LCOGT Server Deployment and Monitoring

An Adventure in Automation

LCOGT.net

Ira Snyder
System Administrator
About Me

Degree in Computer Science

I have worked in Astronomy for my entire career, currently for LCOGT, and before that for Caltech, at the Owens Valley Radio Observatory.

I’ve done everything from Linux device drivers, to array control software, to system administration work.

I have a cat who parties too much.
Overview

- Introduce automation tools for OS installation and configuration
- Describe the ideal world of software deployment
- Describe LCO monitoring system

I want to convince you that you should invest the time to learn automation tools, so that you can spend less time troubleshooting problems, and more time doing science.
LCO Computing Overview

- LCO operates 19 telescopes at 8 observatories around the globe
- Only two of these observatories have full time staff
- We run about 160 Linux computers (not counting user desktops!)
- Our IT team consists of two people, both based at our headquarters

This is an awful lot of computing power to manage!
Questions for the Audience!
OS Deployment Tools

- **Kickstart** - automated OS installation tool
  - Supports Red Hat, CentOS, Debian, Ubuntu, etc.

- **Puppet** - system configuration and automation tool
  - Supports Linux and UNIX (most variants), as well as Windows
  - Much easier than writing shell scripts for the same tasks!

Use Kickstart to install a minimal operating system, then use Puppet to handle the heavy lifting of installing packages, modifying configuration files, and starting services.
Benefits of Automated OS Deployment

- Consistency!
- No out of date documentation on a wiki somewhere
- Change tracking (Who made a change, and why did they make it?)
- Trivial to reproduce a machine for testing, or for disaster recovery
<table>
<thead>
<tr>
<th>Site</th>
<th>MAC</th>
<th>Hostname</th>
<th>IP Address</th>
<th>Operating System</th>
<th>Boot Mode</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>bpl</td>
<td>00:30:48:f1:6d:6c</td>
<td>ccl.bpl.lco.gtn</td>
<td>dhcp (eth0)</td>
<td>CentOS 7 x86_64</td>
<td>local</td>
<td><img src="Actions.png" alt="Actions" /></td>
</tr>
<tr>
<td>bpl</td>
<td>00:23:7d:a4:fb:a4</td>
<td>inst.1m0a.doma.bpl.lco.gtn</td>
<td>10.7.11.64 (eth0)</td>
<td>CentOS 7 x86_64</td>
<td>local</td>
<td><img src="Actions.png" alt="Actions" /></td>
</tr>
<tr>
<td>bpl</td>
<td>00:19:bb:38:dc:74</td>
<td>inst.igla.bpl.lco.gtn</td>
<td>10.7.23.64 (eth0)</td>
<td>CentOS 7 x86_64</td>
<td>local</td>
<td><img src="Actions.png" alt="Actions" /></td>
</tr>
<tr>
<td>bpl</td>
<td>00:25:90:2d:18:82</td>
<td>pubsubdb.bpl.lco.gtn</td>
<td>dhcp (eth0)</td>
<td>CentOS 7 x86_64</td>
<td>local</td>
<td><img src="Actions.png" alt="Actions" /></td>
</tr>
<tr>
<td>coj</td>
<td>00:25:90:82:59:4c</td>
<td>core1.coj.lco.gtn</td>
<td>dhcp (eth0)</td>
<td>CentOS 7 x86_64</td>
<td>local</td>
<td><img src="Actions.png" alt="Actions" /></td>
</tr>
<tr>
<td>coj</td>
<td>00:25:90:2d:1e:4e</td>
<td>core2.coj.lco.gtn</td>
<td>10.10.8.16 (eth0)</td>
<td>CentOS 7 x86_64</td>
<td>local</td>
<td><img src="Actions.png" alt="Actions" /></td>
</tr>
<tr>
<td>cpt</td>
<td>00:1b:78:cf:4f:a8</td>
<td>inst.1m0a.doma.cpt.lco.gtn</td>
<td>10.8.11.64 (eth0)</td>
<td>CentOS 7 x86_64</td>
<td>local</td>
<td><img src="Actions.png" alt="Actions" /></td>
</tr>
<tr>
<td>cpt</td>
<td>00:19:bb:38:fc:e4</td>
<td>inst.1m0a.domb.cpt.lco.gtn</td>
<td>10.8.13.64 (eth0)</td>
<td>CentOS 7 x86_64</td>
<td>local</td>
<td><img src="Actions.png" alt="Actions" /></td>
</tr>
<tr>
<td>cpt</td>
<td>00:1b:78:ac:ee</td>
<td>inst.1m0a.domc.cpt.lco.gtn</td>
<td>10.8.15.64 (eth0)</td>
<td>CentOS 7 x86_64</td>
<td>local</td>
<td><img src="Actions.png" alt="Actions" /></td>
</tr>
</tbody>
</table>
# Kickstart Generator Web Interface

## Edit existing machine

<table>
<thead>
<tr>
<th>Required Information</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hostname</td>
<td>pubsubdb.bpl.lico.gfn</td>
</tr>
<tr>
<td>MAC Address</td>
<td>00:25:90:2d:18:32</td>
</tr>
<tr>
<td>Operating System</td>
<td>CentOS 7 x86_64</td>
</tr>
<tr>
<td>Boot mode</td>
<td>Boot from local disk</td>
</tr>
</tbody>
</table>

### Disk Configuration

| Partition Schema           | LCOGT - LCOGT default partition scheme |

### Network Configuration

<table>
<thead>
<tr>
<th>Device</th>
<th>eth0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>dhcp</td>
</tr>
</tbody>
</table>
Example of Puppet Code

```puppet
package { 'ntp':
  ensure => installed,
}

file { '/etc/ntp.conf':
  ensure => file,
  owner => 'root',
  group => 'root',
  mode  => '0644',
  source => "puppet:///modules/ntp/ntp.conf",
  require => Package['ntp'],
  notify => Service['ntp'],
}

service { 'ntp':
  ensure => running,
  enable => true,
}
```
Example: building an instrument control computer

Before this work was started, the way to build a new instrument control computer was:
- Install CentOS (using a CD!)
- Examine a working instrument control computer somewhere else on the network
- Copy stuff until it works

Now it is this Puppet code (and that's it!):

```puppet
node 'inst.1m0a.doma.elp.lco.gtn' {
  network::static {
    primary:
    ipaddr => '10.9.11.64',
  }

  class { '::telescopecontrolsystem::instruments':
    instruments => ['ef01',
      'fl05',
      'kb22',
    ],
  }
}
```
Key Points

- Start small. Grow your automation over time as you get more experience.
- Introduce automation to existing systems to help you manage them.
- Consistency will help you spend less time troubleshooting problems.
Software Deployment
Goals for the Software Deployment Process

The goal is to be able to repeatedly deploy an identical set of software to the target machine(s).

- If it runs on a developer machine, it will run in production
- Easy to roll back to known working versions when there are problems
- Great for testing purposes (Can easily reproduce bugs)
Software Deployment, the Present

Our current strategy is to automatically build binaries (Java bytecode) at each commit into version control. The latest commit is automatically deployed each day at the telescope’s local noon. If a problem occurs, a quick revert in version control, and a manually triggered redeploy, will get the software working again.

This isn’t an ideal system. Completely repeatable and reproducible software deployments are still a work in progress at LCO. We are still learning!
Software Deployment, the Future

In order to make software deployment more repeatable, we are exploring several different strategies.

- Build everything needed to run a telescope (or observatory) into a single archive, and deploy that
- Use Docker to build a single immutable image of both our OS and code
Monitoring the Network
Monitoring Tools

Tools in use at LCO:

- Nagios using Check_MK with Thruk web interface (hardware monitoring)
- Elasticsearch / Logstash / Kibana (software logs, FITS headers)
- OpenTSDB / Grafana (storing and graphing metrics)

Don’t be afraid to use different tools! Use the best software for the job at hand.
Monitoring Benefits

- Hardware monitoring will help you find problems before they happen
  - Disks becoming full
  - Hardware failures (disks accumulating errors, RAID card batteries, etc.)
- Having visibility into the software often finds hidden bugs
- Good visibility often finds bugs before they become a serious problem
Example of finding a hidden bug
Monitoring Gotchas

● Beware of alerting fatigue!
● Question the purpose of every new alert you add.
● Unless there is an action that can be taken by the receiver, they shouldn’t be receiving the alert.
● This requires constant tweaking. As your software evolves and gets better, you should changing the alerts you generate.
Automation will help you do more science faster!
Thanks!

Contact us:

Las Cumbres Observatory
6740 Cortona Dr. Ste. 102
Goleta, CA 93117

Ira Snyder
isnyder@lcogt.net