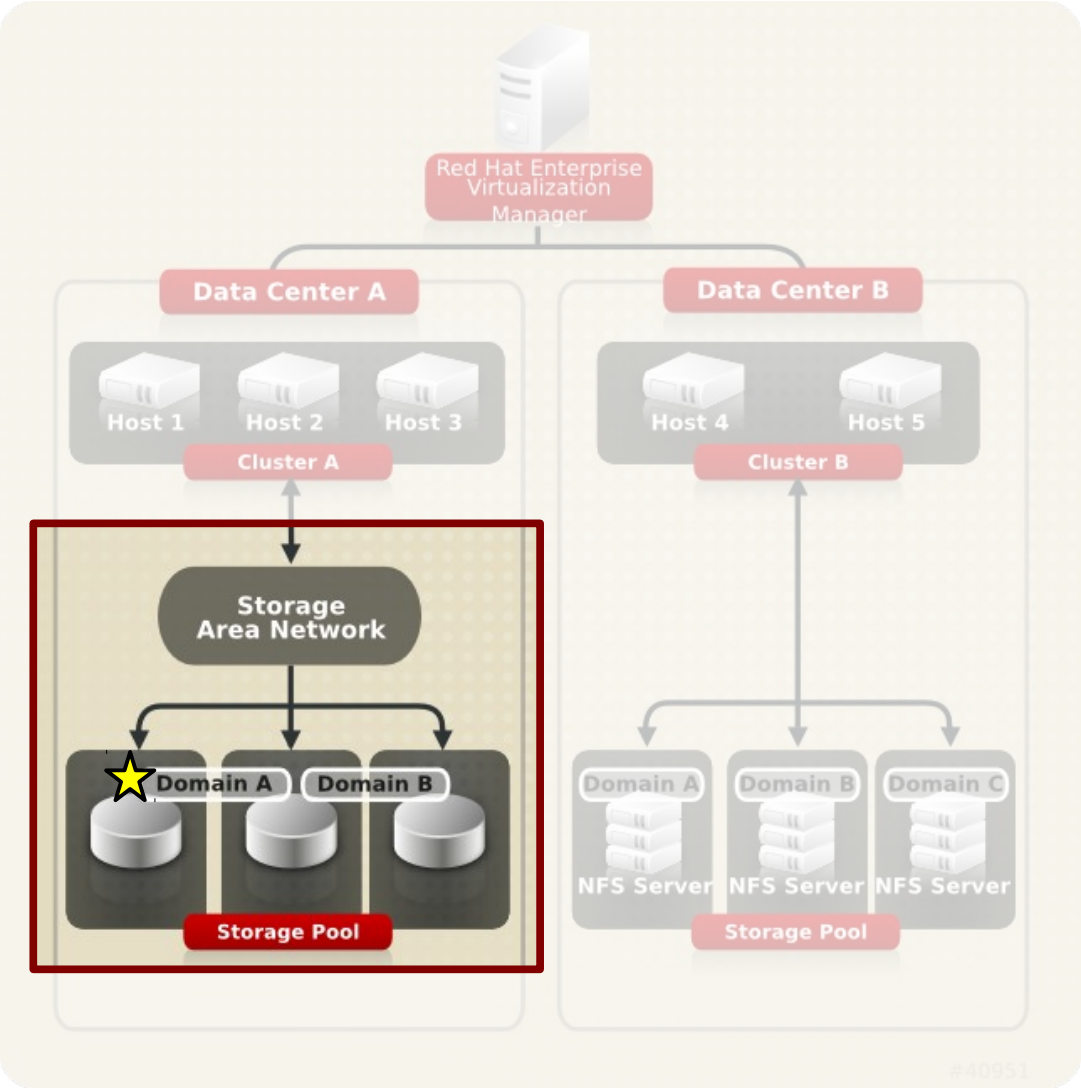
- Use LVM for segmentation
- Very specialized use of LVM
- Mailbox
  - Thin provisioning
- Devices managed by device-mapper and multipath
- Domain is a VG
- Metadata is stored in a single LV and in LVM tags
- Volumes are LVs



Storage Architecture

Master Domain

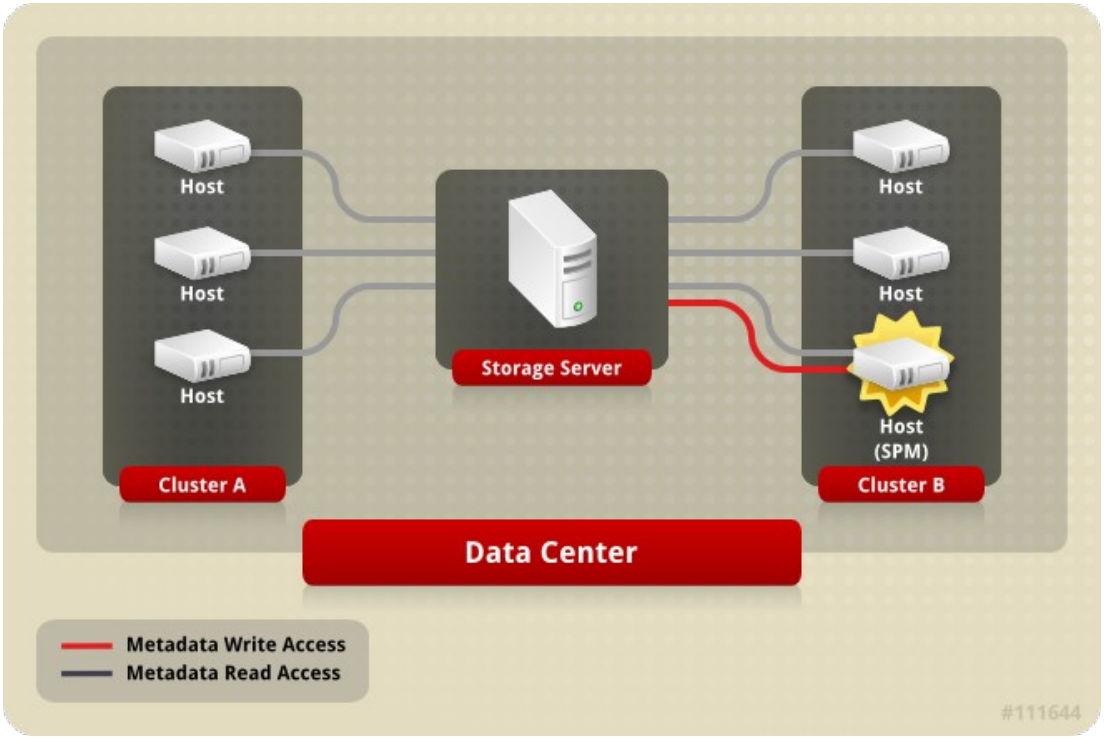
- Used to store:
  - Pool metadata
  - Backup of OVFs
    (treated as blobs)
  - Async tasks
    (persistent data)
- Contains the clustered lock for the pool



Storage Pool Manager (SPM)

- The SPM is a role assigned to one host in a data center giving the host sole authority to make all storage domain structure changes
- The role of SPM can be migrated to any host in a data center

- Creation, deletion and manipulation of Virtual Disks, Snapshots and Templates
- Allocation of storage for sparse block devices (on SAN)
- Single meta data writer
- SPM lease mechanism (Chockler and Malkhi 2004, Light-Weight Leases for Storage-Centric Coordination)
- Storage-centric mailbox

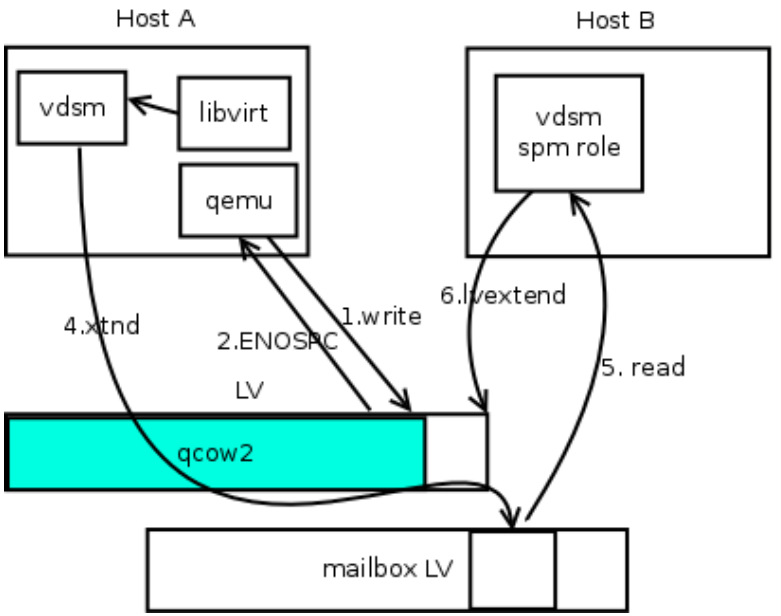


Thin Provisioning

Over-Commitment is a storage function which allows RHEV-M to logically allocate more storage than is physically available

- Generally, Virtual Machines use less storage than what has been allocated to them
- Virtual Machine to operate completely unaware of the resources that are actually available

- QEMU identifies the highest offset written onto the logical volume
- VDSM monitors the highest offset marked by QEMU
- VDSM requests to the SPM to extend the logical volume when needed



API Examples

Host:

- getVdsCapabilities
- getVdsStats

Virtual Machine:

- create, destroy, pause, continue
- changeCD, changeFloppy
- migrate, hibernate
- getAllVmStats, getVmStats

Network:

- addNetwork, delNetwork, editNetwork
- setSafeNetworkConfig, setupNetworks
  - ConnectivityCheck

Async Tasks:

- getAllTasksStatuses, getTaskStatus
- clearTask
- stopTask, revertTask

Roadmap

Work in progress

- SANLock
- Live Snapshots
- Disks and network hotplug
- Connection management
- Support any shared filesystem
- NFSv4
- Direct LUN
- New API
  (clean, eg: createVG and createStorageDomain, stable, oVirt-API look and feel)

Future

- SDM
- cgroups
  (CPU, memory, I/O, network)
- Monitoring using collectd?
- Support sending events, QMF
- Split VDSM into reusable autonomous parts
  - Spin storage off as a generic image repository

How To Contribute

- **Website and Repository:**
  - http://www.ovirt.org/wiki/Vdsm
  - http://gerrit.ovirt.org/gitweb?p=vdsm.git
- **Mailing lists:**
  - vdsm-devel@lists.fedorahosted.org
  - vdsm-patches@lists.fedorahosted.org
- **IRC:**
  - #vdsm on Freenode
- **Core Team:**
  Dan Kenigsberg, Saggi Mizrahi, Federico Simoncelli, Igor Lvovsky, Eduardo Warszawasky, Ayal Baron