CernVM
SoFTware Appliance

The CernVM Project

A new approach to software distribution

Carlos Aguado Sanchez@cern.ch
Jakob Blomer@cern.ch
Predrag Buncic@cern.ch

CernVM
SoFTware Appliance

Outline

- Motivation
- Mission
- Requirements
- Service architecture
- Conclusions

CernVM
SoFTware Appliance

Outline

- Motivation
- Mission
- Requirements
- Service architecture
- Conclusions

CernVM
SoFTware Appliance

Motivation

- LHC experiment frameworks are a complex world
  - Heavy, multiplatform (arch+OS+compiler)
  - Grid and development environments
- Industry trend to Multi/Many-Core CPU architectures
- Demand for scalable, parallel applications (simulation, reconstruction, analysis)
  - Large architectural impact, different programming model

CernVM
SoFTware Appliance

Outline

- Motivation
- Mission
- Requirements
- Service architecture
- Conclusions

CernVM
SoFTware Appliance

Project mission

- Portable Analysis Environment using Virtualization Technology (WP9)
  - Approved in 2007 (2+2 years) as R&D activity in CERN/PH Department
- Project goals:
  - Provide a complete, portable and easy to configure user environment for developing and running LHC data analysis
  - Reduce effort to install, maintain and keep up to date the experiment software
  - Decouple application lifecycle from evolution of system infrastructure
  - Lower the cost of software development by reducing the number of platforms

CernVM
SoFTware Appliance

Constraints

- This is a proof of concept: preproduction environment
  - Oriented to create community
  - Not a final production service, but worldwide
- Economic
  - Modest budget for external support (tools) and HW (during all project long)
    - Netapps, SAN clusters, Barracuda, Inifiband, etc.
    - RightScale, VSA, etc.

CernVM
SoFTware Appliance

Objectives

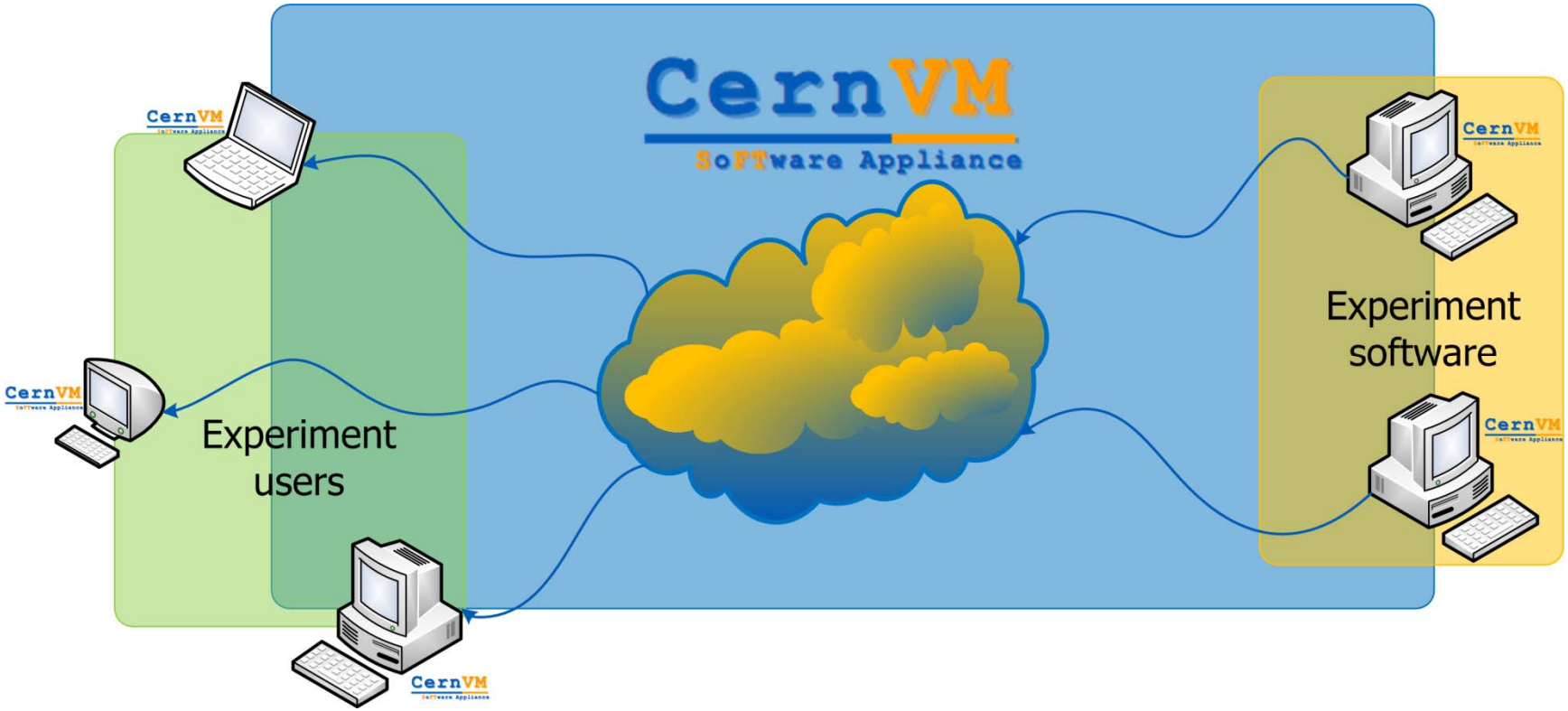
- CernVM: versatile virtual platform
  - Baseline virtual machine: CernVM Virtual Software Appliance
    - Management of the baseline image for all experiments with possible further customization
  - Software Delivery Network: the CernVM File System (CVMFS)
    - HTTP compliant

CernVM
SoFTware Appliance

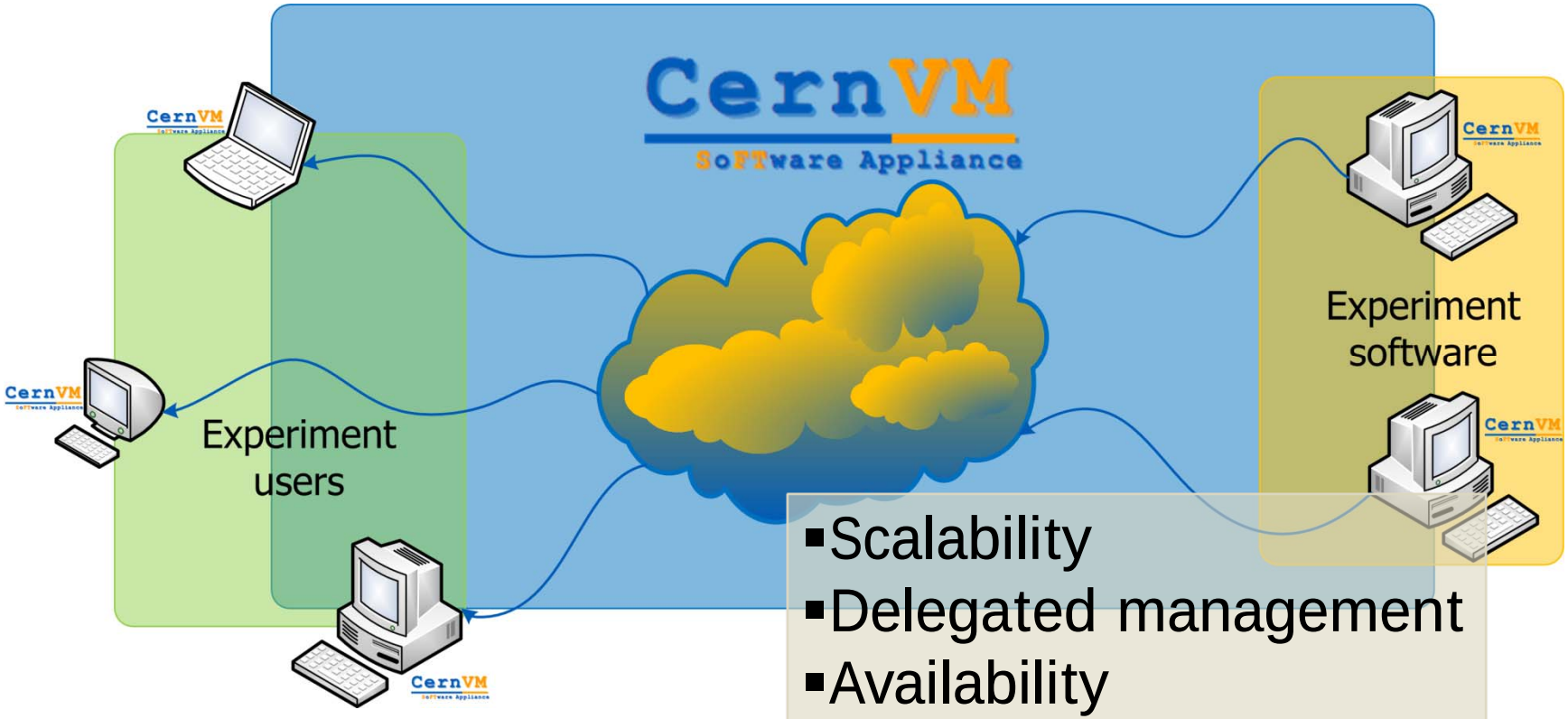
Outline

- Motivation
- Mission
- Requirements
- Service architecture
- Conclusions

CernVM App Store



CernVM App Store



CernVM
SoFTware Appliance

Requirements

- Ability to provide:
  - Virtual Appliance Configuration Management
  - Uniform and persistent URL namespace
    - With a large HTTP-repository behind
  - All services sandboxed as appliances
    - Deployment and management interface

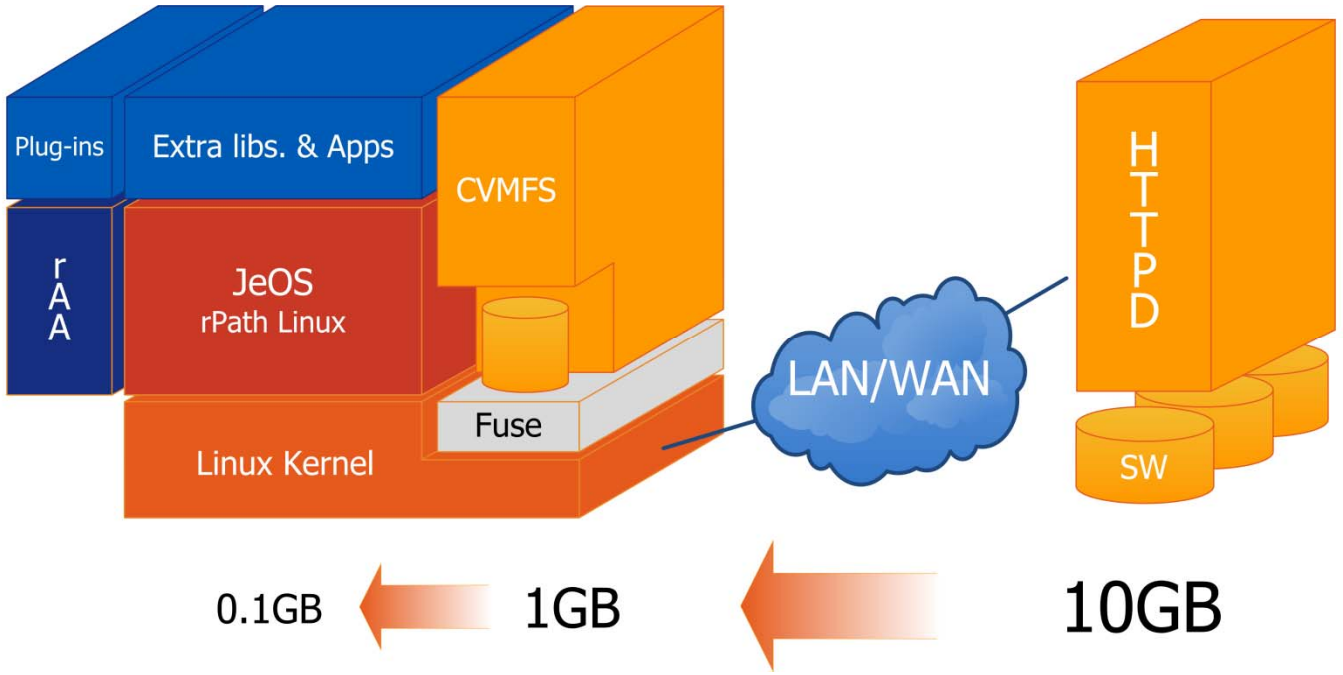
CernVM
SoFTware Appliance

Outline

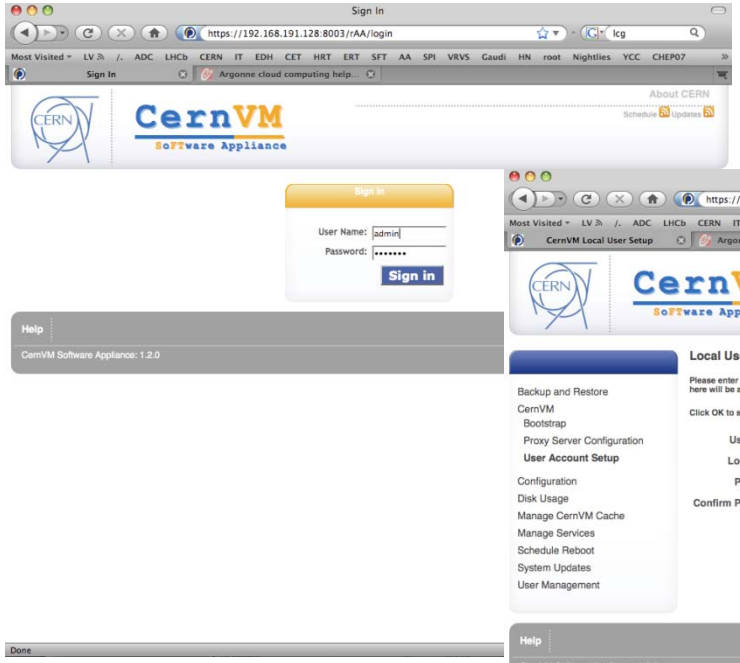
- Motivation
- Mission
- Requirements
- Service architecture
- Conclusions

CernVM
SoFTware Appliance

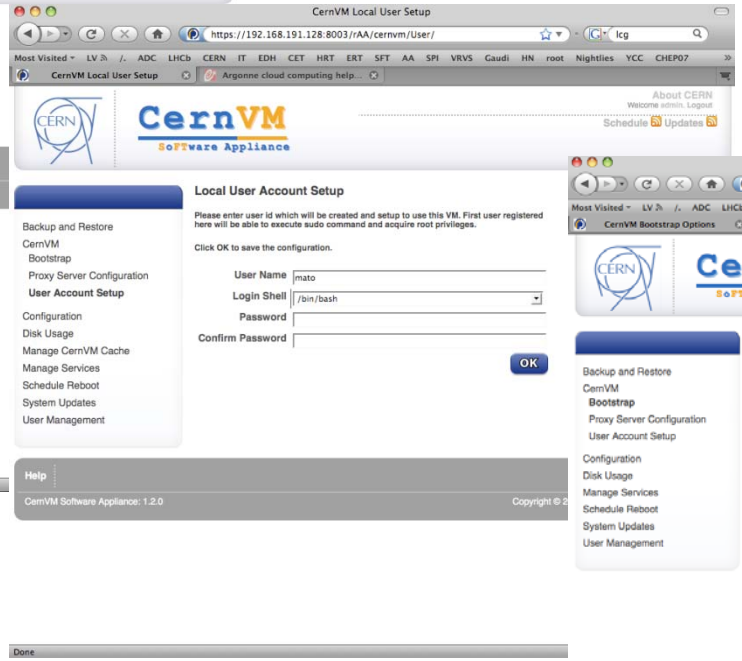
Application model



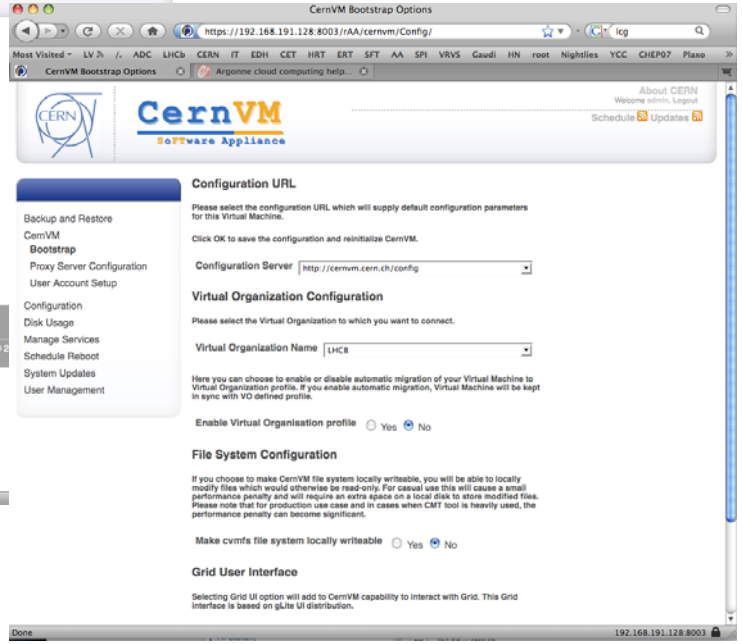
As easy as 1,2,3



1. Login to Web interface

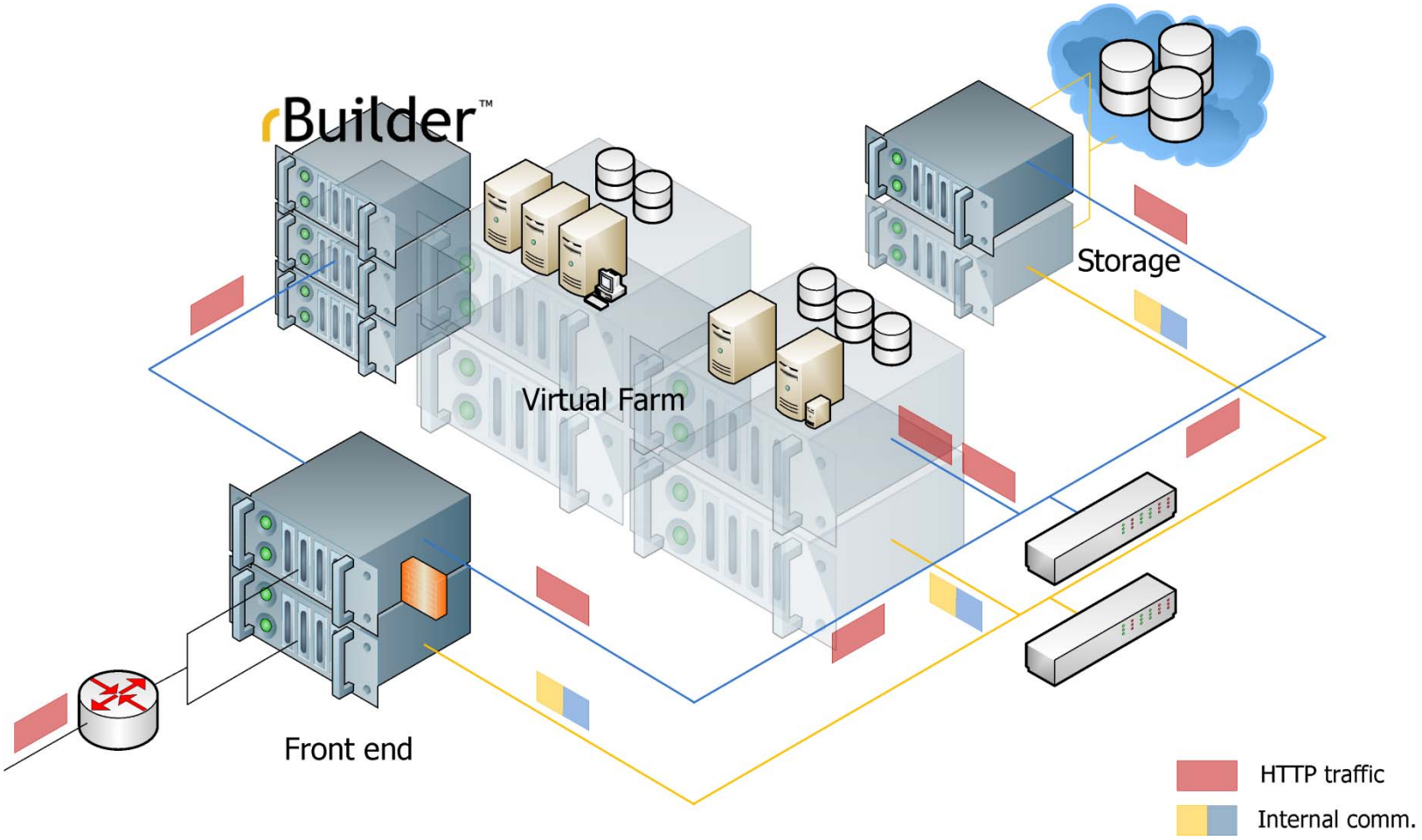


2. Create user account

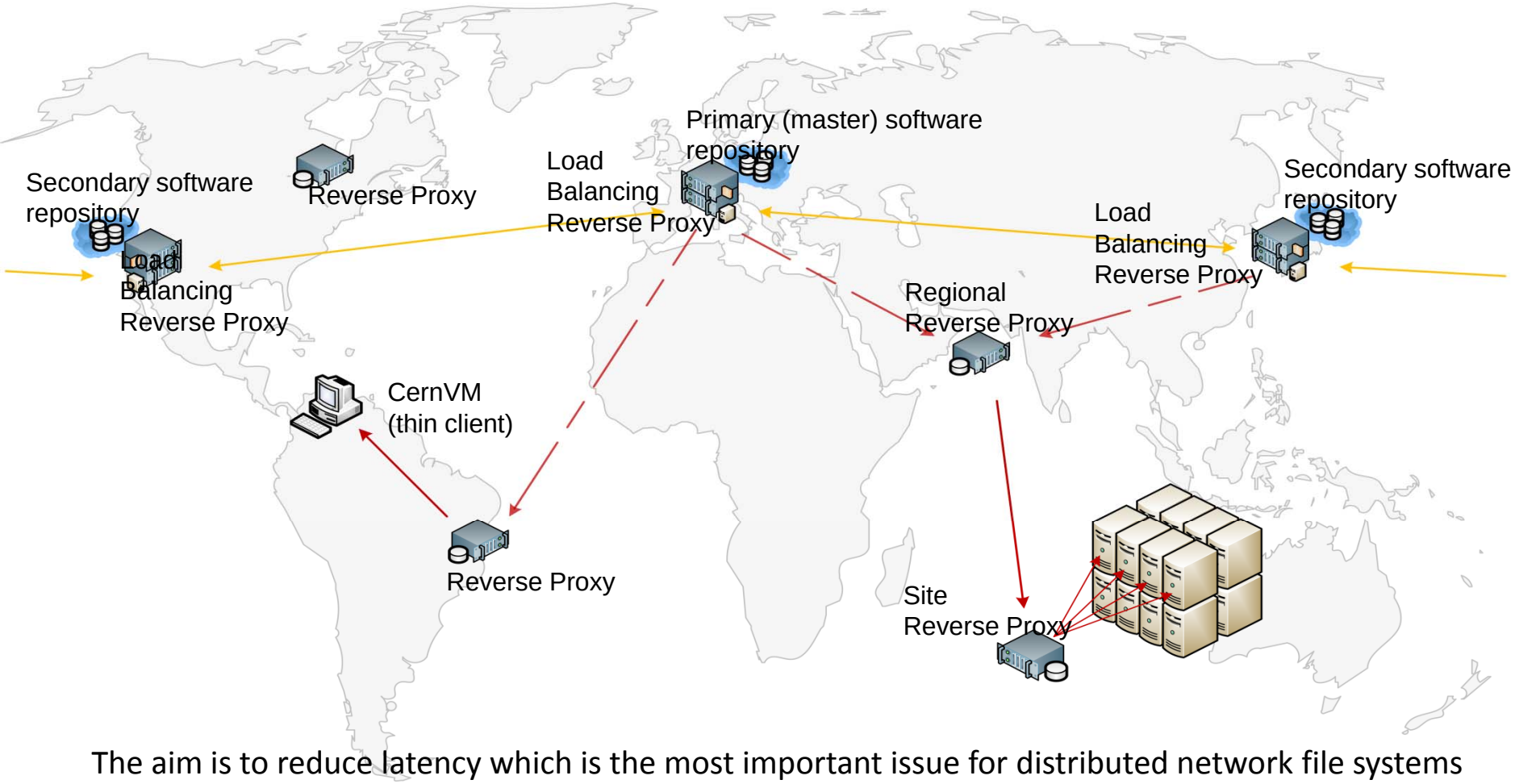


3. Select experiment, appliance flavor and preferences

Service architecture I



Service architecture II

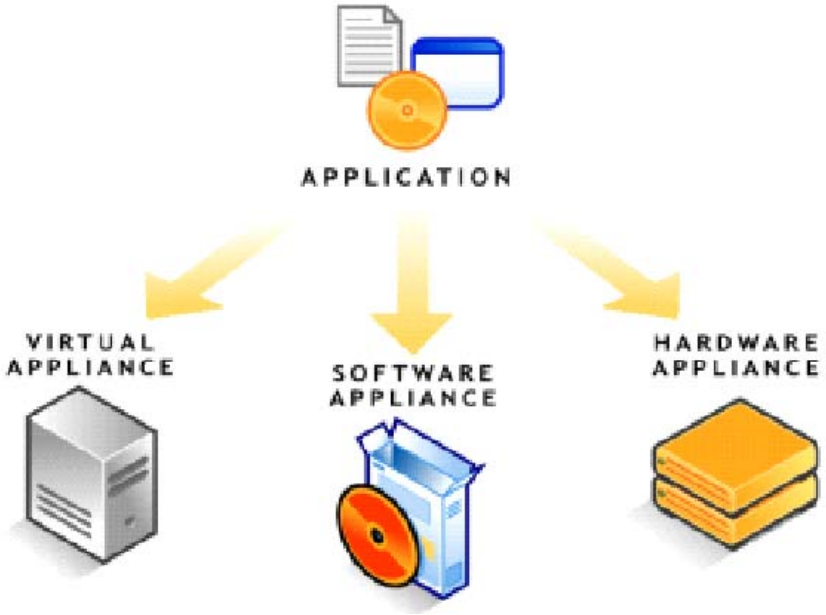


The aim is to reduce latency which is the most important issue for distributed network file systems

CernVM
SoFTware Appliance

rBuilder™ service

- rPath philosophy:
  - JeOS
  - Transactional model for software deployment (Conary)
  - Simple user interface (rAA)



CernVM
SoFTware Appliance

rBuilder™ service

- rPath philosophy:
  - JeOS
  - Transactional model for software deployment (Conary)
  - Simple user interface (rAA)

Build types

- Installable CD/DVD
- Stub Image
- Raw Filesystem Image
- Netboot Image
- Compressed Tar File
- Demo CD/DVD (Live CD/DVD)
- Raw Hard Disk Image
- Vmware ® Virtual Appliance
- Vmware ® ESX Server Virtual Appliance
- Microsoft ® VHD Virtual Apliance
- Xen Enterprise Virtual Appliance
- Virtual Iron Virtual Appliance
- Parallels Virtual Appliance
- Amazon Machine Image
- Update CD/DVD
- Appliance Installable ISO

CernVM
SoFTware Appliance

Conary Package Manager

```
class Root(CPackageRecipe):
    name='root'
    version='5.19.02'

    buildRequires = ['libpng:devel',
                     'libpng:devellib','krb5:devel',
                     'libstdc++:devel','libxml2:devel',
                     'openssl:devel','python:devel',
                     'xorg-x11:devel', 'zlib:devel',
                     'perl:devel', 'perl:runtime']

    def setup(r):
        r.addArchive('ftp://root.cern.ch/root/%(name)s v%(version)s.source.tar.gz')
        r.Environment('ROOTSYS',%(builddir)s')
        r.ManualConfigure('--prefix=/opt/root ')
        r.Make()
        r.MakeInstall()
```

CernVM
SoFTware Appliance

CDN: the URL

`scheme://hostname:port/path?query#fragment`

- scheme: http | https
- hostname:

  cernvm.cern.ch

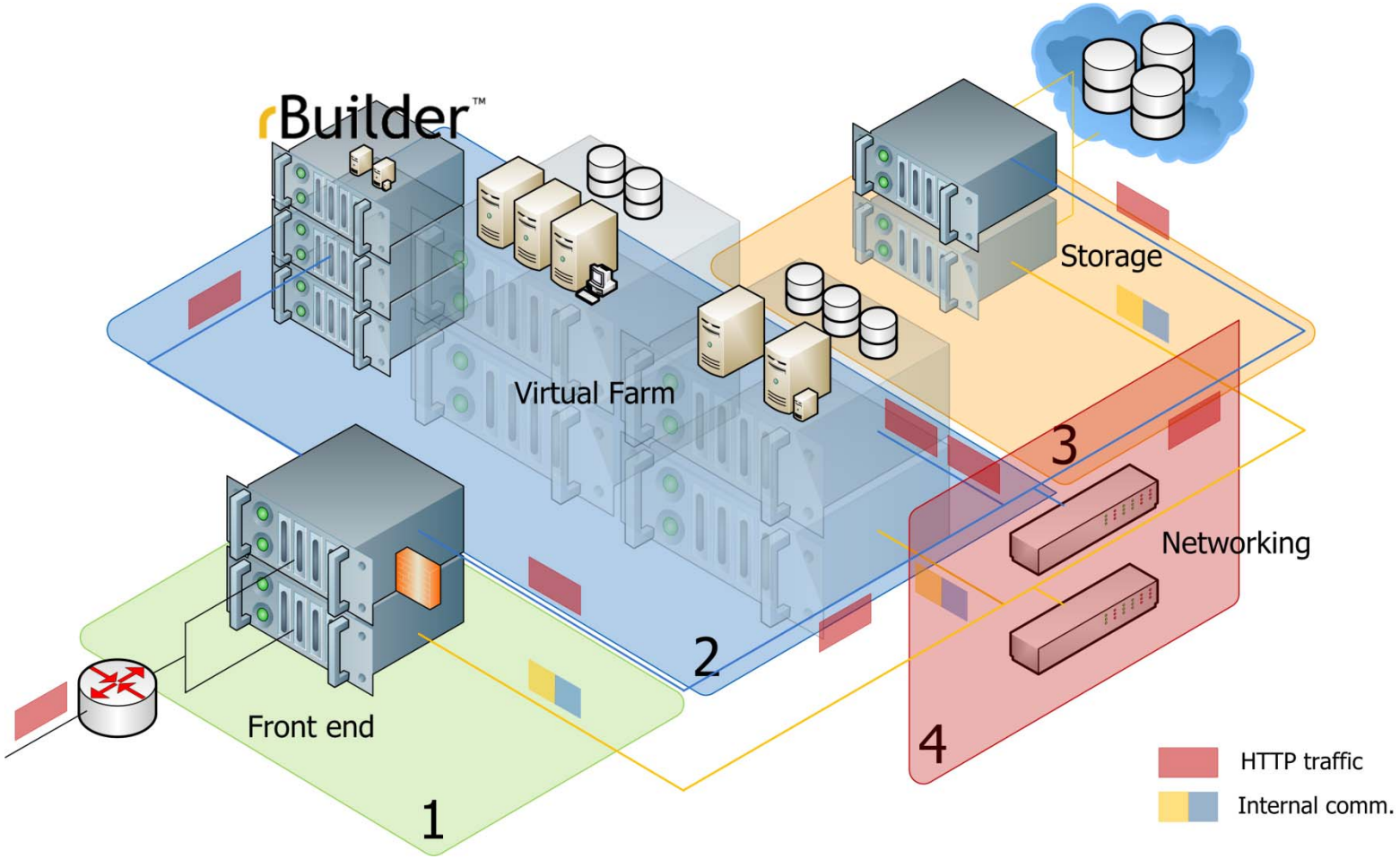
  cernvm-webfs.cern.ch

  rbuilder.cern.ch

  …

CernVM
SoFTware Appliance

Service architecture



CernVM
SoFTware Appliance

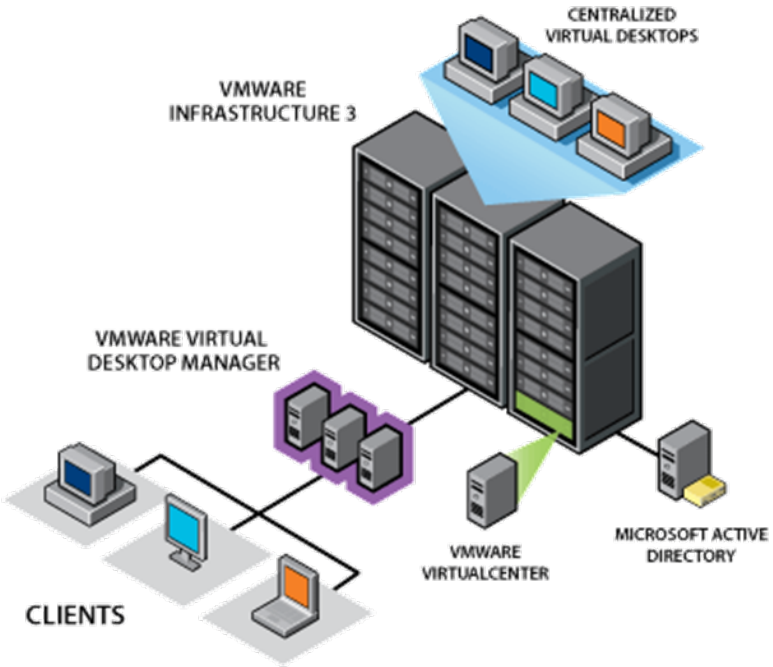
1: Front end

- Gateway to back end services
- DNS Load-balanced cluster hosting public services:
  - Switch L7: combination of Squid/Apache
    - ‘Router’ + webcache
    - SSL engine
    - URL mapper
  - LDAP directory (multimaster replica)

CernVM
SoFTware Appliance

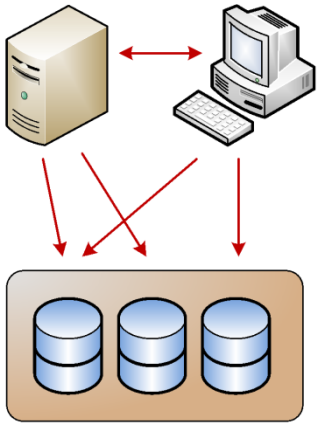
2: CPU provisioning

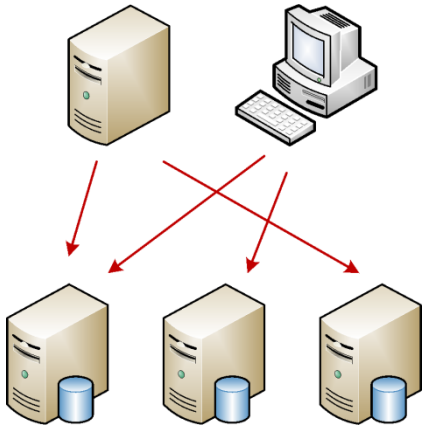
- Virtualization platforms
  - Management tools are key: provision (deployment and life cycle), resource allocation, integration
- VMWare Infrastructure
  - VMotion
  - HA: High Availability
  - DRS: Distributed Resource Scheduling



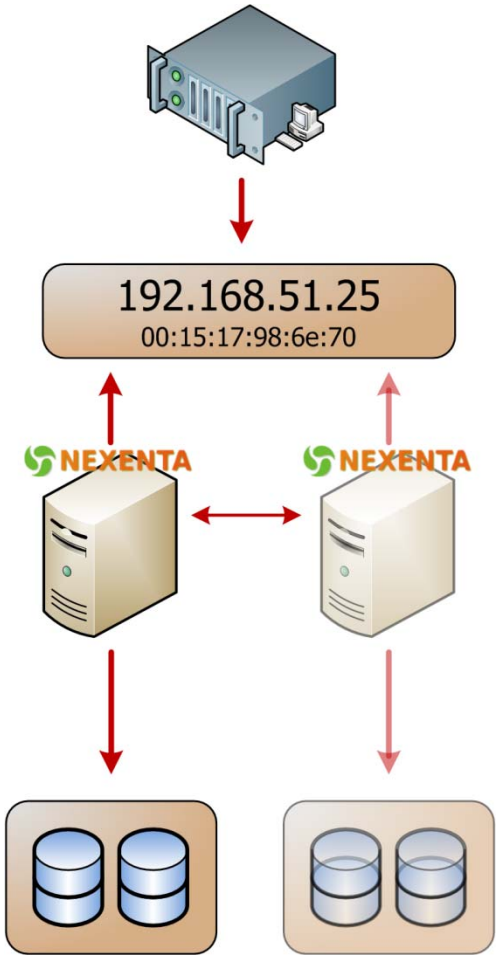
3: Storage provisioning I

- Several ways
  - Shared disk file systems
    - Fibre Channel Protocol, iSCSI …
    - On top: GFS, OCFS,GPFS…
  - Network file systems
    - NFS, AFS, SMB, Lustre …
- Thin provisioning
  - Virtual SAN Appliance
  - Openfiler





3: Storage provisioning II



- NexentaStor
  - 2-node cluster with active replication
  - LDAP: users, groups, netgroups
  - ZFS
    - CVS-like semantics
    - SW Raid
- NFS
  - System disk (vmdk)
  - Data vols. (NFS mounts)

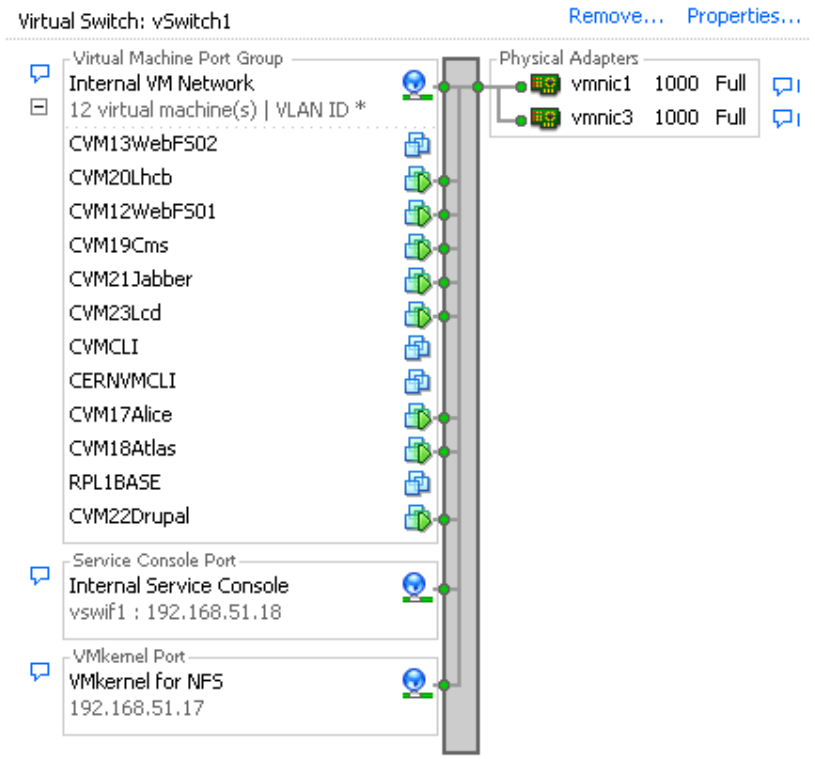
CernVM
SoFTware Appliance

4: Networking

- Two different physical networks
  - Public: dedicated CERN IP service (/24)
    - Only the front end is opened in the central firewall
  - Private: access to storage and VM operations
    - Aggregation of interfaces is used between CPU servers and the storage

CernVM
SoFTware Appliance

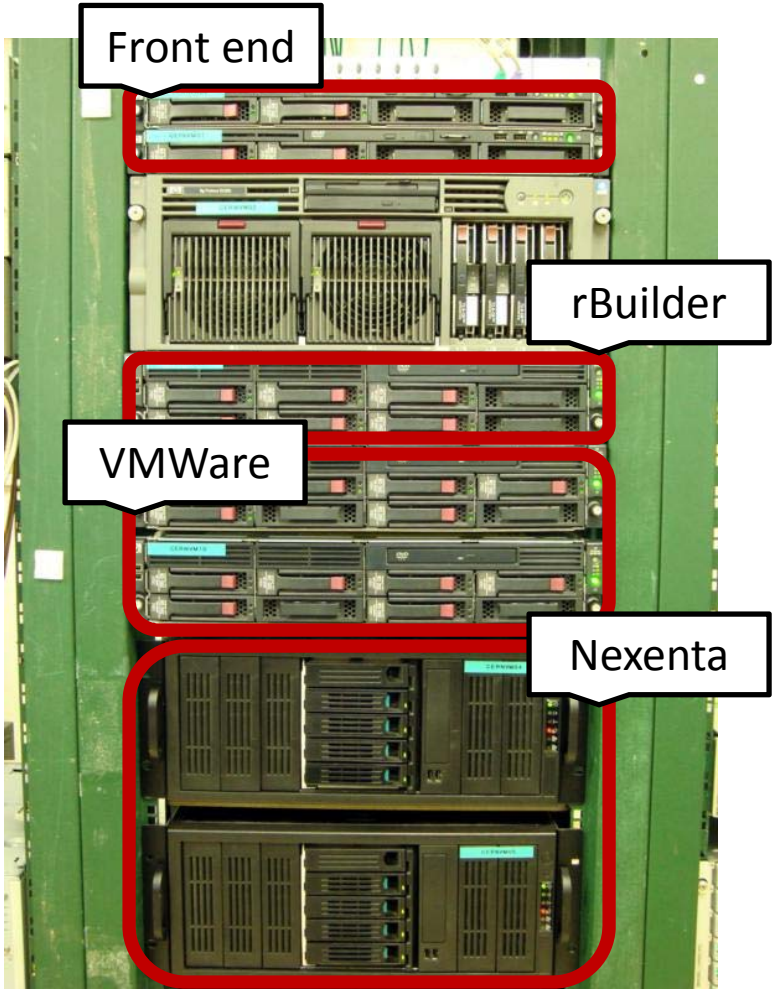
4:Networking

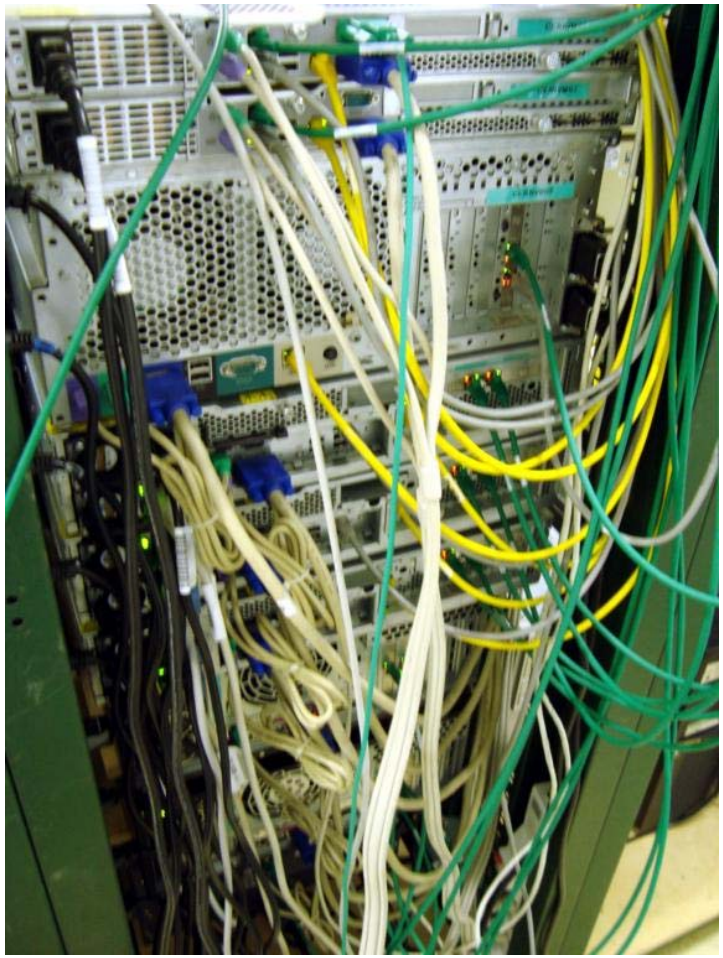


- Network emulated within ESX:
  - VMs can easily move
- Connection to NexentaStor via 2xGb

CernVM
SoFTware Appliance

This is it!





CernVM
SoFTware Appliance

And some statistics

- With those 8 machines we currently run:
  - CVMFS (clients and web servers), rBuilder, LDAP, Zenoss, Boinc, djabberd, SFT-Web
- Traffic (April 2009):
  - +15GB from experiment repositories
  - +600 downloads of CernVM
  - +800 different IPs
  - +50GB total HTTP traffic
  - ~30% hits in the cache

CernVM
SoFTware Appliance

Outline

- Motivation
- Mission
- Requirements
- Service architecture
- Conclusions

CernVM
SoFTware Appliance

Conclusions

- Complete solution for LHC community
  - Run your release in your HW without management issues
- Versatile and modular infrastructure
  - Easy deployment of new services and resources
  - Good performance for commodity HW
  - Easy replicable by definition (DNS-related techniques)

CernVM
SoFTware Appliance

Thanks!

Mailing lists:

- cernvm-talk@cern.ch (open list for announcements and discussion)
- cernvm.support@cern.ch (end-user support for the CernVM project)

Savannah Portal:

- Please submit bugs and feature requests to Savannah at
  - http://savannah.cern.ch/projects/cernvm

CernVM Home Page:

- http://cernvm.cern.ch

rBuilder & Download Page:

- http://rbuilder.cern.ch

CernVM Wiki:

- http://cernvm.cern.ch/project/trac/cernvm

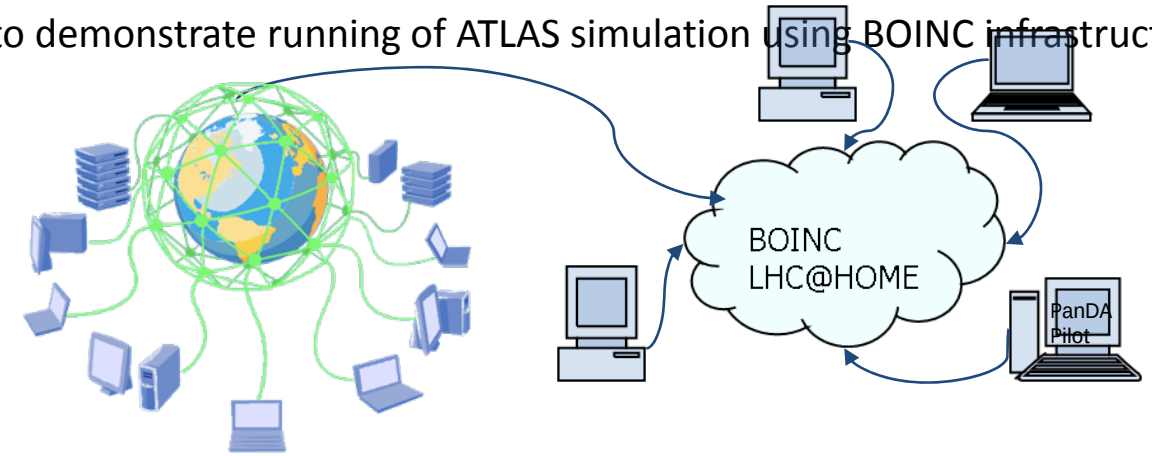
ATLAS Wiki:

- https://twiki.cern.ch/twiki/bin/view/Atlas/CernVM

Bridging Grids & Clouds

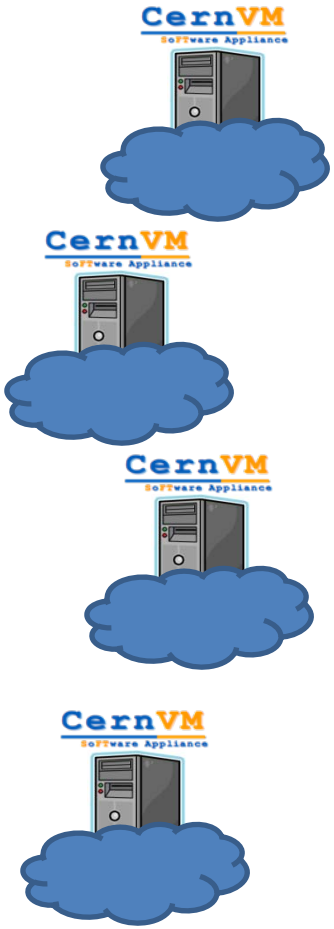
- BOINC
  - Open-source software for volunteer computing and grid computing
  - http://boinc.berkeley.edu/

- CernVM CoPilot development
  - Based on BOINC, LHC@HOME experience and CernVM image
  - Image size is of outmost importance to motivate volunteers
  - Can be easily adapted to Pilot Job frameworks (AliEn,Dirac, Panda)
    - … or Condor Worker, or proofd..
  - Aims to demonstrate running of ATLAS simulation using BOINC infrastructure and PanDa



CernVM CoPilot

AliEn/DIRAC/PanDA



Adapter

0. Send host JDL (free disk space, free memory, available packages)

1. Append framework specific information and request a job

2. Send user job JDL from Task Queue

3. Send input files and commands for execution (packages are already there)

4. When the job is done send back the output files (and the result of validation)

5. Register output files