Centralized Fabric Management Using Puppet, Git & GLPI

Jason A. Smith
Brookhaven National Lab
Motivation

● Several years ago most admin work was either done manually or using home-made scripts
  - ssh in a loop from a management gateway
  - time-consuming, reactive, fire-fighting work
  - little sharing of work & backup expertise
● Standardize and unify our sysadmin work
● Self documenting build and config system
● Audit trail for complete change management
● Separate dev/test/prod env (little extra work)
Components

- Cobbler/RHEV – New system provisioning
- Puppet – Centralized config management
  - Complete service config after provisioning
  - Dashboard monitoring & change auditing
- Git – Puppet catalog repository
  - Distributed development & historical record
- GLPI – Asset mgmt. & node classification
  - Fusioninventory-agent: auto asset inventory
  - ENC uses GLPI, custom DB & dashboard
Provisioning

- Cobbler for hardware installs (poster #539):
  - Powerful Cheetah templating language and config/code reuse with “Snippets”
  - Single ks template used for most systems
  - Specify OS version & arch, network (MAC, IP, etc) & template metadata to install base OS, including fusioninventory-agent & puppet

- RHEV 3.0 for virtual machines:
  - Single template image used for new systems
  - 10 node cluster with 4TB of shared fiber storage
Cobbler Screenshot
RHEV Screenshot
Why Git?

- Distributed version control system
- Faster, completely localized project copies
  - Commits and other work can be done offline
  - Local copy contains complete history
- Reduced single point of repository failure
  - Git can merge changes between many “servers”
- Simple, fast & clean branching (and merging)
  - Branches easily merged with other branches
  - All changes can be treated as branches
Why Puppet?

- Cfengine, puppet, chef, etch, bcfg2, AutomateIt
- Puppet was selected for several reasons:
  - Simple yet powerful DSL (Domain-Specific Lang) & RAL (Resource Abstraction Layer)
  - Explicitly declared dependency graphing model
    - Provides better deterministic state convergence
  - Central config catalog & dependency resolution
    - Better security, conflict resolution & logic analysis
  - Web dashboard, GraphViz config visualization
  - Long history, stable codebase, large user base
GLPI Node Classification

![GLPI Node Classification Screenshot](image)

### GLPI Node Details

- **Name:** gcemaster02
- **Type:** ----
- **Model:** PowerEdge M410
- **Location:** BCF
- **Manufacturer:** Dell Inc.
- **OS:** Red Hat Enterprise Linux Workstation release 6.2 (Santiago)
- **OS Version:** 2.6.32-220.4.2.el6.x86_64
- **Service Pack:** #1 SMP Mon Feb 6 16:39:28 EST 2012
- **Auto update OCSNG:** Yes
- **Last update:** 2012-02-15 16:00 (imported from OCSNG)

**Contact:**
- **User:** [Nobody]
- **Technician in charge of the hardware:** Jason Smith
- **Network:** 192.2
- **Domain:** rcf.lbl.gov
- **Serial Number:** F0STLM
- **Status:** Testing
- **Update Source:** ----
- **Comments:**

- **Server localhost.Agent:** fusioninventory-agent_v2.1.14
- **Last OCS Inventory date:** 2012-03-08 16:16
- **Import date in GLPI:** 2012-03-08 16:20

**Puppet Classes:**
- base
- os::lsivs::fisfs[/var/sysvg:100]
- ganglia::node[/cluster-gcc_servers]
- yarn::configure[/testing/99]
- glpi::mysql::server
- glpi::fusioninventory-agent
- puppet::client
- puppet::server
- puppet::dashboard

**Puppet Parameters:**
- [tables_allow_tcp_list=https://8132.133.199.6.238
238]
238]
- [git_allobj_from=133.199.6.238
238]
- [ssh_root_key_list=jaizuki pryor roat smithj f11skj]

**Update Custom Fields**
Puppet Environments

- 3 puppet environments linked to git branches:
  - Development: extensive module changes
  - Testing: small changes and wider testing
    - Changes staged for more manual tests by wider audience before merging into production
  - Production: main server management
    - Changes must be approved before they are merged into the production branch/environment

- Git branches are automatically sync'ed to puppet environments by push hooks.
  - Also verifies puppet syntax and other checks
Production Approval

Git/Puppet updates to production that are pending approval.

Hello Jason A. Smith, there are currently 2 changes waiting for approval:

<table>
<thead>
<tr>
<th>Date</th>
<th>Age</th>
<th>User</th>
<th>Changes</th>
<th>Changelog</th>
<th>Approve</th>
<th>Reject</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fri Mar 9 15:43:12 2012</td>
<td>2 days</td>
<td>Zhenping Liu</td>
<td>diff</td>
<td>pending-zhliu-cb36590-20120309T204312UTC</td>
<td>merge</td>
<td>delete</td>
</tr>
<tr>
<td>Mon Mar 12 10:18:27 2012</td>
<td>1 minute</td>
<td>Jason A. Smith</td>
<td>diff</td>
<td>pending-smithj4-cb36590-20120312T141827UTC</td>
<td>merge</td>
<td>delete</td>
</tr>
</tbody>
</table>

Instructions:

- The table above lists all changes to puppet's production environment that are currently pending approval.
- The diff link in the Changes column uses the cgit interface to display the detailed changes to all files contained in that pending update.
- The branch link in the Changelog column uses the cgit interface to display the commit history of that branch since it diverged from production.
- Use the merge link in the Approve column to accept the changes and merge them into production.
- Use the delete link in the Reject column to delete the branch if you do not want it merged into production.
- Email notifications are sent after confirmation of the chosen action.
- A side effect of this approval process is that you might see a lot of these old temporary pending branches accumulate in your locally cloned repo. You can clean these up by using the "git remote prune origin" command.
Side-by-side diff

Diffstat (limited to 'gce/glpi/manifests/fusioninventory-agent.pp') (more/less context) (ignore whitespace changes)

RW-RW- gce/glpi/manifests/fusioninventory-agent.pp 24 0
1 files changed, 19 insertions, 5 deletions

diff --git a/gce/glpi/manifests/fusioninventory-agent.pp b/gce/glpi/manifests/fusioninventory-agent.pp
index 93ed6fb...d4f2140 100644
--- a/gce/glpi/manifests/fusioninventory-agent.pp
+++ b/gce/glpi/manifests/fusioninventory-agent.pp
@@ -1,8 +1,22 @@
-class glpi::fusioninventory-agent {
+class glpi::fusioninventory-agent [ $server=undef ] {
+  # Check for parameterized class invocation or global parameter:
+  if ($server) {
+    # Called as a parameterized class, use that server name:
+    $server_name = $server
+  } elsif ($glpi_server) {
+    # Global parameter is set, use that server name:
+    $server_name = $glpi_server
+  } else {
+    # Default puppet server name:
+    $server_name = 'puppet.racl.bnl.gov'
+  }
+  # Install the fusioninventory-agent package:
+  package { 'fusioninventory-agent':
+    - ensure => installed,
+    - require => Yumrepo[ 'base' ],
+    + ensure => latest,
+    + schedule => daily,
+    + require => Yumrepo[ 'base' ],
+    + notify => Service[ 'fusioninventory-agent' ],
+  }
+}
Puppet Dashboard

**Group: Atlas dCache**

**Parameters**
- No parameters

**Groups**
- No groups

**Derived groups**
- No derived groups

**Daily run status**
Number and status of runs during the last 30 days:

**Nodes for this group**

<table>
<thead>
<tr>
<th>Hostname</th>
<th>Source</th>
<th>Latest report</th>
</tr>
</thead>
<tbody>
<tr>
<td>dsmr02.usatlas.bnl.gov</td>
<td>dsmr02.usatlas.bnl.gov</td>
<td>2012-03-16 16:46 EDT</td>
</tr>
<tr>
<td>ddocap02.usatlas.bnl.gov</td>
<td>ddocap02.usatlas.bnl.gov</td>
<td>2012-03-16 16:46 EDT</td>
</tr>
<tr>
<td>dsmr05.usatlas.bnl.gov</td>
<td>dsmr05.usatlas.bnl.gov</td>
<td>2012-03-16 16:46 EDT</td>
</tr>
<tr>
<td>ddocap08.usatlas.bnl.gov</td>
<td>ddocap08.usatlas.bnl.gov</td>
<td>2012-03-16 16:31 EDT</td>
</tr>
<tr>
<td>ddocap06.usatlas.bnl.gov</td>
<td>ddocap06.usatlas.bnl.gov</td>
<td>2012-03-16 16:20 EDT</td>
</tr>
</tbody>
</table>
Puppet Config & Scalability

- Still using 2.6.16 on RHEL5 with ruby 1.8.5
  - testing upgrade to 2.7 on RHEL6 with ruby 1.8.7
- Apache with Phusion Passenger (mod_rails)
- Queue daemon with activemq for fast DB updates of storeconfigs
- Over 2k agents currently using puppet
- Noticed MySQL errors with inventory service enabled at a rate >= about 1 client/second
- Tomcat/JRuby in future for improved scalability
Future Plans

● Change Management
  - Policy & procedures used to control changes made to production systems (ITIL, DevOps).
  - Changes made only during official windows.
  - Absolutely no unauthorized changes, no “cowboy” type behavior tolerated.
  - Use testbed environment to test changes before putting them into production.
    • Create replica of prod using VMs for auto-tests
  - Tools like Puppet, Git & GLPI can help make changes and keep a historical change record.
Automated Validation

- Add a new “validation” git branch & puppet environment
  - Contents: production with all changes currently pending approval automatically merged in
- Replica of all critical production services using RHEV VMs
- Automated testing of production and proposed changes using puppet agent runs and nagios monitoring of all Vms to validate that all production systems still work as expected
Why do it?

- Uncontrolled change can work sometimes, but often cause self inflicted problems and future firefighting episodes & upgrade nightmares.
- Stop duplicating work and effort, standardize.
- Stop making time consuming manual changes.
- Without it, servers become like snowflakes: they may all start out identical, but over time, config drift eventually makes each one unique.
Benefits

- Shift staff time from perpetual reactive firefighting mode, that often only addresses the symptoms, to more proactive work, that addresses the root causes of problems (fire prevention).

- Repeatable and standard build & config process means it is often faster and easier to rebuild problematic servers, rather than waste hours or days troubleshooting problems.
References

- Cobbler: http://cobbler.github.com
- RHEV: http://www.redhat.com/products/virtualization
- FusionInventory: http://fusioninventory.org
- GLPI: http://www.glpi-project.org
- Git: http://git-scm.com
- Puppet: http://puppetlabs.com
- Email: smithj4@bnl.gov