Virtual Machine Provisioning and Performance for Scientific Computing

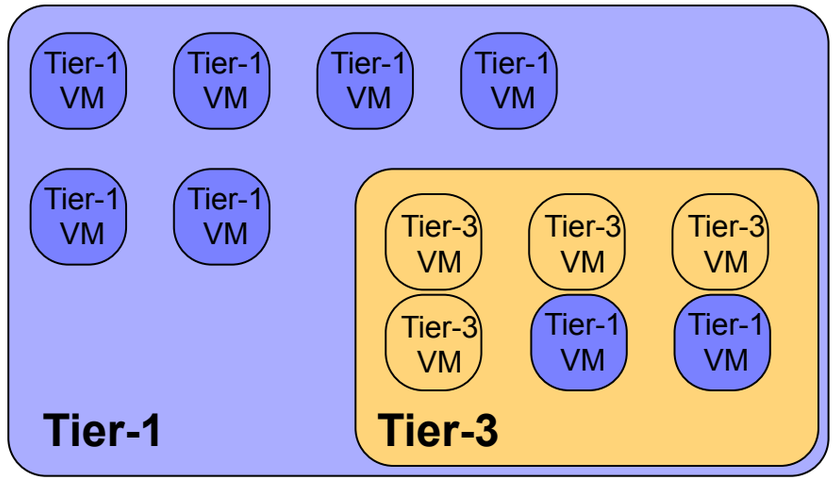
Davide Salomoni
INFN-CNAF

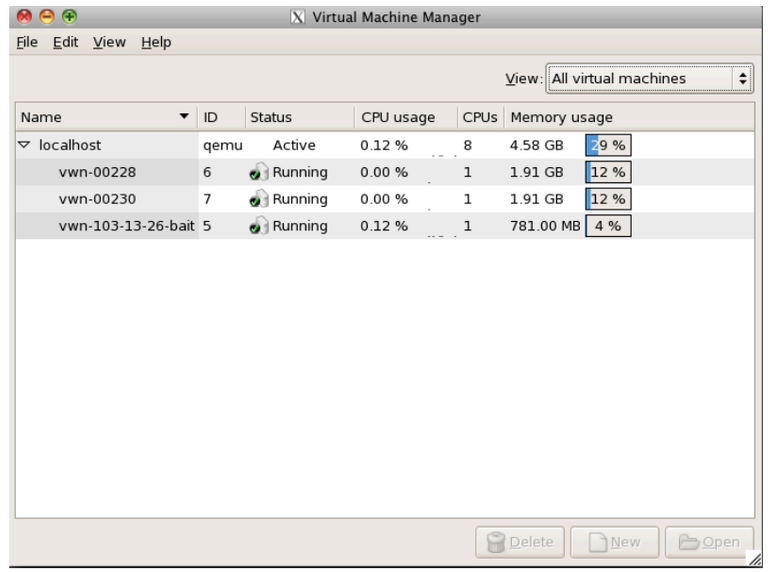
Outline

- WNoDeS Updates
- VM and Cloud Provisioning
- Performance Tests
- Network Virtualization

WNoDeS Updates (1)

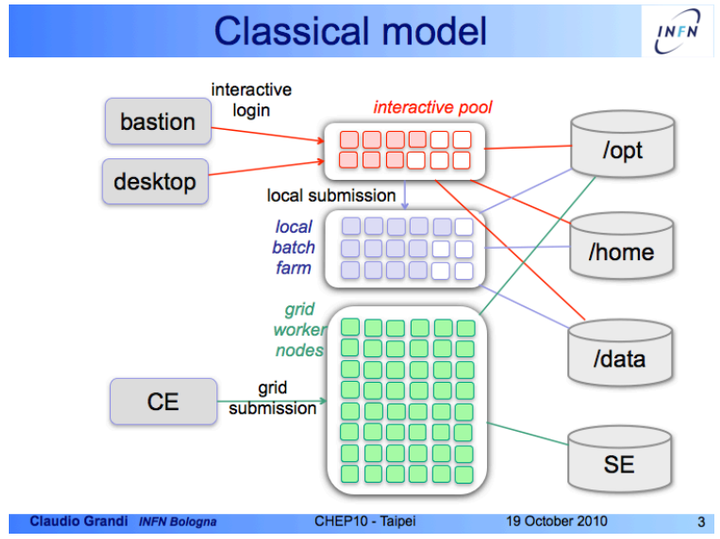
- More flexibility in VLAN usage
  - E.g., you can dedicate a VLAN to a certain customer, and request that VLAN to be instantiated on certain HVs only.
    - Will be used at CNAF to implement a “Tier-3” infrastructure, where Tier-3 users can only run on some hardware, which can be exploited by Tier-1 users as well, if Tier-3 users are not active.
    - Important developments may still be needed on Virtual Networking, see later.
- `libvirt` now used to manage and monitor VMs
  - Either locally of via a Web app (see later).
- Improved handling of VM images
  - Automatic purge of old VM images on HVs.
  - Tags can now be associated to images.
  - Download of VM images to HVs now via `http` or Posix I/O.
- **Job migration not needed anymore**
  - Enables WNoDeS to be ported to other LRMS.
- New way to support Cloud VMs
  - LRMS not needed anymore on Cloud VMs.

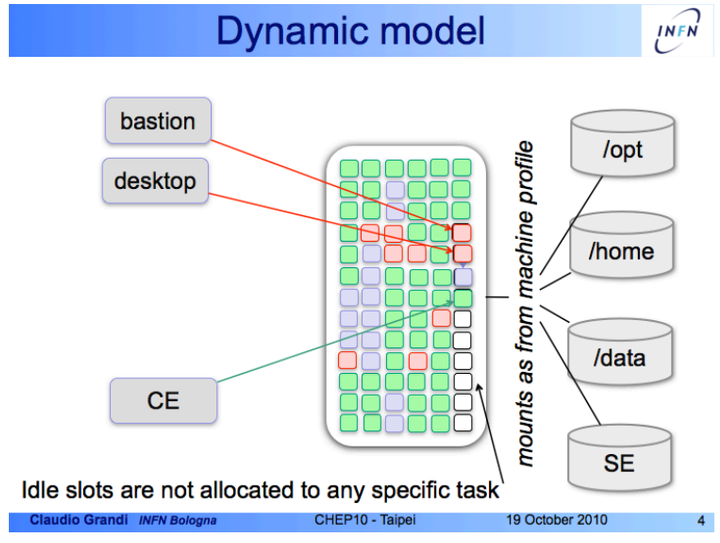




WNoDeS Updates (2)

- Support for LVM partitioning (see performance measurement)
  - Each VM will have its LVM partition; this improves performance and aids in ensuring flexibility (e.g. for Cloud requests) and VM isolation.
- Support for `sshfs` or `nfs` gateway see performance measurements)
- Command line tools to manage VM images
- New web applications for Cloud provisioning and for WNoDeS monitoring (see later)
- Virtual Interactive Pools (VIP)
  - Presented at CHEP'10
  - MD grant working on VIP, starting 4/2011
- Support for insertion of VM states into the DOCET database
- New plug-in architecture





Getting rid of `bmig` (job migration)

- `bmig` (the original way for WNoDeS to move jobs from bait to VM) is supported by LSF but not, for instance, by PBS/Torque
  - A serious limit to porting WNoDeS to other LRMS
- Currently testing a reservation-based alternative:
  - a bait asks its HV to prepare a VM according to the job requirements as usual; but then *the bait makes the pre-exec script intentionally fail* - this causes the job to be put back into its queue
  - The bait will also:
    - define a reservation request so that when requeued the running job will be forced to run on its designated VM only
    - attach this reservation to the requeued job
  - First tests with LSF successful
    - Need to do stress tests under heavy load
    - Note: the Job ID is unaffected (good)
  - A reservation-based mechanism also simplifies the interaction with the LRMS and puts less strain on it
- INFN Bari is going to test the same machinery with Torque/Maui
  - Thanks to G.Donvito, V.Spinoso

Outline

- WNoDeS Updates
- VM and Cloud Provisioning
- Performance Tests
- Network Virtualization

VM Provisioning

- WNoDeS allows full customizations of VMs
  - I.e., the parameters to define the VMs are all available to the system
  - Realistically, at the Tier-1, we have decided to characterize VMs according to a fixed set of parameters
    - This should answer most if not all of the practical request, while at the same time limiting entropy
    - The billing model has to be set up accordingly
- There is more to this
  - Definition of “custom images”
    - Through modifications of pre-defined image sandboxes and subsequent saving and retrieval of custom images
  - Storage: “dropbox-like” (easy), QoS-constrained (less easy)
- Golden rule: do not over-implement ahead of time (“premature optimization is the rule of all evil”, D.Knuth)

Specifically, for “Cloud” requests:

➡Small: 1 core, 1.7 GB RAM, 50 GB HD
➡Medium: 2 cores, 3.5 GB RAM, 100 GB RAM
➡Large: 4 cores, 7 GB RAM, 200 GB RAM
➡Extra-large: 8 cores, 14 GB RAM, 400 GB RAM

Current “Grid” VM images normally fall into the “Small” instance.

Two further options foreseen, initially for Grid and VIP jobs:

➡Whole-node, hard: all hardware cores, (1.7 * num. cores) GB RAM, (50 * num. cores) GB HD
➡Whole-node, soft: all available cores (*with a minimum*), (1.7 * num. cores) GB RAM, (50 * num. cores) GB HD

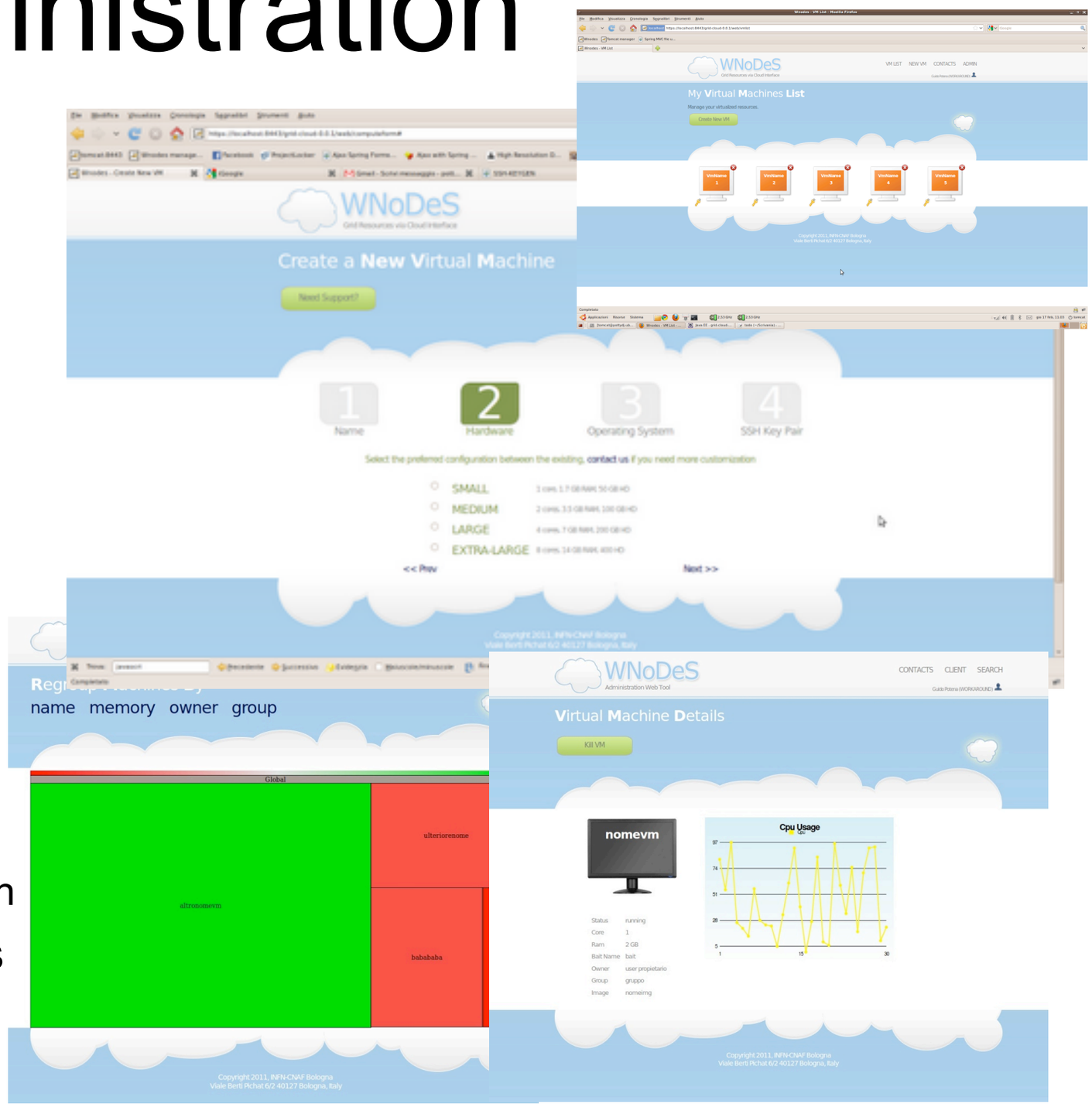
Note: network and distributed storage not considered above.

Cloud Provisioning

- The hard way: API-based (e.g., OCCI)
  - Not really meant for direct human consumption and therefore essentially never used directly (at least at the INFN Tier-1)
- More practically, via a web-based application
  - We don't need yet another portal, though
  - Need to converge around the general concept of "resource allocation & utilization"
    - Grid, Cloud, or else (i.e. - hopefully a single Grid- or Cloud-submission/allocation portal)
    - With integrated authentication and authorization
      - Several possibilities here - we'd much like to re-use what we already have, though; namely, VOMS and the Argus Authorization Service
    - Plenty of room for collaboration with IGI

Cloud Provisioning and WNoDeS Administration

- Two MD thesis on this, to be discussed soon
- VOMS/Argus partially integrated into the Cloud portal
- Selection of Cloud instances according to a few pre-defined configurations
  - ssh key pair to access the allocated VMs
  - Possibility to instantiate multiple VMs at once
- Treemap-based representation of running VMs for admin purposes
  - Plus details on CPU and I/O utilization
- As mentioned, plan to integrate this into a more general “resource access portal”

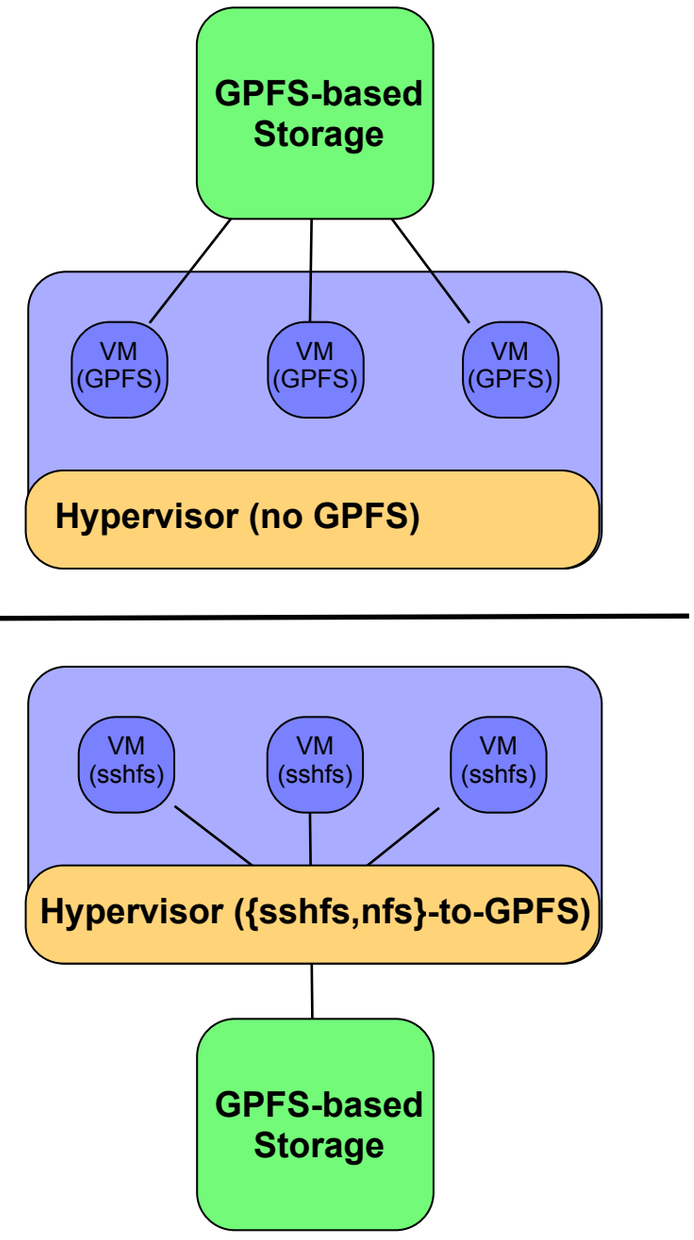


Outline

- WNoDeS Updates
- VM and Cloud Provisioning
- Performance Tests
- Network Virtualization

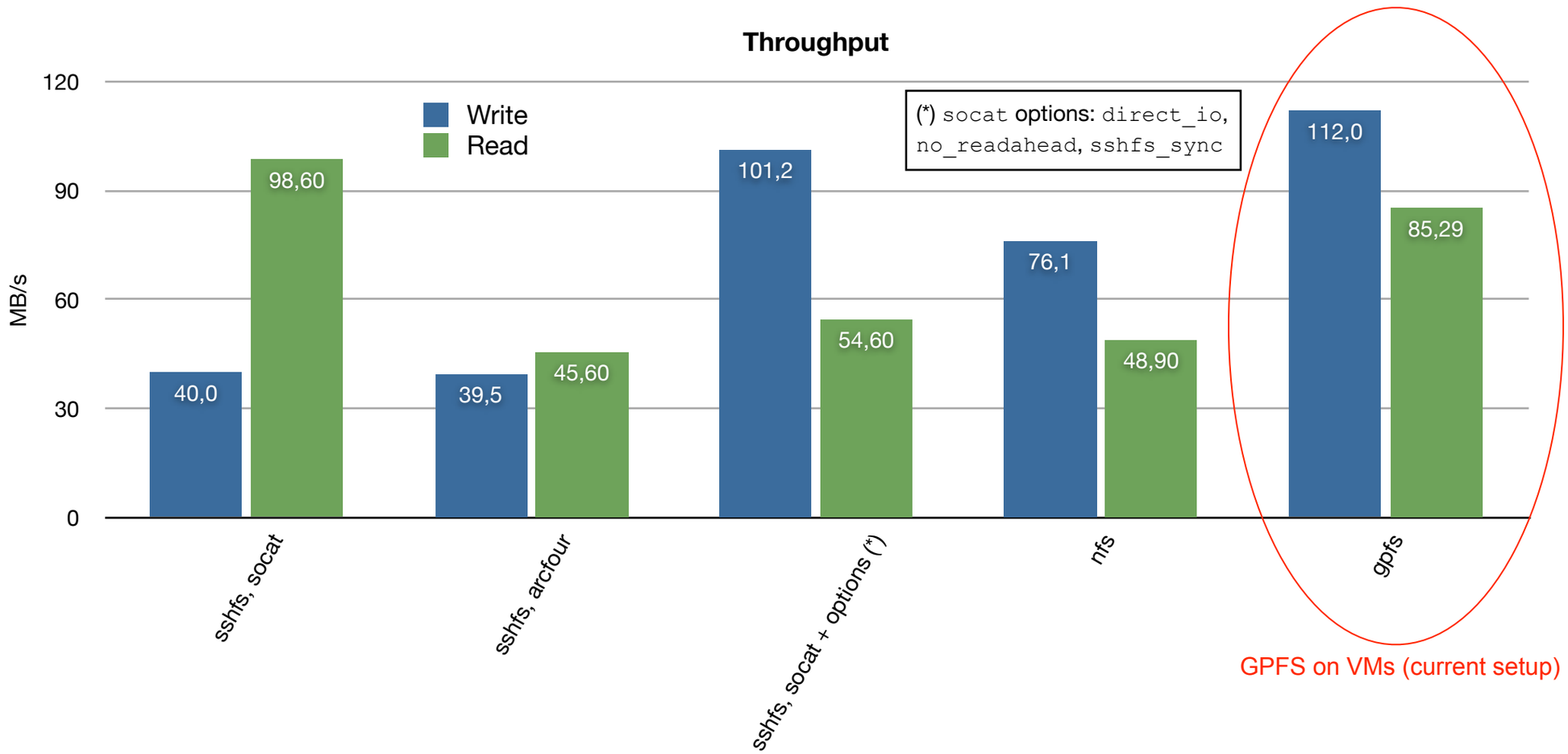
Performance test of alternatives to mounting GPFS on VMs

- The issue (not strictly GPFS-specific) is that any CPU core may become a GPFS (or any other distributed FS) client. This leads to GPFS clusters of several thousands of nodes
  - This is *large*, even according to IBM, requires special care and tuning, and may impact performance and functionality of the cluster
  - We investigated two alternatives to this, both assuming that an HV would distributed data to its VMs
    - `sshfs`, a FUSE-based solution
    - a GPFS-to-NFS export

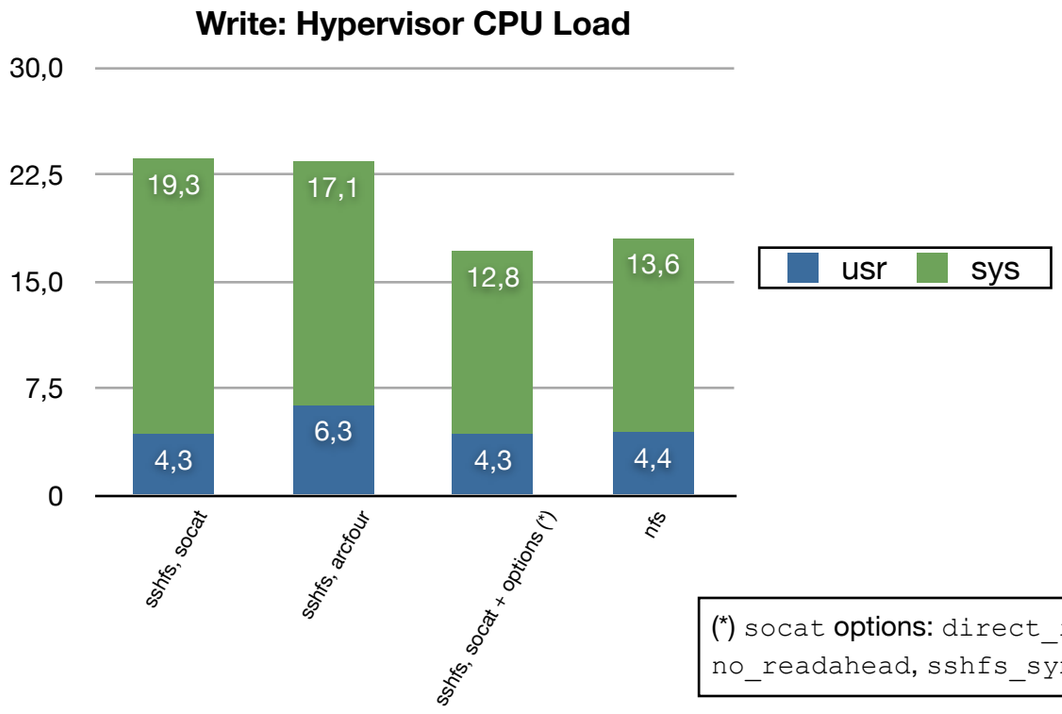


`sshfs` vs. `nfs`: throughput

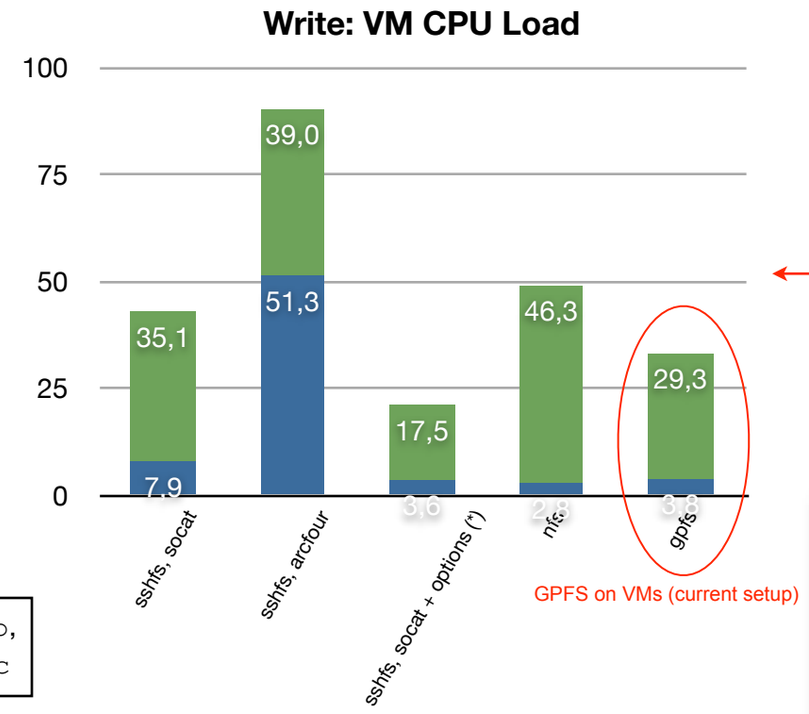
- `sshfs` throughput constrained by encryption (even with the lowest possible encryption level)
- Marked improvement (throughput better than `nfs`) using `sshfs` with *no encryption* through `socat`, esp. with some tuning
  - File permissions are not straightforward with `socat`, though - complications with e.g. `glexec`-based mechanisms



sshfs vs. nfs: CPU usage

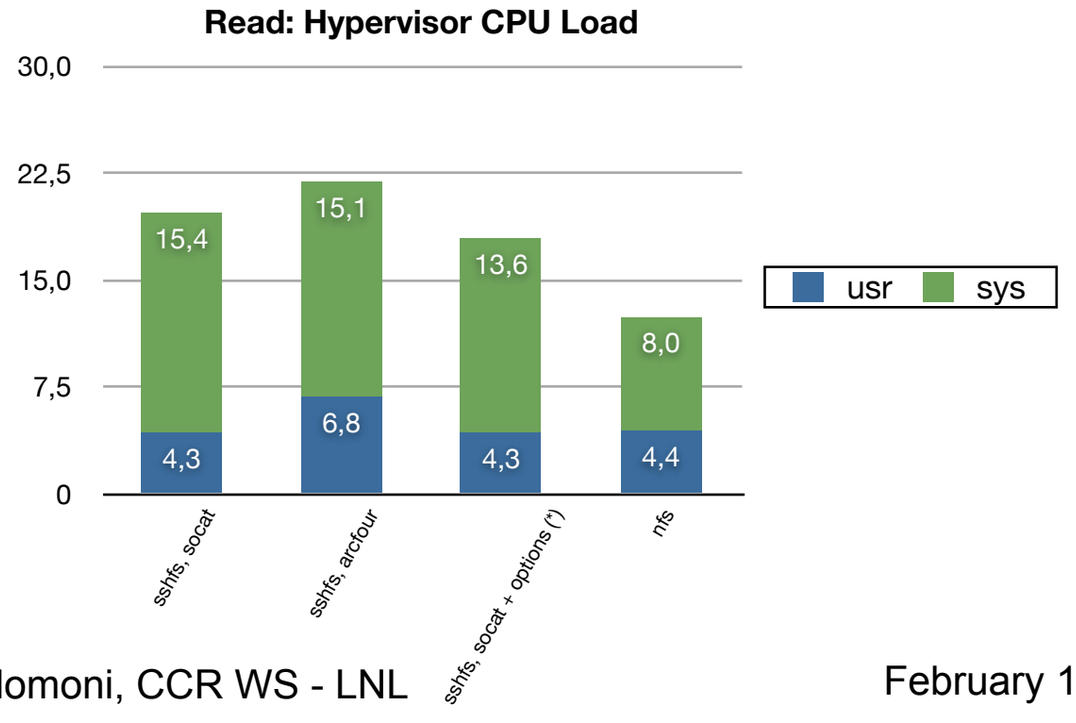


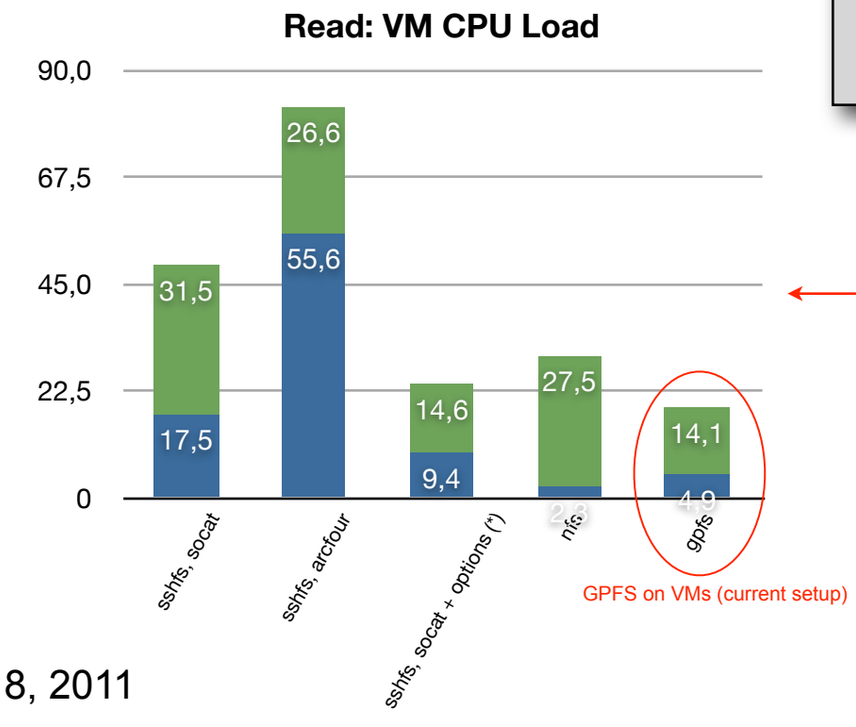
(*) `socat` options: `direct_io`, `no_readahead`, `sshfs_sync`



Write

Overall, `socat`-based `sshfs` w/ appropriate options seems the best performer





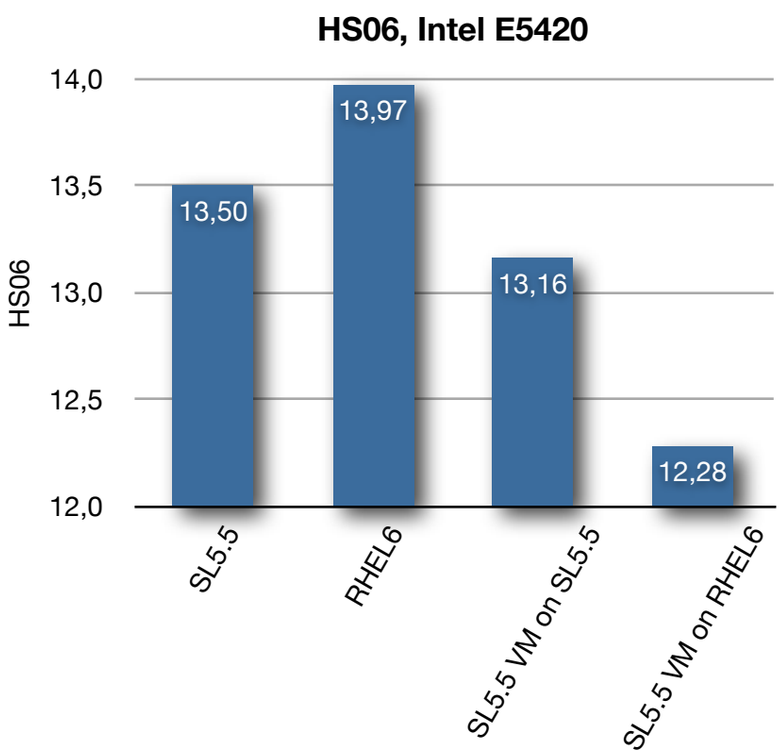
Read

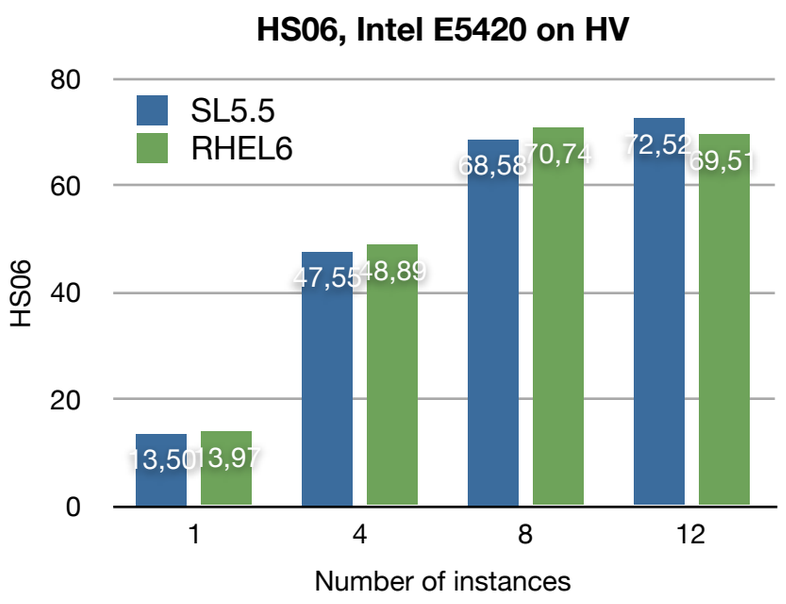
VM-related Performance Tests

- All tests: since SL6 was not available yet, we used RHEL 6
- Classic HEP-Spec06 for CPU performance
- `iozone` to test local I/O
- Network I/O not shown here
  - `virtio-net` has already been proven to be quite efficient (90% or more of wire speed)
- Local I/O has historically been a problem for VMs
  - WNoDeS not an exception, esp. due to its use of the KVM `-snapshot` flag
  - The new WNoDeS release will still use `-snapshot`, but for the root partition only; `/tmp` and local user data will reside on a (host-based) LVM partition
- Several things are improving in the I/O area, though
  - KVM-specifics: page sharing (KSM), Transparent Huge Pages (test ongoing)
  - Plus network-related optimizations not shown here, namely vhost-net, SR-IOV and vmchannel
- Note: in our performance test, we disabled (or at least tried to disable) caching

HS06 on Hypervisors and VMs (Intel E5420)

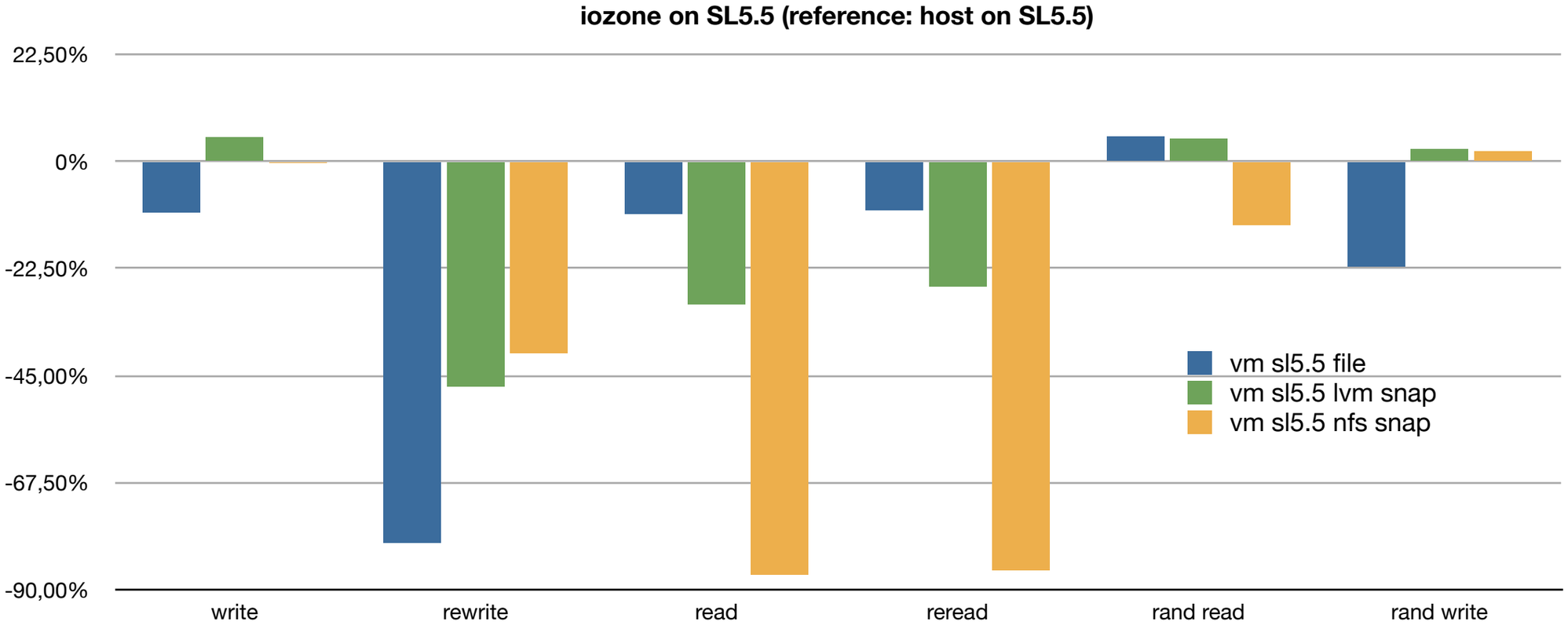
- Slight performance increase of RHEL6 vs. SL5.5 on the hypervisor
  - Around +3% (exception made for 12 instances: -4%)
- Performance penalty of SL5.5 VMs on SL5.5 HV: -2.5%
- Unexpected performance loss of SL5.5 VMs on RHEL6 vs. SL5.5 HV (-7%)
  - Test to be completed with multiple VMs





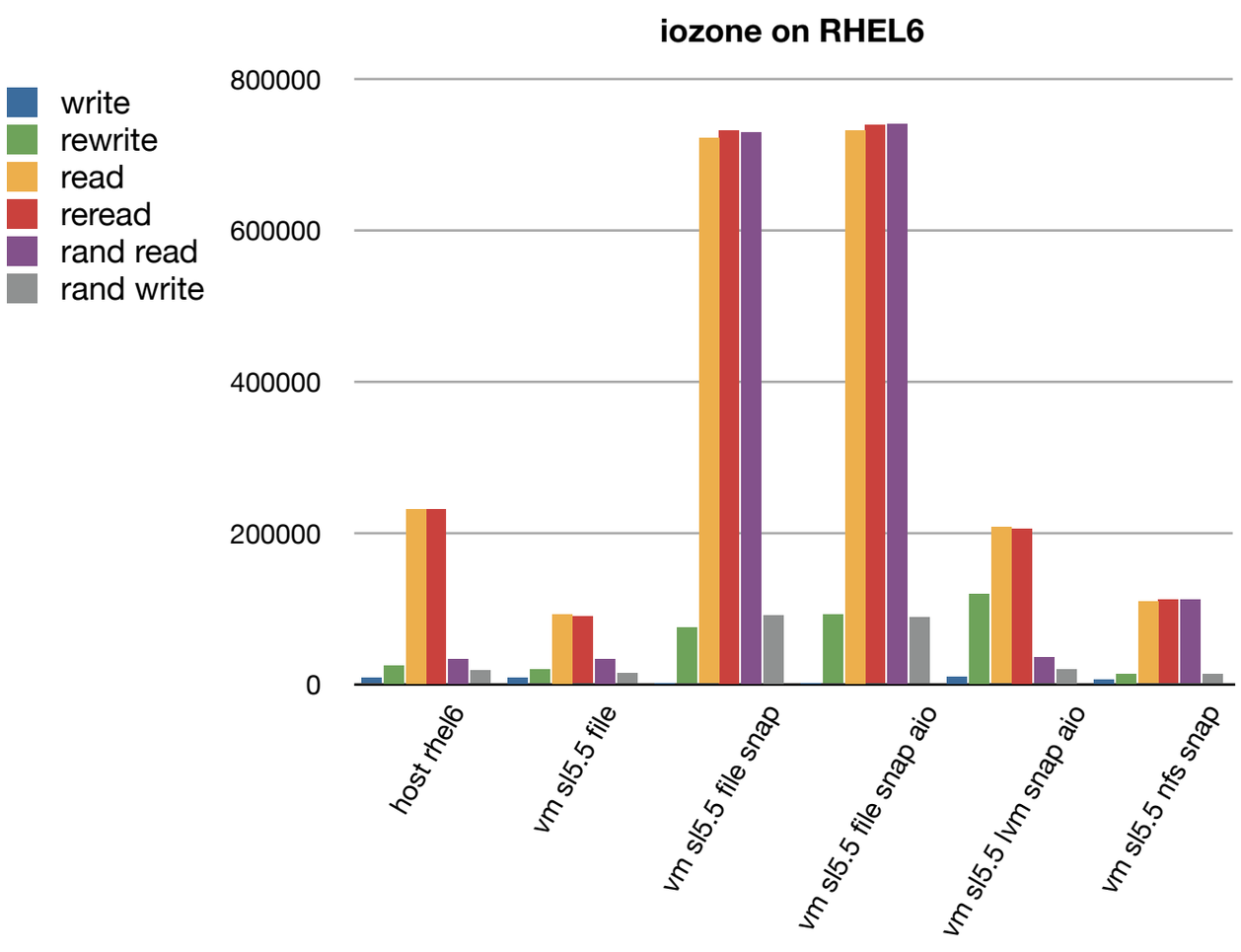
`iozone` on SL5.5 (SL5.5 VMs)

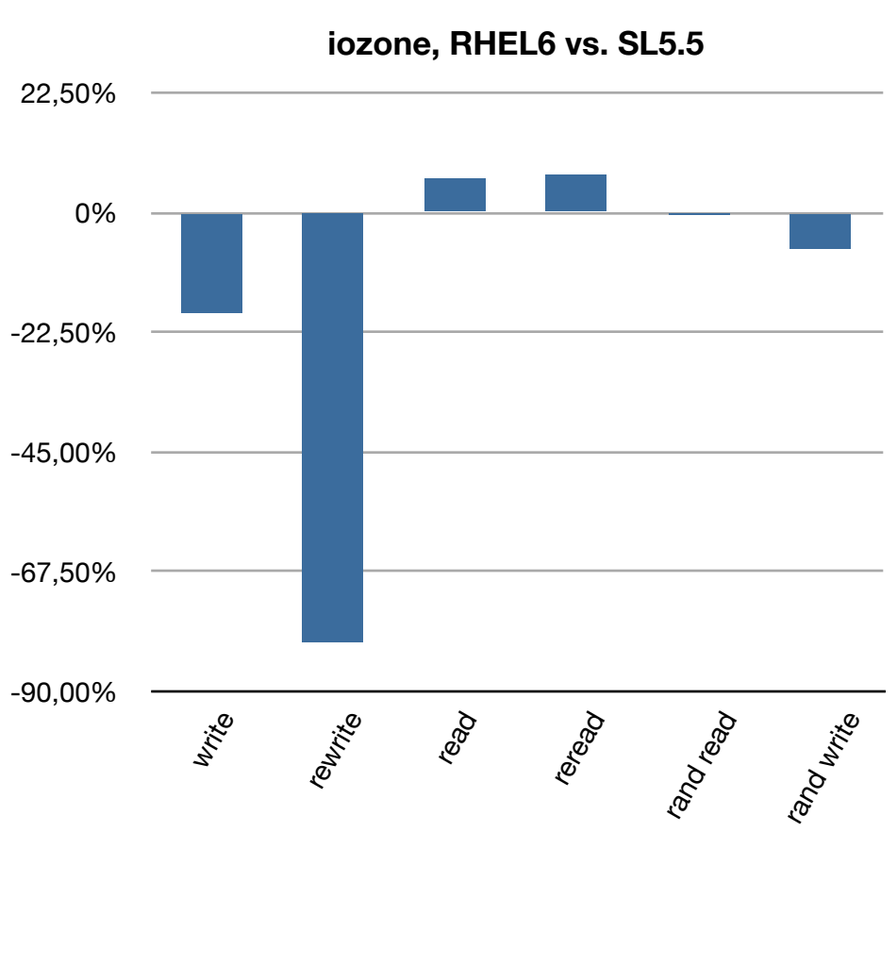
- `iozone` tests with caching disabled, file size 4 GB on VMs
- host with SL5.5 taken as reference
- VM on SL5.5 with just `-snapshot` crashed
- Based on these tests, WNoDeS will support `-snapshot` for the root partition and a native LVM partition for `/tmp` and user data
  - A per-VM single file or partition would generally perform better, but then we'd practically lose VM instantiation dynamism



`iozone` on RHEL6 (SL5.5 VMs)

- Consistently with what was seen with some CPU performance tests, `iozone` on RHEL6 surprisingly performs often worse than on SL5.5
  - RHEL6 supports native AIO and `preadv/pwritev`: group together memory areas before reading or writing them. This is maybe the reason for some funny results (unbelievably good performance) of the `iozone` benchmark.
- Assuming RHEL6 performance will be improved by RH, using VM with `-snapshot` for the root partition and a native LVM patition for `/tmp` and user data in WNoDes seems a good choice here as well
  - But we will not upgrade HVs to RHEL6/SL6 until we are able to get reasonable results in this area





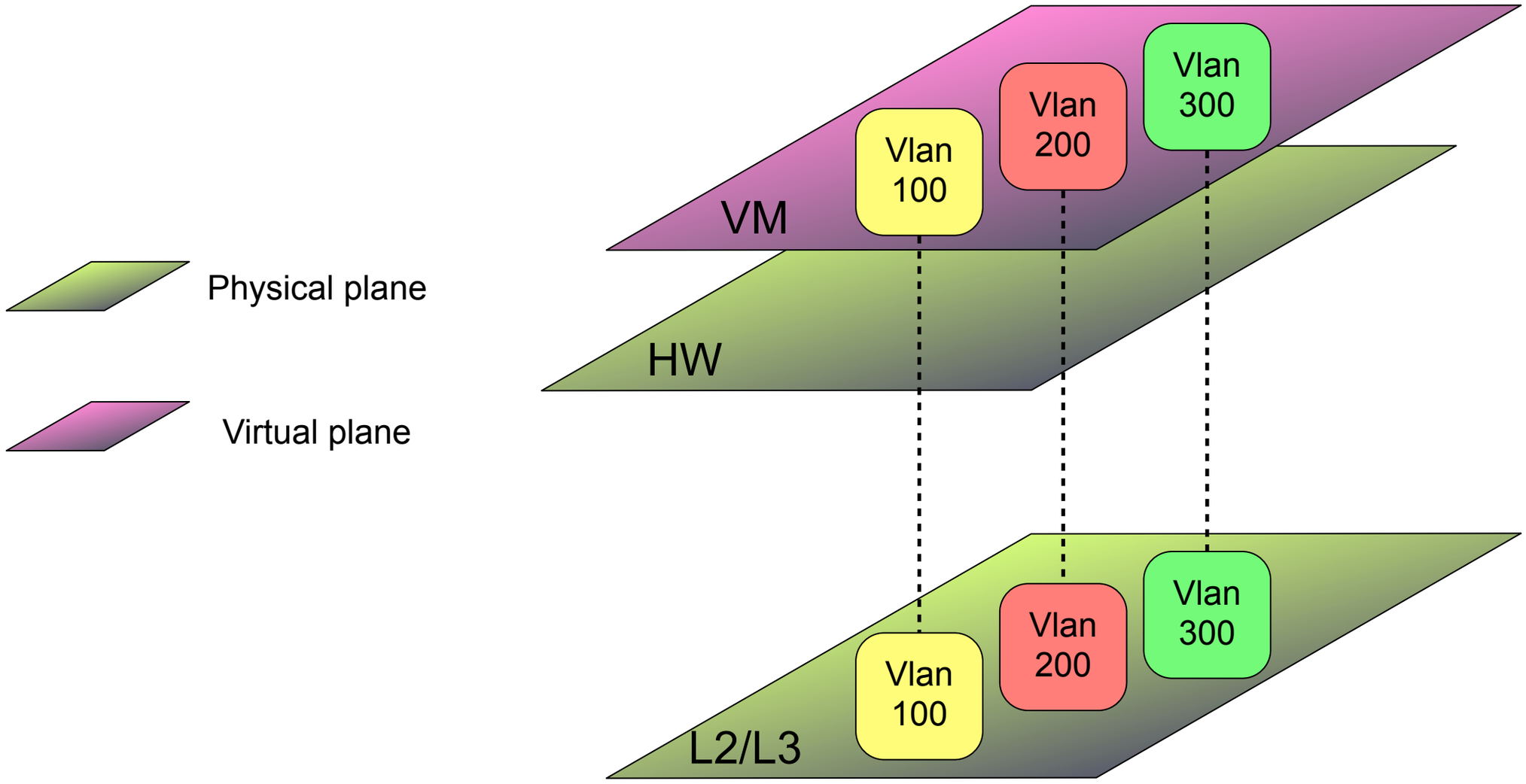
Outline

- WNoDeS Updates
- VM and Cloud Provisioning
- Performance Tests
- Network Virtualization

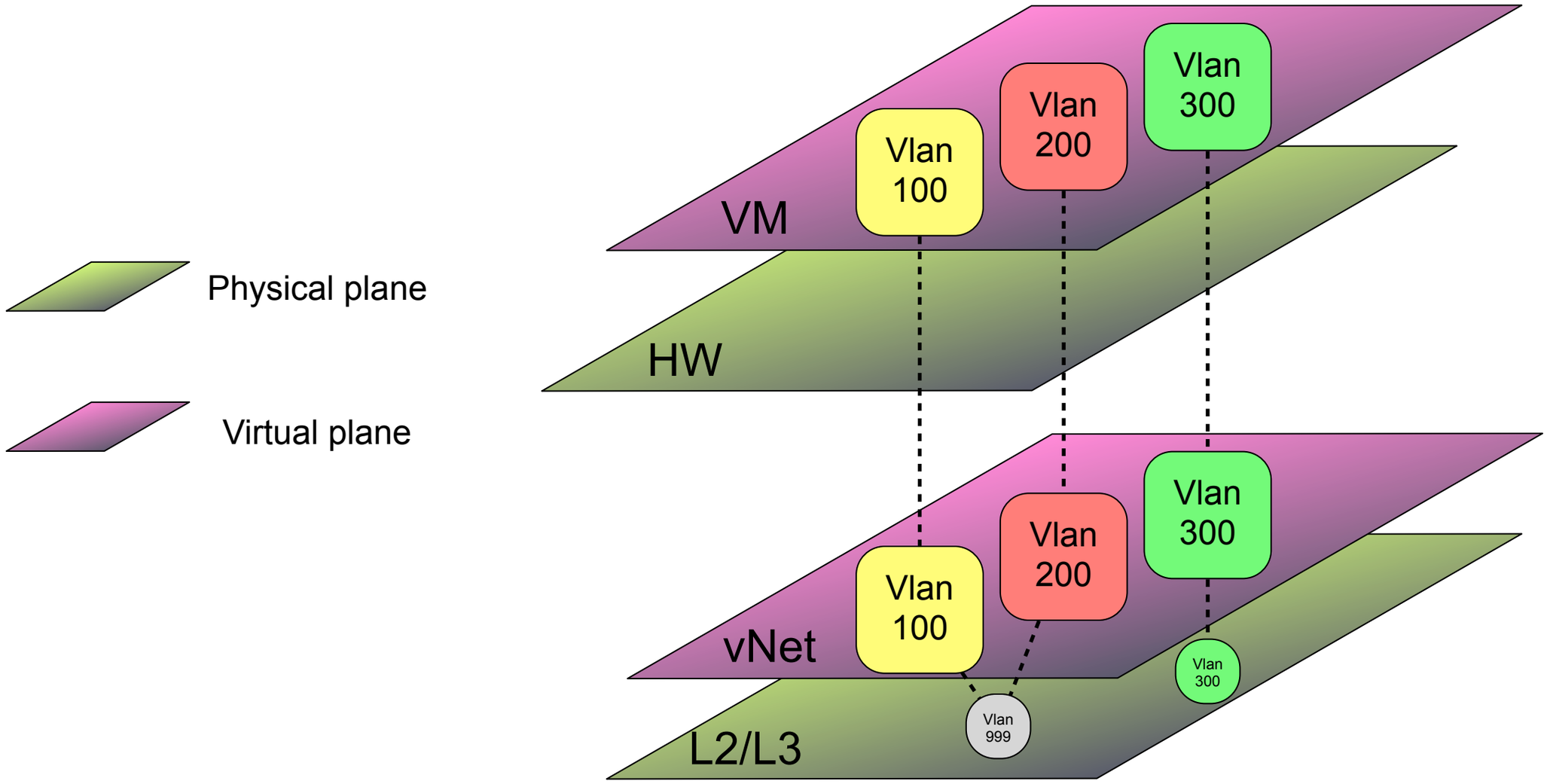
A Missing Step: Network Virtualization

- Server virtualization has progressed steadily in the past years
- Network virtualization much less so
  - Enterprise networks are often static, locked down, proprietary, complex
- Key missing features:
  - inter-VM traffic analysis
    - Support for NetFlow, sFlow, SPAN or OpenFlow
  - interface rate limiting, per-port QoS policies (→ per-flow management)
  - per-customer VLANs
    - Note: normally up to 4096 VLANs - cf. proposals like RFC5517 or IEEE 802.1ad/802.1ah
    - With private/public IP address assignment
  - dynamic reconfiguration of the network state per-VM or per-VM group
- **Network state should become a property of the virtual interface**

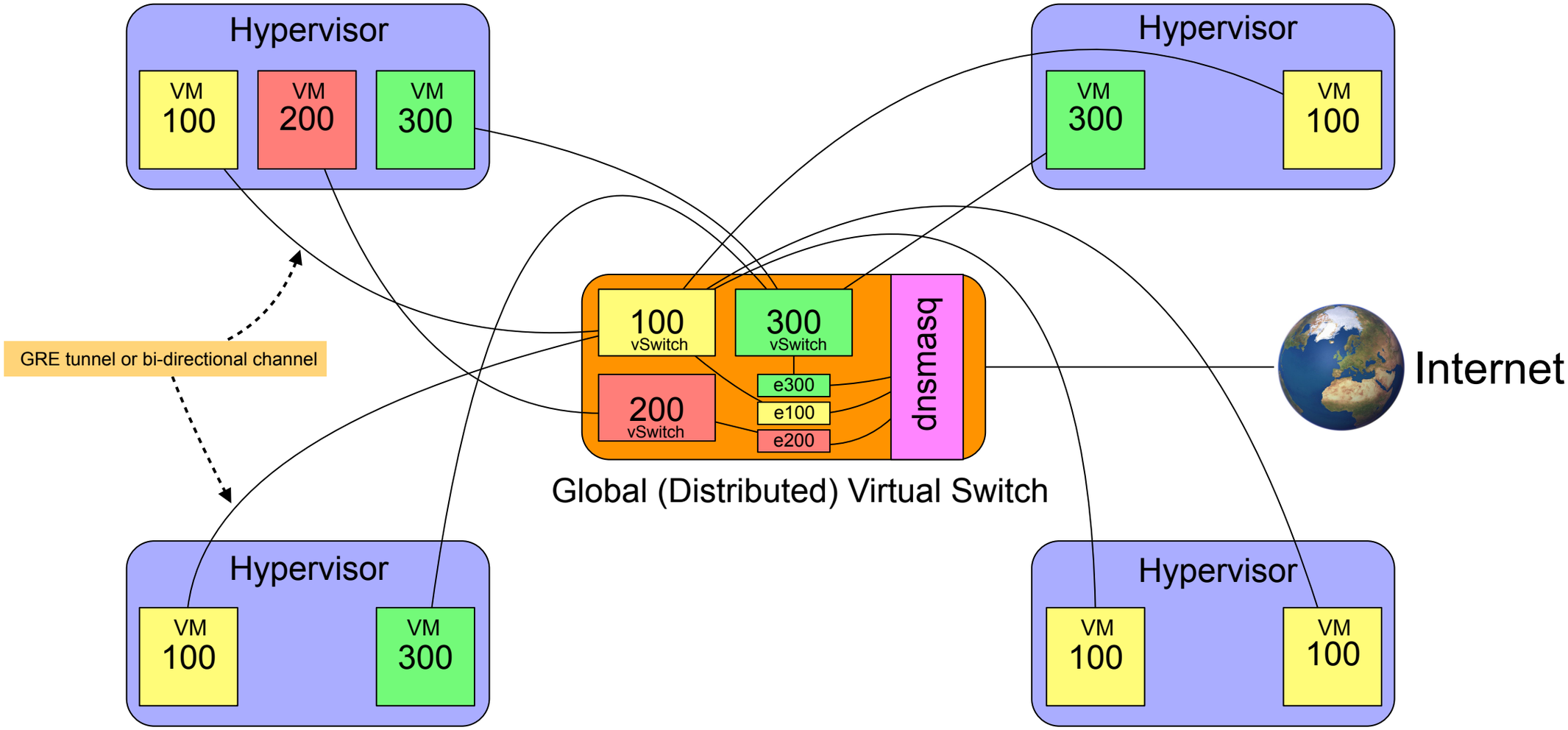
Virtual Plane, VM



Virtual Plane, VM + Network

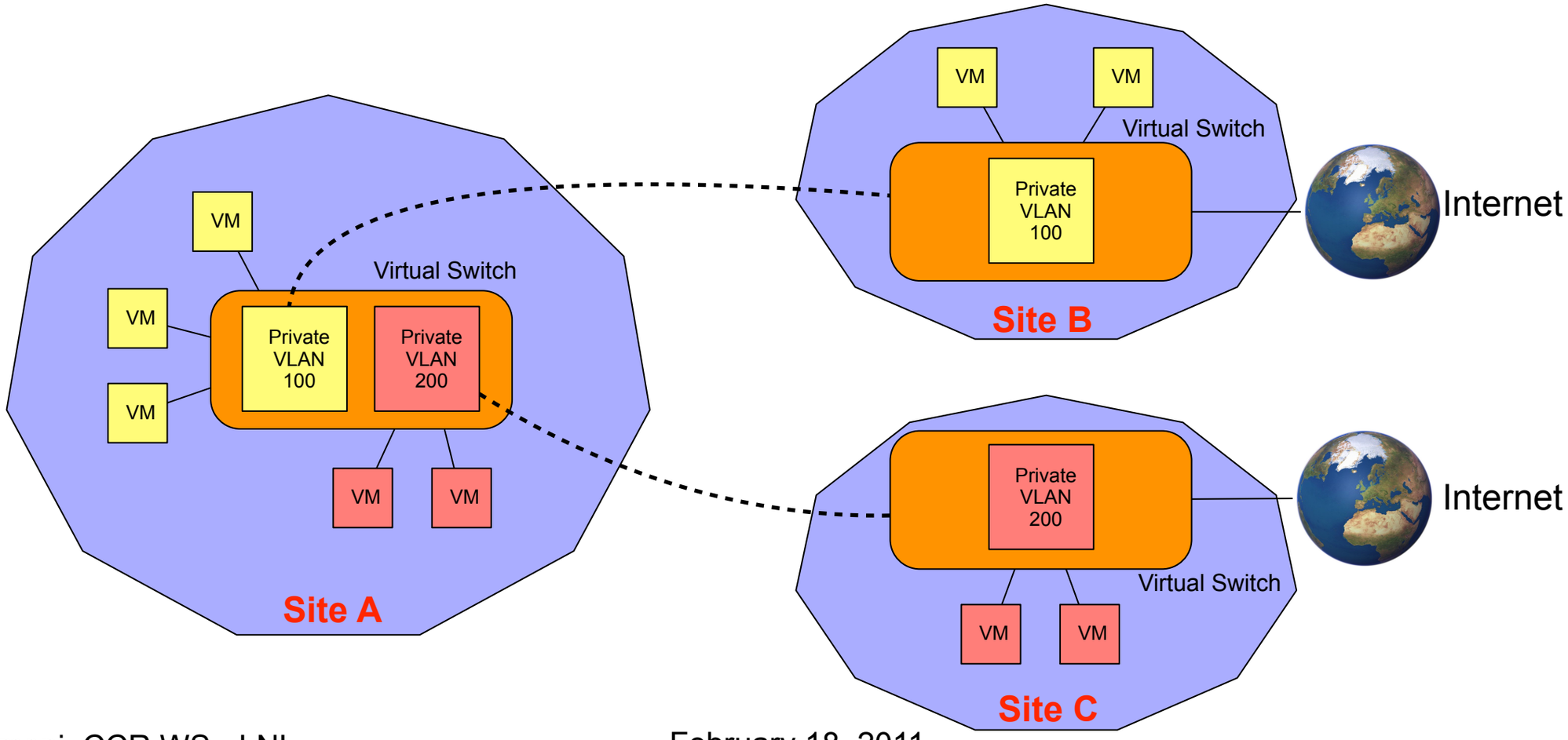


Virtual Network Topology



Virtual Network Topology for Multiple Centers

- Since the overlay network is served by a virtual switch, nothing prevents to *dynamically* extend the overlay network to multiple centers
  - E.g. to transparently connect remote resources and make them available for instance for flash requests
  - QoS considerations will play an important role
    - May integrate with more network-centric initiatives like FEDERICA/2
- MD thesis on WNoDeS dynamic virtual networking starting 4/2011



Conclusions

- **WNoDeS** is evolving, thanks to the experience gained at the Tier-1
  - Installed and running also here (LNL Tier-2) - thanks esp. to G.Maron, M.Biasotto, A.Crescente. (anybody interested in trying it out is welcome)
  - An important goal is to have it running on LRMS other than LSF
  - Interactions with distributed file systems in large clusters may be complicated (not really a WNoDeS-specific issue)
  - The flexibility of the system is being exploited e.g. by the VIP interface
- **VM provisioning** can have many degrees of freedom, and for the user point of view should really be integrated into a coherent Grid/Cloud portal
- **Performance** testing is always interesting, and we are getting better as we understand old and new knobs
  - VM (CPU, I/O) performance tuning and testing may be quite different from similar conventional activities.
  - Several related things still need to be done (e.g. VM pinning, VM brokerage)
- **Network virtualization** is perhaps not a big issue with conventional ("Grid") resource usage, but becomes essential with Cloud-related assignments
  - Still quite an R&D area; good opportunities for collaboration with Network providers
- The difficulty is not so much in *virtualizing (even a large number of) resources*. It is much more in having a scalable, extensible, efficient, integrated (with storage, grid, local, cloud interfaces) system.

Thanks

- A. Chierici: VM performance tuning and test
- A.K. Calabrese: `sshfs` vs. `nfs` vs. GPFS performance tuning and test
- A. Italiano, G. Dalla Torre: WNoDeS core
- G. Potena, L. Cestari, D. Andreotti: WNoDeS Cloud interface and Web apps
- C. Grandi: VIP testing

- wnodes@lists.infn.it, http://web.infn.it/wnodes