Efficient and Scalable Operating System Provisioning with Kadeploy3

Luc Sarzyniec
<luc.sarzyniec@inria.fr>
Plan

1. Introduction
   - Use cases
   - Challenges
   - Key features

2. Kadeploy internals

3. Example usages at large scale

4. Conclusion
Use cases

• System administration for HPC clusters
  ▶ Install and configure large number of nodes
  ▶ Manage a library of pre-configured system images
  ▶ Reliability of the installation process
  ▶ Hardware compatibility

• Scientific and experimental context (Grid’5000)
  ▶ Launch experiments in a clean environment
  ▶ Custom environments (specific libraries, OS)
  ▶ Execute root commands

• History
  ▶ 2001-2008: CLIC, Grenoble (kadeploy 1,2)
  ▶ 2008-2011: Aladdin-G5K (kadeploy 3)
  ▶ 2011-2013: Inria ADT Kadeploy
Challenges

• Large scale usage (Grid’5000, production clusters)
  ▶ Efficiency
  ▶ Reliability
  ▶ Scalability

• Different kind of usage
  ▶ Users: newbies → experts
  ▶ Command line or scripts

• Ecosystem
  ▶ Usage of standard technologies
  ▶ Software/Hardware independent

• Interaction with other technologies
  ▶ Batch scheduler
  ▶ Network isolation
Key features

- Fast and reliable deployment process
- Support of any kind of OS (Linux, BSD, Windows, ...)
- Hardware independent
- Rights management (karights)
  - Integration with batch schedulers
  - Users custom system images
- System images library management (kaenv)
- Statistics collection (kastat)
- Frontend to low level tools
  - reboot (kareboot)
  - power on/off (kapower)
  - serial console (kaconsole)
- Simple: `kadeploy -e debian-base -m node[1-42].domain.local`
- Scriptable deployments (client-server architecture)
Plan

1. Introduction

2. Kadeploy internals
   - Boot over network
   - Deployment process overview
   - Automata for reliable deployment
   - Reboot and Power operations
   - Parallel operations
   - File broadcast methods

3. Example usages at large scale

4. Conclusion
Boot over network

- Based on PXE protocol
- Standard technology, implemented by network cards
- Several BIOS implementations (PXElinux, GPXElinux, iPXE)
- Several methods to retrieve the kernel to boot (TFTP, HTTP)
Deployment process overview

1. Reboot the nodes
   ▶ Create PXE profile files
   ▶ Trigger remote reboot

2. Prepare and install the nodes
   ▶ Boot on the minimal system
   ▶ Prepare nodes
   ▶ Send the system image
   ▶ Install and configure the system

3. Reboot on the installed system
   ▶ Update PXE and Remote reboot
   ▶ Nodes boot on new system
Deployment process overview

1. Reboot the nodes
   - Create PXE profile files
   - Trigger remote reboot

2. Prepare and install the nodes
   - Boot on the minimal system
   - Prepare nodes
   - Send the system image
   - Install and configure the system

3. Reboot on the installed system
   - Update PXE and Remote reboot
   - Nodes boot on new system
Deployment process overview

1. Reboot the nodes
   ▶ Create PXE profile files
   ▶ Trigger remote reboot

2. Prepare and install the nodes
   ▶ Boot on the minimal system
   ▶ Prepare nodes
   ▶ Send the system image
   ▶ Install and configure the system

3. Reboot on the installed system
   ▶ Update PXE and Remote reboot
   ▶ Nodes boot on new system
Deployment process overview

1. Reboot the nodes
   ▶ Create PXE profile files
   ▶ Trigger remote reboot

2. Prepare and install the nodes
   ▶ Boot on the minimal system
   ▶ Prepare nodes
   ▶ Send the system image
   ▶ Install and configure the system

3. Reboot on the installed system
   ▶ Update PXE and Remote reboot
   ▶ Nodes boot on new system
Deployment process overview

1. Reboot the nodes
   ▶ Create PXE profile files
   ▶ Trigger remote reboot

2. Prepare and install the nodes
   ▶ Boot on the minimal system
   ▶ Prepare nodes
   ▶ Send the system image
   ▶ Install and configure the system

3. Reboot on the installed system
   ▶ Update PXE and Remote reboot
   ▶ Nodes boot on new system
Deployment process overview

1. Reboot the nodes
   - Create PXE profile files
   - Trigger remote reboot

2. Prepare and install the nodes
   - Boot on the minimal system
   - Prepare nodes
   - Send the system image
   - Install and configure the system

3. Reboot on the installed system
   - Update PXE and Remote reboot
   - Nodes boot on new system
Deployment process overview

1. Reboot the nodes
   - Create PXE profile files
   - Trigger remote reboot

2. Prepare and install the nodes
   - Boot on the minimal system
   - Prepare nodes
   - Send the system image
   - Install and configure the system

3. Reboot on the installed system
   - Update PXE and Remote reboot
   - Nodes boot on new system
Deployment process overview

1. Reboot the nodes
   ▶ Create PXE profile files
   ▶ Trigger remote reboot

2. Prepare and install the nodes
   ▶ Boot on the minimal system
   ▶ Prepare nodes
   ▶ Send the system image
   ▶ Install and configure the system

3. Reboot on the installed system
   ▶ Update PXE and Remote reboot
   ▶ Nodes boot on new system
Deployment process overview

1. Reboot the nodes
   ▶ Create PXE profile files
   ▶ Trigger remote reboot
2. Prepare and install the nodes
   ▶ Boot on the minimal system
   ▶ Prepare nodes
   ▶ Send the system image
   ▶ Install and configure the system
3. Reboot on the installed system
   ▶ Update PXE and Remote reboot
   ▶ Nodes boot on new system
Automata for reliable deployment

Kadeploy deployment process management:

- Process split in 3 macro steps
- Retries, timeout for each macro step
- Split nodeset if some nodes fails
- Fallback macro steps (Final reboot: Kexec $\rightarrow$ HardReboot)

![Diagram of deployment process]

1. **Macrostep 1 (Min. env. setup)**
   - Configure PXE profiles on TFTP or HTTP server
   - Trigger reboot
   - Wait for nodes to reboot
   - Configure nodes (partition disk, ...)

2. **Macrostep 2 (Env. installation)**
   - Broadcast system image to nodes
   - Do post-installation customization of nodes

3. **Macrostep 3 (Final reboot)**
   - Configure PXE profiles on TFTP or HTTP server
   - Trigger reboot using IPMI or SSH
   - Wait for nodes to reboot on deployed environment
Reboot and Power operations

- Critical part of the software
- Escalation of several level of commands
- Compatible with remote hardware management protocols
- Administrator defined commands
  - soft reboot: direct execution of the reboot command
  - hard reboot: hardware remote reboot mechanism such as IPMI
  - very hard: remote control of the power distribution unit (PDU)
- Managing groups of nodes (e.g. PDU reboots)
- Windowed operations (DHCP DoS, electric hazard)
Parallel operations

Remote commands, TakTuk based
- Hierarchical connections between the nodes
- Adaptative work-stealing algorithm
- Auto-propagation mechanism

File broadcast, Kastafior based
- Chain-based broadcast
- Initialization of the chain with tree-based parallel command
- Saturation of full-duplex networks in both directions
- Other methods available: Chain, TakTuk, Bittorrent
File broadcast methods

P2P file broadcast

Topology aware chained file broadcast
Plan

1. Introduction

2. Kadeploy internals

3. Example usages at large scale
   - Kadeploy on Grid’5000
   - Installing a cloud of VM with Kadeploy

4. Conclusion
Kadeploy on Grid’5000

Grid’5000 deployment’s statistics (since 2009)
- 620 users
- Total: 170,000 deployments
- Average: 10.3 nodes
- Largest: 635 nodes (multi-site)

Benchmark
- 130 nodes of *graphene* from Nancy site
- 5 deployments of a 137MB environment (Small)
- 5 deployments of a 1429MB environment (Big)
Kadeploy on Grid’5000

Grid’5000 deployment’s statistics (since 2009)

- 620 users
- Total: 170,000 deployments
- Average: 10.3 nodes
- Largest: 635 nodes (multi-site)

Benchmark

- 130 nodes of graphene from Nancy site
- 5 deployments of a 137MB environment (Small)
- 5 deployments of a 1429MB environment (Big)

<table>
<thead>
<tr>
<th>Deployment steps</th>
<th>Small</th>
<th>Big</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average time in first and last reboots</td>
<td>3m 58s</td>
<td></td>
</tr>
<tr>
<td>Average file broadcast/decompression time</td>
<td>31s</td>
<td>2m 6s</td>
</tr>
<tr>
<td>Average deployment time</td>
<td>9m 36s</td>
<td>11m 15s</td>
</tr>
</tbody>
</table>
Installing a cloud of VM with Kadeploy

Virtualized infrastructure
- 4000 VMs on 635 nodes (4 Grid’5000 sites)
- 10-20 ms latency
- 1 single virtual cluster

Virtual machines
- 1 VM per core
- 914MB RAM per VM (disk: 564MB, VM: 350MB)
- 3-18 VMs per node

Deployment results
- 430MB environment
- 57 minutes of deployment
- 3838 nodes deployed successfully (96%)
Conclusion

- Scalable OS provisioning for HPC clusters
- Small infrastructure cost
- Efficient and fail-tolerant
- Stable, in production on Grid’5000 since 2009
- Actively supported and developed
Efficient and Scalable Operating System Provisioning with Kadeploy3

Luc Sarzyniec
<luc.sarzyniec@inria.fr>