Managing Xen with SmartFrog

CERN openlab II quarterly review
31 January 2007
Preview of VHPC’07

Xavier Grehant
Focus

- On-demand execution environments must be:
  - Virtual
  - Distributed
  - Configurable
  - Composable

- For batch jobs (grid)
  - In contrast to Virtual Workspaces, Tycoon

- And software quality assurance (QA) tasks
  - In contrast to NMI builds & tests infrastructure
Why virtual resource management?

- Exploits benefits of virtual machines
  - Compatibility
  - Data isolation
  - Resource sharing and isolation

- Virtualization results in much more flexibility

- Xen enables automation
  - It does not provide the management system
SmartDomains: Xen and SmartFog

- **Xen**
  - High performance
  - Advanced features
  - Popularity

- **SmartFrog**
  - Description language
    - Configure and choreograph components
    - Tree of components with attributes
  - Daemons
    - Peer-to-peer network for deployment
  - Deployment engine
    - Interprets description
    - Dispatches work to daemons
    - Liveness, dependencies, references
SmartDomains: Xen and SmartFog

- Xen VM deployment with SmartFrog

- Users submit a description to launch the pool of VMs

- SmartDomains automates deployment and management
Usage: getting resource

> sfStart localhost pool virtualPoolDesc.sf
> sfTerminate localhost pool

- Simplicity on/off required for batch computing

- Other virtualization management systems:
  - Enterprise systems:
    - Platform VMO, Cassat Collage, OpenQRM, DynamicOE: Let admin define high-availability policies among apps
  - Open source systems:
    - Enomalism, Virtual Workspace + GPE: interface to Xen VMM
Usage: describing resource

volumesize "1g";
swapSize "512m";
usingExistingVolumes false;
keepVolumes "statuquo";
saveImage true;
saveImageName "";
saveImagePath "/tmp";
saveImageExtension "tar";
makeFs "mkfs.ext3";
volumeBaseName "volume1";
tempMountPoint "/tmp/mnt";
kernel "/boot/vmlinuz-2.6-xen";
ramdisk "/boot/initrd-2.6-xen.img";
netmask "255.255.0.0";
memory 512;
vcpus 1;
domainLivenessDelay 2000;
domainLivenessFactor 3;
domainLivenessCheck true;
extra "fastboot noUSB";
domainName "virtual-domain";
baseImage "slc3-WN.tgz";
ip "123.456.78.90";
gateway "98.765.43.21";
hostname "host";
Usage: describing resource

```c
#include "org/smartfrog/components.sf"
#include "ch/cern/openlab/smardomains/components.sf"
#include "org/smartfrog/services/shellscript/components.sf"

DefaultXenDomain extends XenDomain {
  - shell LAZY ATTRIB myShell;
  - kernel "/boot/vmlinux-2.6-xen";
  - gateway "";
  - netmask "255.255.0.0";
}

PhysicalHost extends Compound {
  - sfSyncTerminate true;
  - myShell extends BashShell;
}

sfConfig extends Compound {
  sfSyncTerminate true;

  computer1 extends PhysicalHost {
    sfProcessHost "localhost.cern.ch";

    loop extends LoopbackStorageBackend {
      - shell LAZY ATTRIB myShell;
      - domainName "domainLoopback";
      - baseImage "/data/xen/sl3-smartfrog.img";
    }

    domain1 extends DefaultXenDomain {
      - domainName "domainLoopback";
      - ip "";
      - hostname "localhost";
      - storageBackend LAZY ATTRIB loop;
    }

    lvm extends LVMStorageBackend {
      - shell LAZY ATTRIB myShell;
      - domainName "domainLVM";
      - baseImage "/data/xen/sl3-smartfrog.img";
    }

    domain2 extends DefaultXenDomain {
      - domainName "domainLVM";
      - ip "";
      - hostname "localhost-dom2";
      - storageBackend LAZY ATTRIB lvm;
    }

  }

  computer2 extends PhysicalHost {
    sfProcessHost "localhost.cern.ch";
    ...
  }
```
Usage: example
Full configurability with base components attributes

- Compared to:
  - Amazon EC2: same server, custom filesystem
  - Tycoon: same filesystem, custom resources

Lifecycle management with components composition

- Never seen before (acknowledged as issue in Xen roadmap)
Administering resources

- Specially suited for trusted community (P2P)
  - A computer bootstraps whole resource
  - Security system follows same scheme
- Predefine specialized components in description language
  - Extension mechanism, links
  - For specific usage, or simplicity of end-users descriptions
- Or provide a web interface
  - Hide descriptions, fill up missing fields
- Example: gLite testing
Further enrich functionality and structure

- Composite pattern:
  - Plug-in functionality
    - Scheduling, balancing, high-availability
  - Create higher-level structures
    - Virtual clusters
  - Modularity and reuse

- Peer-to-peer
  - Scope of an algorithm: the P2P network
    - As opposed to Tycoon where bidding scope is inside a physical host
  - No single point of failure

```java
simpleScheduler extends Scheduler {
    hosts ["host1", "host2", "host3"];
}
VMs2Dispatch extends Schedulee {
    scheduler LAZY ATTRIB simpleScheduler;
    - extends VM {...}
}
Conclusion

- In the future, resource = VM

- SmartDomains uniqueness
  - Batch jobs tests: on / off
  - Distributed: workflows and lifecycle management
  - Peer-to-peer
  - Composition

- Applications:
  - Batch computing
  - QA tasks
  - Direct / specialized / enriched usage