FermiGrid
Highly Available Grid Services

Eileen Berman, Keith Chadwick
Fermilab

Work supported by the U.S. Department of Energy under contract No. DE-AC02-07CH11359.

Outline

FermiGrid - Architecture & Performance

FermiGrid-HA - Why?

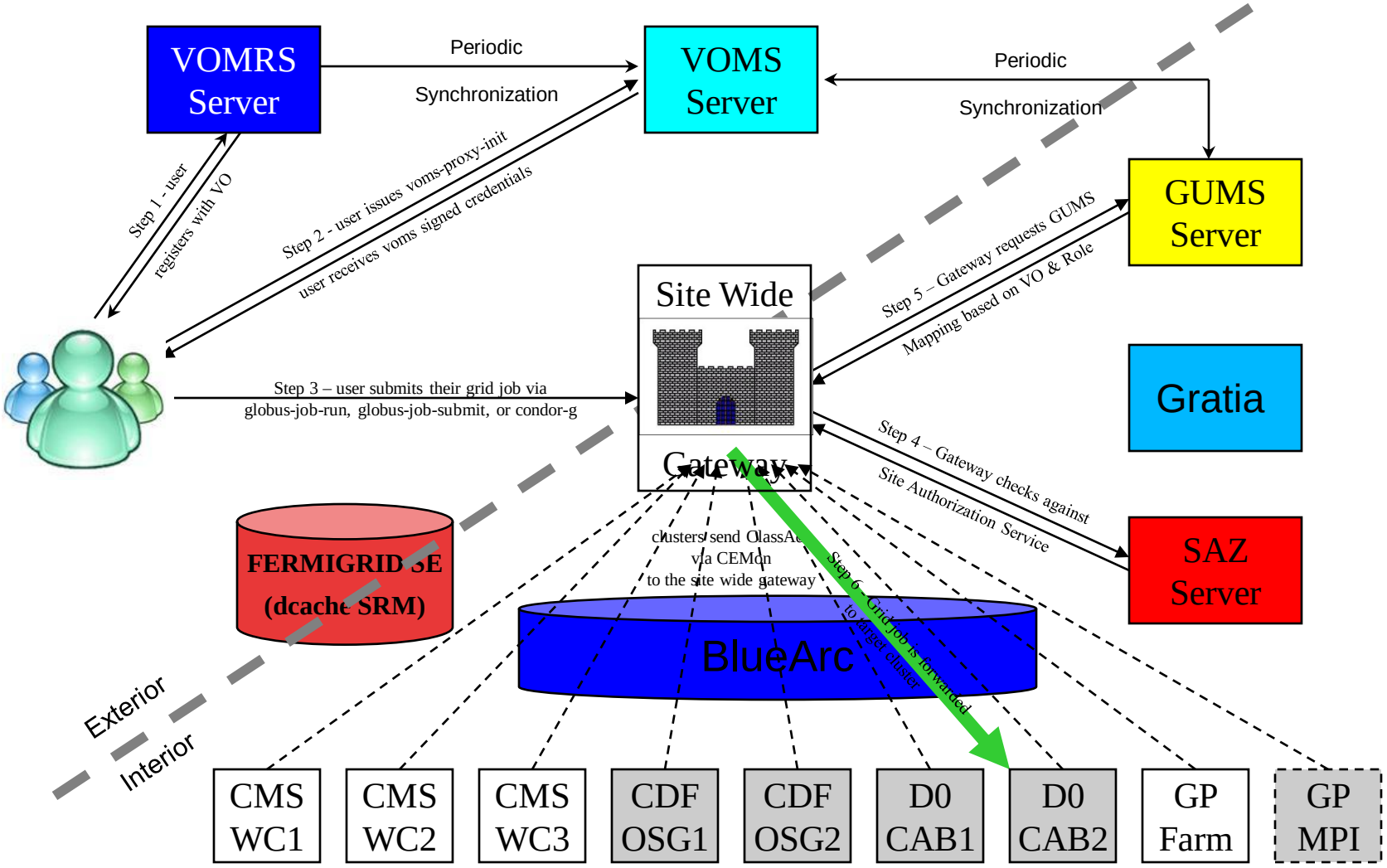
FermiGrid-HA - Requirements & Challenges

FermiGrid-HA - Implementation

Future Work

Conclusions

FermiGrid - Architecture (mid 2007)



FermiGrid-HA - Why?

The FermiGrid "core" services (VOMS, GUMS & SAZ) control access to:

- Over 2,500 systems with more than 12,000 batch slots (and growing!).
- Petabytes of storage (via gPlazma / GUMS).

An outage of VOMS can prevent a user from being able to submit "jobs".

An outage of either GUMS or SAZ can cause 5,000 to 50,000 "jobs" to fail for each hour of downtime.

Manual recovery or intervention for these services can have long recovery times (best case 30 minutes, worst case multiple hours).

Automated service recovery scripts can minimize the downtime (and impact to the Grid operations), but still can have several tens of minutes response time for failures:

- How often the scripts run,
- Scripts can only deal with failures that have known "signatures",
- Startup time for the service,
- A script cannot fix dead hardware.

FermiGrid-HA - Requirements

Requirements:

- Critical services hosted on multiple systems (n ≥ 2).
- Small number of "dropped" transactions when failover required (ideally 0).
- Support the use of service aliases:
  - VOMS: fermigrid2.fnal.gov -> voms.fnal.gov
  - GUMS: fermigrid3.fnal.gov -> gums.fnal.gov
  - SAZ: fermigrid4.fnal.gov -> saz.fnal.gov
- Implement "HA" services with services that did not include "HA" in their design.
  - Without modification of the underlying service.

Desirables:

- Active-Active service configuration.
- Active-Standby if Active-Active is too difficult to implement.
- A design which can be extended to provide redundant services.

FermiGrid-HA - Challenges #1

Active-Standby:

- Easier to implement,
- Can result in “lost” transactions to the backend databases,
- Lost transactions would then result in potential inconsistencies following a failover or unexpected configuration changes due to the “lost” transactions.
  - GUMS Pool Account Mappings.
  - SAZ Whitelist and Blacklist changes.

Active-Active:

- Significantly harder to implement (correctly!).
- Allows a greater “transparency”.
- Reduces the risk of a “lost” transaction, since any transactions which results in a change to the underlying MySQL databases are “immediately” replicated to the other service instance.
- Very low likelihood of inconsistencies.
  - Any service failure is highly correlated in time with the process which performs the change.

FermiGrid-HA - Challenges #2

DNS:

- Initial FermiGrid-HA design called for DNS names each of which would resolve to two (or more) IP numbers.
- If a service instance failed, the surviving service instance could restore operations by “migrating” the IP number for the failed instance to the Ethernet interface of the surviving instance.
- Unfortunately, the tool used to build the DNS configuration for the Fermilab network did not support DNS names resolving to >1 IP numbers.
  - Back to the drawing board.

Linux Virtual Server (LVS):

- Route all IP connections through a system configured as a Linux virtual server.
  - Direct routing
  - Request goes to LVS director, LVS director redirects the packets to the real server, real server replies directly to the client.
- Increases complexity, parts and system count:
  - More chances for things to fail.
- LVS director must be implemented as a HA service.
  - LVS director implemented as an Active-Standby HA service.
- LVS director performs “service pings” every six (6) seconds to verify service availability.
  - Custom script that uses curl for each service.

FermiGrid-HA - Challenges #3

MySQL databases underlie all of the FermiGrid-HA Services (VOMS, GUMS, SAZ):

- Fortunately all of these Grid services employ relatively simple database schema,
- Utilize multi-master MySQL replication,
  - Requires MySQL 5.0 (or greater).
  - Databases perform circular replication.
- Currently have two (2) MySQL databases,
  - MySQL 5.0 circular replication has been shown to scale up to ten (10).
  - Failed databases “cut” the circle and the database circle must be “retied”.
- Transactions to either MySQL database are replicated to the other database within 1.1 milliseconds (measured),
- Tables which include auto incrementing column fields are handled with the following MySQL 5.0 configuration entries:
  - auto_increment_offset (1, 2, 3, … n)
  - auto_increment_increment (10, 10, 10, …)

FermiGrid-HA - Technology

Xen:

- SL 5.0 + Xen 3.1.0 (from xensource community version)
  - 64 bit Xen Domain 0 host, 32 and 64 bit Xen VMs
- Paravirtualisation.

Linux Virtual Server (LVS 1.38):

- Shipped with Piranha V0.8.4 from Redhat.

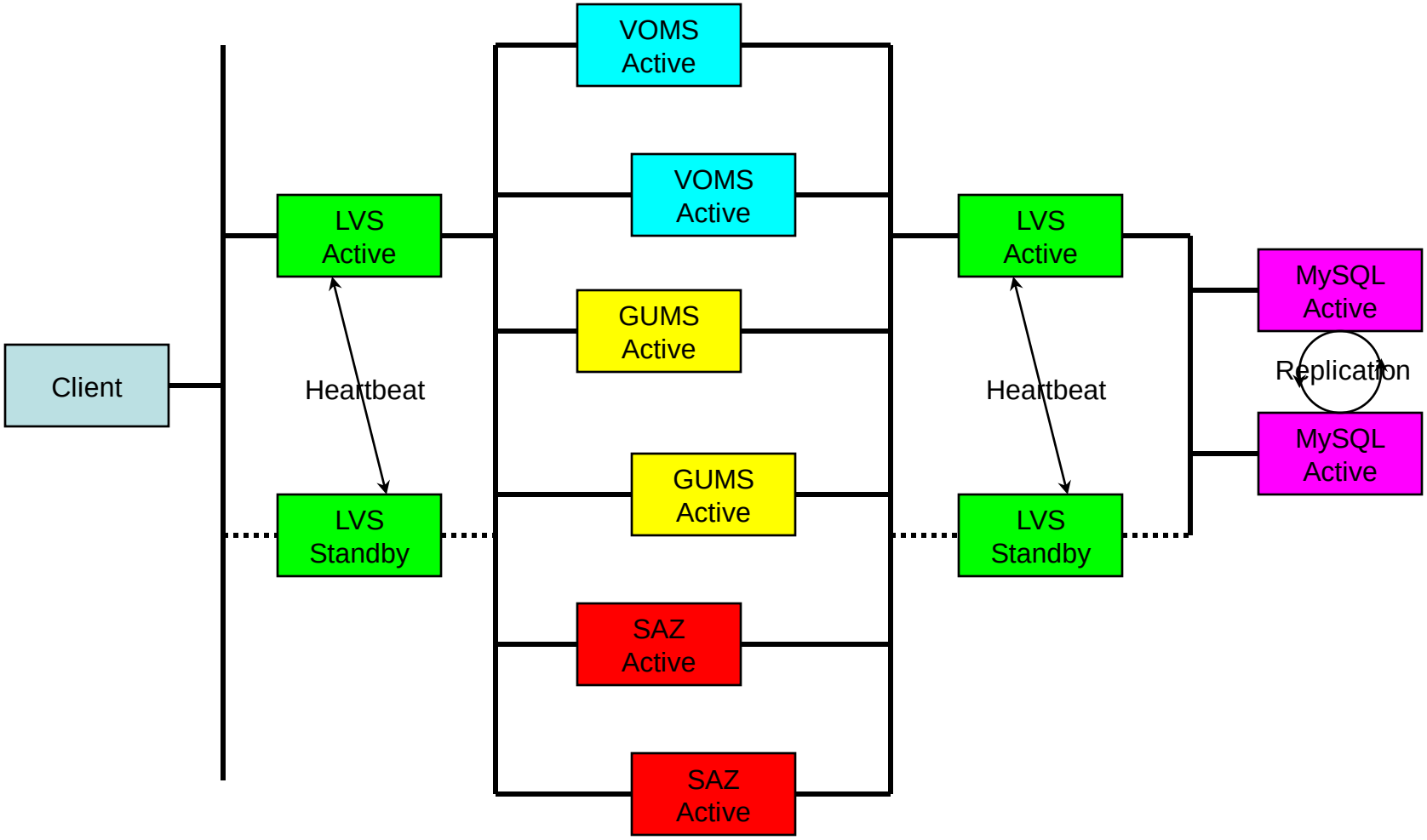
Grid Middleware:

- Virtual Data Toolkit (VDT 1.8.1)
- VOMS V1.7.20, GUMS V1.2.10, SAZ V1.9.2

MySQL:

- MySQL V5 with multi-master database replication.

FermiGrid-HA - Component Design



FermiGrid-HA - Client Communication

1. Client starts by making a standard request for the desired grid service (voms, gums or saz) using the corresponding service “alias”
   voms=voms.fnal.gov, gums=gums.fnal.gov, saz=saz.fnal.gov, fg-mysql.fnal.gov

2. The active LVS director receives the request, and based on the currently available servers and load balancing algorithm, chooses a “real server” to forward the grid service request to, specifying a respond to address of the original client.
   voms=fg5x1.fnal.gov, fg6x1.fnal.gov
   gums=fg5x2.fnal.gov, fg6x2.fnal.gov
   saz=fg5x3.fnal.gov, fg6x3.fnal.gov

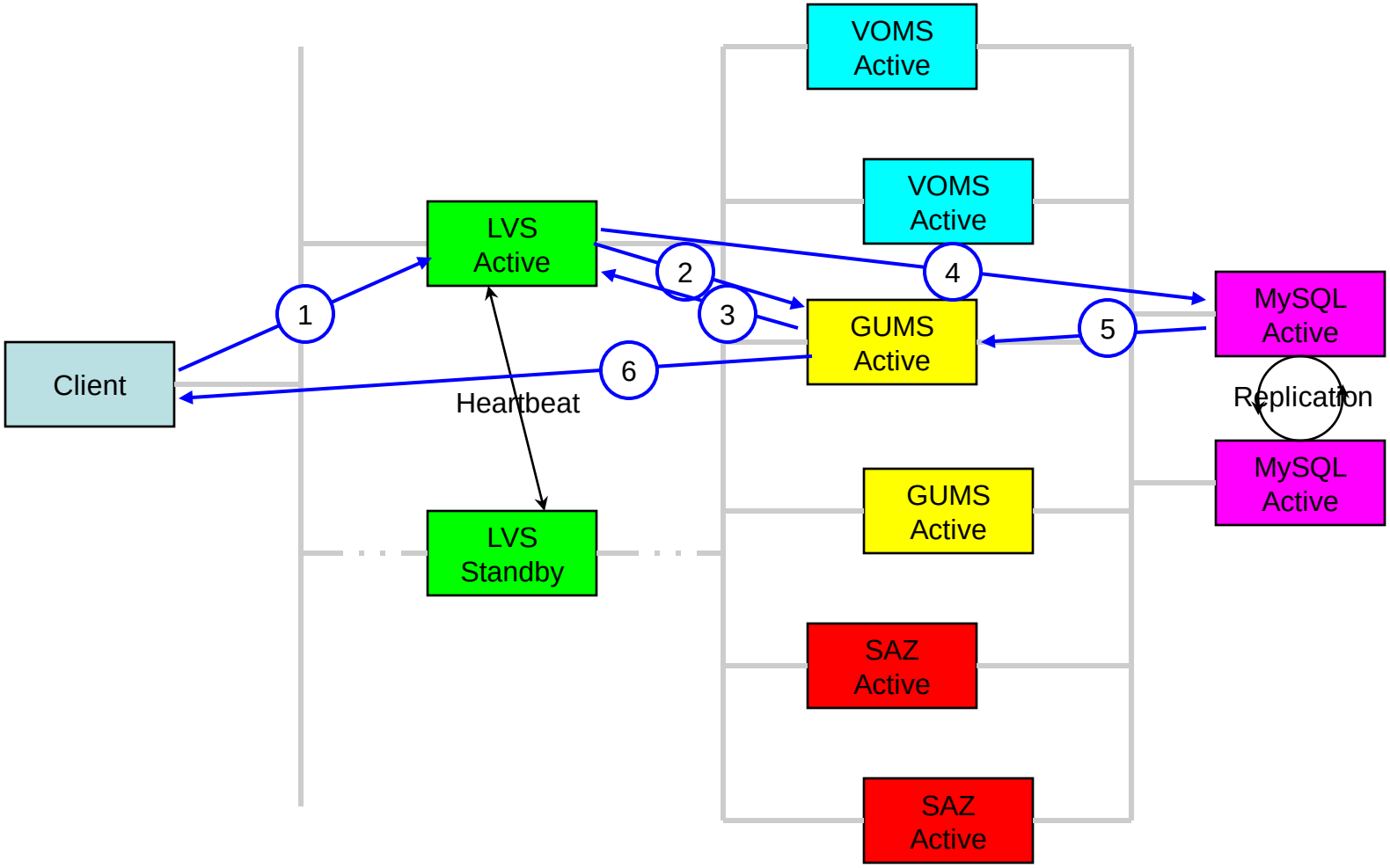
3. The “real server” grid service receives the request, and makes the corresponding query to the mysql database on fg-mysql.fnal.gov (through the LVS director).

4. The active LVS director receives the mysql query request to fg-mysql.fnal.gov, and based on the currently available mysql servers and load balancing algorithm, chooses a “real server” to forward the mysql request to, specifying a respond to address of the service client.
   mysql=fg5x4.fnal.gov, fg6x4.fnal.gov

5. At this point the selected mysql server performs the requested database query and returns the results to the grid service.

6. The selected grid service then returns the appropriate results to the original client.

FermiGrid-HA - Cleint Communication Animation



FermiGrid-HA - Host Configuration

The fermigrid5&6 Xen hosts are Dell 2950 systems.

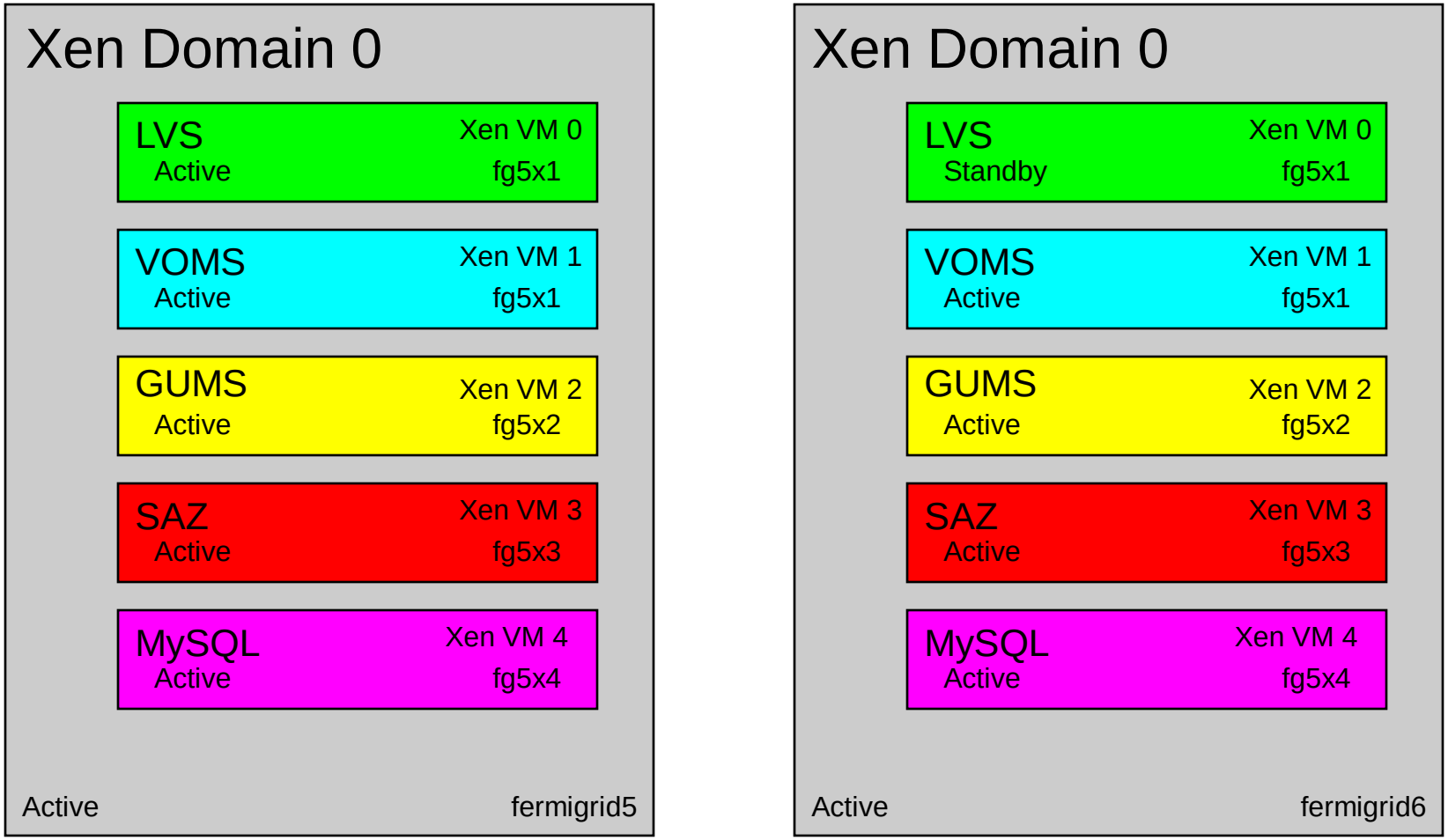
Each of the Dell 2950s are configured with:

- Two 3.0 GHz core 2 duo processors (total 4 cores).
- 16 Gbytes of RAM.
- Raid-1 system disks (2 x 147 Gbytes, 10K RPM, SAS).
- Raid-1 non-system disks (2 x 147 Gbytes, 10K RPM, SAS).
- Dual 1 Gig-E interfaces:
  - 1 connected to public network,
  - 1 connected to private network.

System Software Configuration:

- Each Domain 0 system is configured with 5 Xen VMs.
  - Previously we had 4 Xen VMs.
- Each Xen VM, dedicated to running a specific service:
  - LVS Director, VOMS, GUMS, SAZ, MySQL
  - Previously we were running the LVS director in the Domain-0.

FermiGrid-HA - Actual Component Deployment



FermiGrid-HA - Performance

Stress tests of the FermiGrid-HA GUMS deployment:

- A stress test demonstrated that this configuration can support ~9.7M mappings/day.
  - The load on the GUMS VMs during this stress test was ~9.5 and the CPU idle time was 15%.
  - The load on the backend MySQL database VM during this stress test was under 1 and the CPU idle time was 92%.

Stress tests of the FermiGrid-HA SAZ deployment:

- The SAZ stress test demonstrated that this configuration can support ~1.1M authorizations/day.
  - The load on the SAZ VMs during this stress test was ~12 and the CPU idle time was 0%.
  - The load on the backend MySQL database VM during this stress test was under 1 and the CPU idle time was 98%.

Stress tests of the combined FermiGrid-HA GUMS and SAZ deployment:

- Using a GUMS:SAZ call ratio of ~7:1
- The combined GUMS-SAZ stress test which was performed demonstrated that this configuration can support ~6.5 GUMS mappings/day and ~900K authorizations/day.
  - The load on the SAZ VMs during this stress test was ~12 and the CPU idle time was 0%.

FermiGrid-HA - Production Deployment

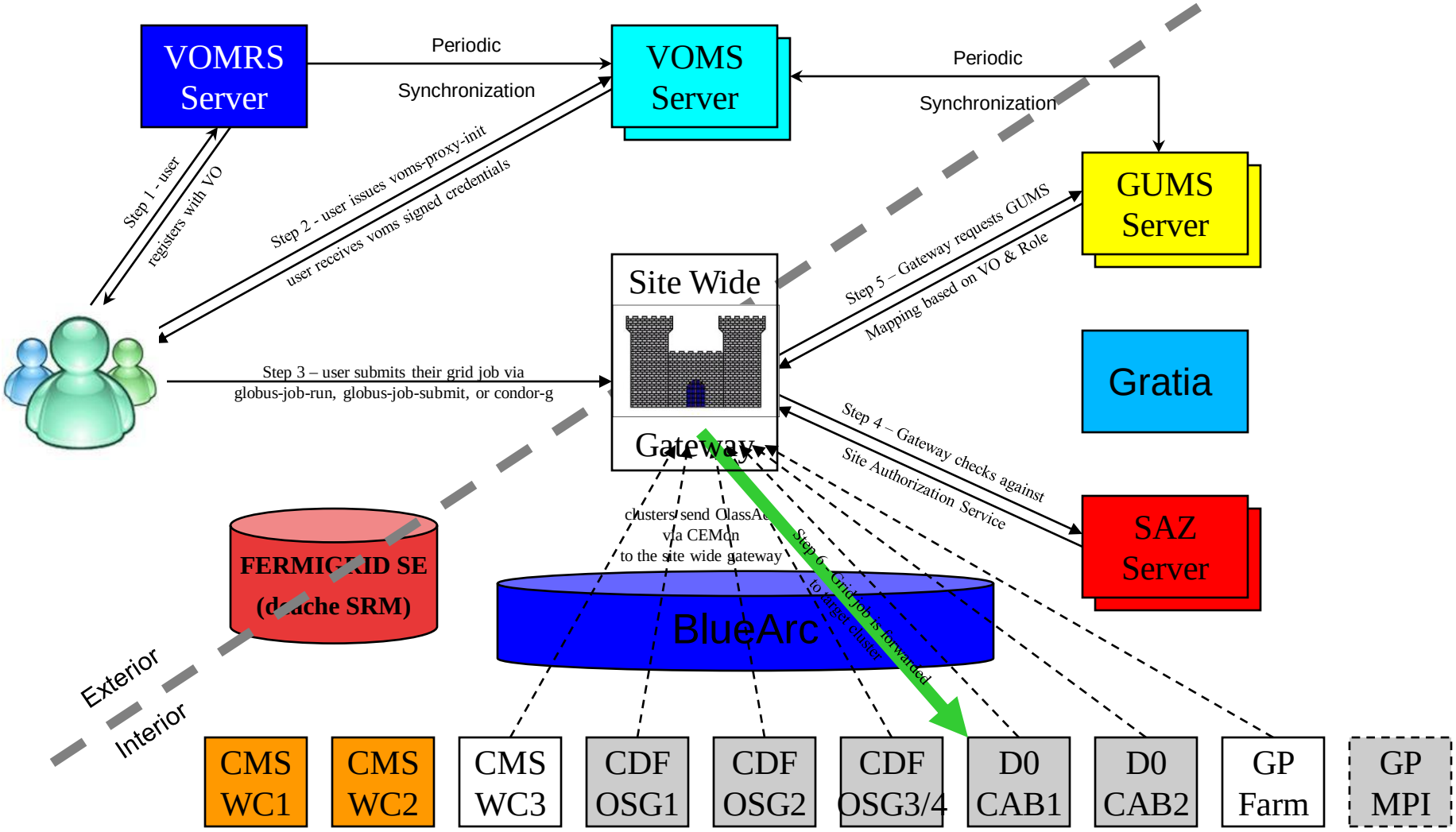
FermiGrid-HA was deployed in production on 03-Dec-2007.

- In order to allow an adiabatic transition for the OSG and our user community, we ran the regular FermiGrid services and FermiGrid-HA services simultaneously for a three month period (which ended on 29-Feb-2008).

We have already utilized the HA service redundancy on several occasions:

- 1 operating system “wedge” of the Domain-0 hypervisor together with a “wedged” Domain-U VM that required a reboot of the hardware to resolve.
- multiple software updates.
- ***Without any user impact!!!***

FermiGrid - Current Architecture



FermiGrid-HA - Future Work

Over the next three to four months we will be deploying "HA" instances of our other services:

- Squid, MyProxy (with DRDB), Syslog-Ng, Ganglia, and others.

Redundant side wide gatekeeper:

- We have a preliminary "Gatekeeper-HA" design
- Based on the "manual" procedure to keep jobs alive during OSG 0.6.0 to 0.8.0 upgrade.
- We expect that this should keep Globus and Condor jobs running.

We also plan to install a test gatekeeper that will be configured to receive Xen VMs as Grid jobs and execute them:

- This a test of a possible future dynamic "VOBox" or "Edge Service" capability within FermiGrid.

FermiGrid-HA - Conclusions

Virtualisation benefits:

+ Significant performance increase,
+ Significant reliability increase,
+ Automatic service failover,
+ Cost savings,
+ Can be scaled as the load and the reliability needs increase,
+ Can perform “live” software upgrades and patches without client impact.

Virtualisation drawbacks:

- More complex design,
- More “moving parts”,
- More opportunities for things to fail,
- More items that need to be monitored.

Fin

Any Questions?