AFS & Kerberos Best Practices Workshop
Carnegie-Mellon University, Pittsburgh June 20-24 2005

A virtualized OpenAFS cell for debugging and experimentation

G. Bracco, P. D'Angelo, S.Migliori

ENEA INFO, C.R. ENEA Frascati
V. E. Fermi 45 Frascati ROMA (Italy)
bracco@frascati.enea.it

Outline

- why a virtualized AFS cell?
- the implementation
- enea.it cell and ENEA-GRID
- an example: solving a volume numeration problem

Why a virtualized AFS cell?

An OpenAFS installation in a set of virtual machines provides a simple and flexible way:

- to approach OpenAFS: AFS/OpenAFS can not be tried without having a full installation of all its components, dbservers, fileservers and clients and this may result in a steep learning curve.
- to experiment with a test OpenAFS cell in order to solve administration issues of a real AFS cell.
- to illustrate OpenAFS behaviour and features in training sessions and tutorials.

Performance is not an issue in the framework of this presentation but virtualized OpenAFS servers are also used in production [sites from openafs-info mailing list: umr.edu, cs.auckland.ac.nz...]

The implementation

Even if performance is not a main issue, at least 3 virtual machines must concurrently run on the host:

- host: a laptop ASUS M2400N, Centrino 1.6 Ghz, 512 MB RAM, HD 60 GB, SUSE 9.1, kernel 2.6.5-7.75

The components:

- processor emulator: QEMU a GPL project by fabrice bellard http://fabrice.bellard.free.fr/qemu
- the emulated OS: Linux RedHat 8.0, minimal installation [~450 MB], no X11
- the network: 802.1d Ethernet Bridging kernel support
- OpenAFS 1.2.11

QEMU (1)

"QEMU is a generic and open source processor emulator which achieves a good emulation speed by using dynamic translation"
Homepage: fabrice.bellard.free.fr/qemu

USENIX 2005 F. Bellard, "QEMU, a Fast and Portable Dynamic Translator"

It emulates x86, PowerPC, ARM, SPARC on several hosts x86, PowerPC, ARM, SPARC, Alpha, MIPS.

GPL licence but an optional linux kernel module [kqemu] is close & proprietary, even if free [...and looking for sponsorship!]

x86 emulated systems: Linux, BeOS, NetBSD, OS2, Solaris, Win.

Emulation PowerPC [PowerMac, PReP], SPARC in progress.

ref: M.Santosa "Build your own virtual cluster", BYTE.com, 2004/07

QEMU (2)

Installation: download qemu-0.7.0.tar.gz

```
cd / ; tar -xzvf qemu-0.7.0.tar.gz
```

Performance are enhanced by the kqemu module:

```
modprobe kqemu
```

Creation of a disk image file as the disk of the emulated system:

```
dd of=disk.img bs=1M seek=800 count=0
```

System installation: boot from the host CD-ROM

```
qemu -hda disk.img -cdrom /dev/cdrecorder -boot d -m 64
```

qemu run: boot from disk image. Network and Time options

```
qemu -hda disk.img -m 64 -localtime -n ./qemu-ifup_0.0.0.0
        -macaddr 52:54:00:12:34:56
```

QEMU networking (1)

More than 1 instance of qemu can be run together.

Each instance of qemu makes use of tun/tap virtual network device which is supported by linux kernel. The parameter:

```
-n ./qemu-ifup_0.0.0.0
```

results in qemu calling a script qemu-ifup_0.0.0.0 executing

```
sudo /sbin/ifconfig $1 0.0.0.0
```

which configures the virtual ethernet device $1=tun0, tun1, tun2...

The MAC address of each instance interface is given by the parameter:

```
-macaddr 52:54:00:12:34:nn
```

where *nn* must be different for each qemu instance.

QEMU networking (2)

Network configuration is completed by using bridging which requires rpm bridge-utils rpm and 802.1d Ethernet Bridging kernel support.

For example: having 3 qemu instances with static IP address 10.128.1.11, 10.128.1.12,10.128.1.13

The commands:

```
brctl addbr br0
ifconfig br0 10.128.1.1 netmask 255.255.255.0
brctl addif br0 tun0
brctl addif br0 tun1
brctl addif br0 tun2
```

result in a working network where the host address is 10.128.1.1

QEMU and Time

AFS requires a reasonable time synchronization between servers.

This proved to be somewhat critical with qemu as the local clock of each qemu instance is affected in an almost erratic way by the host system load. My recepi:

- qemu option: -t localtime
- for host kernel 2.6.x: kernel option clock=pit
- forced synchronization:

  each minute a crontab job on the host triggers the synchronization of each qemu instance with the host itself, using rsh and rdate:

  /usr/bin/rsh 10.128.1.11 rdate -s 10.128.1.1

  /usr/bin/rsh 10.128.1.12 rdate -s 10.128.1.1

  /usr/bin/rsh 10.128.1.13 rdate -s 10.128.1.1

DNS

To have the proper direct/reverse IP name resolution for each qemu instance bind has been installed on each qemu instance.

The configuration has been obtained by modifying a caching only nameserver setup: /etc/named.conf

options { directory "/var/named"; cache-file "cachefile.dns"; forwarders {*site_DNS_addresses*} ; forward only;};

and loading the local bind cache: /var/named/cachefile.dns [the file must have user & group ownership “named” and chmod 600]

```
; local
11.1.128.10.in-addr.arpa. 43098 PTR     rh80-1.frascati.enea.it.
...
; local
 rh80-1.frascati.enea.it. 8283   A       10.128.1.11
```

This example corresponds to RH8 bind configuration

OpenAFS on QEMU

OpenAFS on 3 qemu instances: rh80-1, rh80-2, rh80-3

Installing rpms from www.openafs.org, Red Hat 8.0:

openafs-1.2.11-rh8.0.1.i386.rpm
openafs-client-1.2.11-rh8.0.1.i386.rpm
openafs-kernel-1.2.11-rh8.0.1.i386.rpm
openafs-kpasswd-1.2.11-rh8.0.1.i386.rpm

openafs-server-1.2.11-rh8.0.1.i386.rpm

Transfer from host to qemu instances via sftp or rcp

Installation set-up scripts prepared on the basis of:
www.openafs.org/pages/doc/QuickStartUnix/

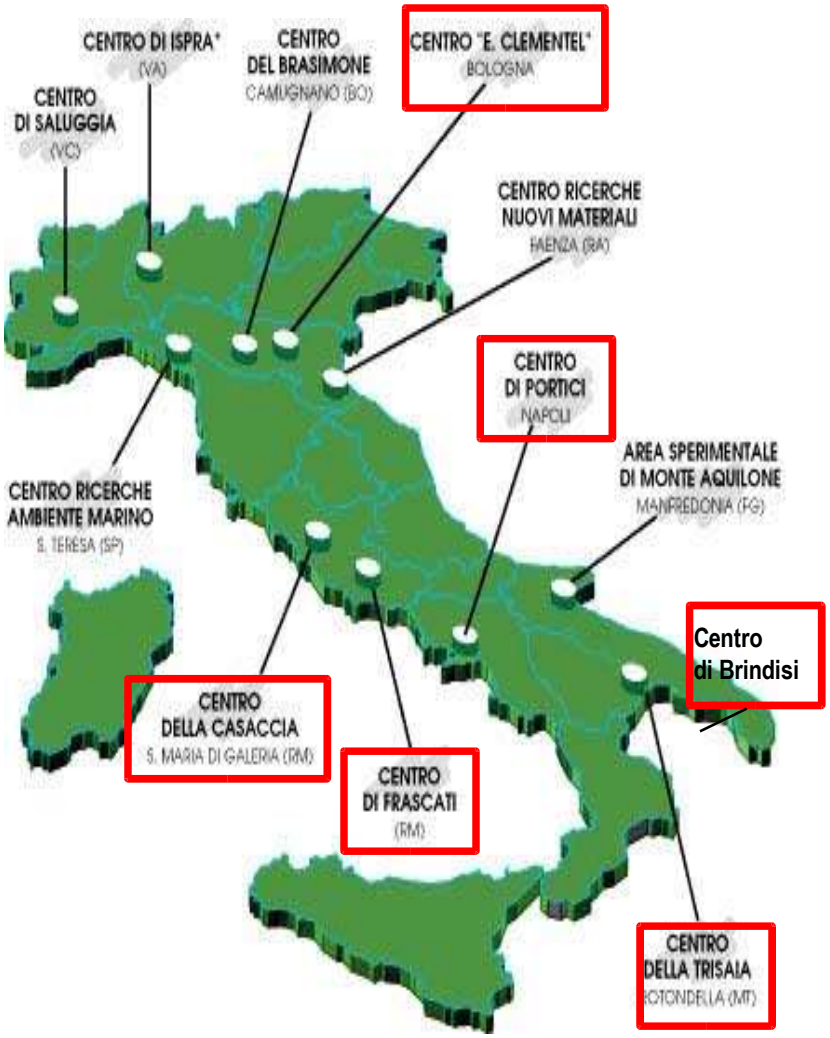
The cellname: qemua root.cell /afs/qemua/

Each instance is a dbserver/fileserver/client; the host can be a client.

A cell administration issue

- ENEA GRID and AFS cell enea.it
- finding a solution for a volume ID problem in cell enea.it

ENEA



ENEA: Italian National Agency for New Technologies, Energy and Environment

12 Research sites and a Central Computer and Network Service (ENEA-INFO) with 6 computer centres managing multi-platform resources for serial & parallel computation and graphical post processing.

Other computer resources in ENEA: departments & individuals.

WAN connectivity: GARR Italian Research and Academy Network: 4-30 Mbits/s

ENEA GRID

INFO-ENEA computational resources:

- hardware: ~300 cpu: IBM SP; SGI Altix & Onyx; Linux clusters 32/64; Apple cluster; Windows servers. Most relevant resource: IBM SP4 128 nodes ~1Tflops
- software: commercial codes (fluent, ansys, abaqus..); elaboration environments (Matlab, IDL, SAS..)

ENEA GRID mission [started 1999]:

- provide a unified user environment and an homogeneous access method for all ENEA researchers, irrespective of their location
- implement tools to facilitate the integration of department and individual resources

ENEA GRID architecture

GRID functionalities

"unique authentication, authorization, resource access and resource discovery" (Foster & Kesselman "Anatomy of the GRID" 2001)

are provided using "mature", multi-platform components:

Distributed File System: AFS/OpenAFS

Resource Manager: LSF Multicluster [www.platform.com]

Unified user interface: Java & Citrix Technologies

AFS/OpenAFS

- manages software and data distribution
- manages user authentication/authorization [still kerberos 4]
- facilitates the integration of department and individual resources

enea.it cell

6 db-servers in 4 sites

- AFS Transarc 3.6.x / AIX 4.3.3
- Frascati (3) Bologna, Casaccia, Trisaia(1)

12 File-servers in 6 sites

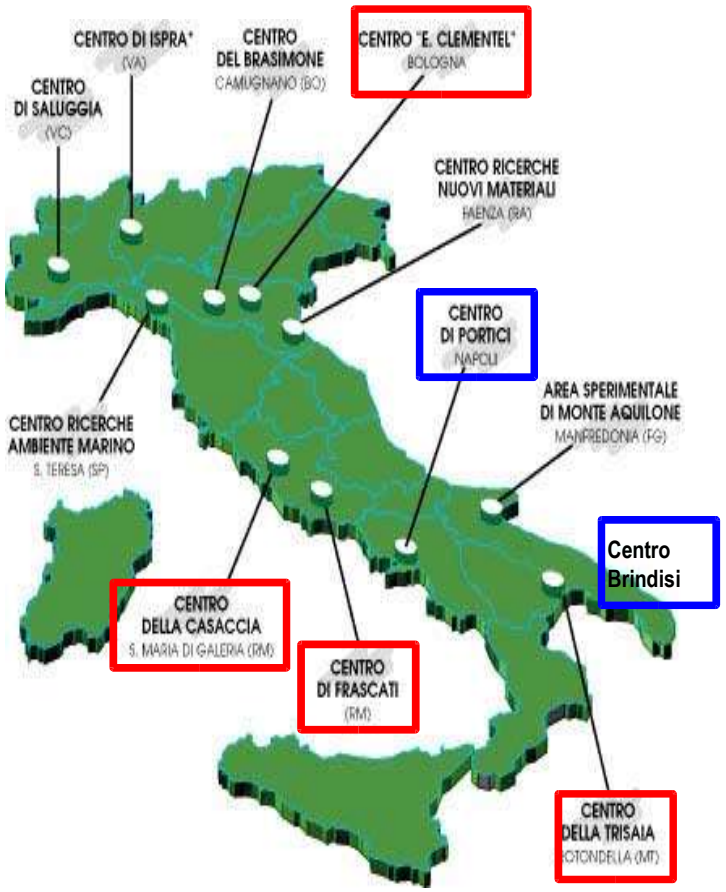
- AFS Transarc 3.6.x / AIX 4.3.3, Solaris 9
- OpenAFS 1.2.11/1.2.13 / Scientific Linux 3.0.4, CASPUR BigBox 3.0

Clients: 50+ comput.systems; 100+ users

- AFS Transarc 3.6.x AIX 5.1/5.2, IRIX, Solaris
- OpenAFS: Linux 32/64[Altix], MacOSX, Win

650 registered users

2 TB data



an example

Finding a solution for a volume ID problem in cell enea.it

volume ID problem (1)

AFS volume ID starts at HEX 2000000 DEC 536870912 which is usually the ID of volume root.afs.

In March 2002 a volume was created with ID=2258389551 while the last regular ID at the time was: 536875338. A BIG JUMP!

The problem was not immediately spotted because the volume almost worked. Problem: it could not be moved to another server.

Transarc support was contacted and patched binaries were provided for: fileserver, volserver, vlserver, salvager, volinfo.

The reason: 2258389551 is larger than the largest signed integer 2^31-1=2147483647

Why this happened? WE DON'T KNOW!

But also stanford.edu cell has strangely large Vol IDs: from openafs-info Jan 2005 mailing list: vol dist.web.ssl ID 2003993553 <2^31-1

volume ID problem (2)

OpenAFS does not support this “large” ID value and so the migration of our cell to OpenAFS was impossible.

In 2004 the decision was taken to correct the problem.

The correction procedure:

(1) Correct the value in DB which is used to provide the ID of the next volume to be created: MaxVolumeId. From the moment when this correction is performed all new volumes are created with standard ID.

(2) Correct the wrong ID of all the “large ID” volumes. About 500 out of ~1000 volumes were affected.

Step (1) is the most critical, as no standard AFS tools are available;

Step (2) is time consuming and must be run after step (2).

Both steps have tested on the virtualized test cell.

volume ID problem (3)

MaxVolumeId is a field from the vldb.DB0 which is located on the dbservers in /usr/afs/db directory. Its value can be printed using

```
vldb_check -database vldb.DB0 -vheader
```

Using hexdump the location of MaxVolumeId was identified and a correction script was prepared, using hexdump, xxd, sed:

```
new=20010000
old=`hexdump -C vldb.DB0|grep "^00000050"|cut -b36-47| tr -d " "`
command="/^0000050:/s/$old/$new/"
xxd -g0 vldb.DB0 | sed $command | xxd -revert > vldb.DB0.mod
```

The script was run on the dbserver which was the sync site:

```
bos shutdown dbserver_name vlserver -wait
    run the correction script and rename vldb.DB0.mod in vldb.DB0
bos restart dbserver_name vlserver
```

The procedure takes < 15 s and the dbserver remained sync site.

volume ID problem (4)

When a new volume was created MaxVolumeId of the sync site was used as ID and then propagated to the other dbservers.

The correction of the “large ID” volumes started by producing the list of all the affected volumes [vos listvldb], then for each volume:

(1) vos dump, (2) vos delete, (3) vos create, (4) vos restore

The procedure was run in a semi-automatic way for volumes without replicas [less than 20 affected volumes had replicas and where corrected by hand] mostly on the fileserver where volumes were located or at least in the same site.

After each correction the comand fs checkv was automatically run on all computing node of ENEA-GRID, by using a LSF batch submit;

The procedure took about 3 weeks, mostly at nigth / weekend and sometimes explicit agreements with selected user had to be found: volumes with size up to 55 GB (3 hours)

What next and acknowledgements

What next

- checking network behaviour [a virtual AFS cell with 6 DB servers in two subnets, taking advantage of kernel ip forwarding, has already been implemented]
- improving the choice of the emulated Linux distribution
- testing and experimenting kerberos 5 transition

ENEA GRID and enea.it AFS cell are operated with the support of many people in various ENEA sites:

S. Taglienti, R. Guadagni, F. Simoni, A. Perozziello, A. De Gaetano, S. Pecoraro, D. Giammattei, G. Mencuccini., M. De Rosa, M. Caiazzo, A. Palumbo, G. Elmo, S. Pierattini, M. Impara, G. Furini, C. Zini...