Bcfg2 Architecture and Deployment Overview

Narayan Desai
MCS Division, Argonne National Lab
desai@mcs.anl.gov

Overview

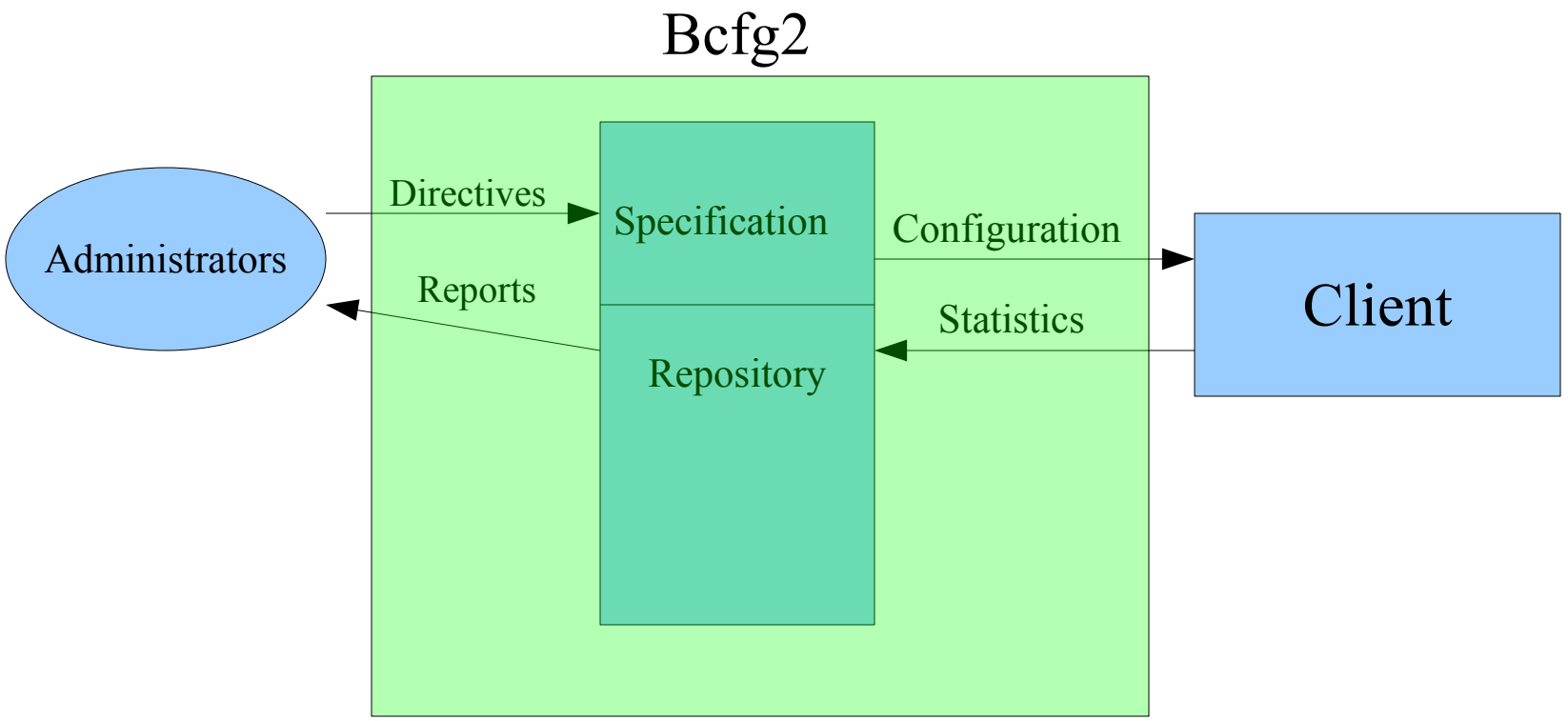
- History
- Bcfg2 Architecture
- Model
- Tool Contrast
- Constructs
  - Specification
  - Metadata
  - Generators
- MCS Deployment
  - Core
  - Client Build Process
  - Care and Feeding
  - Reports
- Misc
- Future Improvements
- Questions

mcs

History

- Long time configuration management tool users and developers
  - Cfg-get
  - Sanity
- Bcfg (1) initial motivations
  - Lack of buy-in from administrators with current tools
  - Poor tool user interfaces
  - Unidirectional information flow
  - Scalability to large groups of administrators and discrete configurations
- Three years later
  - Bcfg2 is (relatively) feature complete
  - Actively deployed thoughout MCS
    - ~550 machines

Bcfg2 Architecture



Model

- All configuration can be reduced to filesystem artifacts
  - Packages
  - Services
  - Configuration Files
  - etc
- Configuration specification is comprehensive
  - All client configuration is represented
  - “Extra” configuration can be detected
- The specification is defined to be correct
  - Deviations are detecting using client execution statistics

Contrast with Other Tools

- CFEngine
- Kickstart
- System Imager
- FAI
- Jumpstart

Specification

- Describes clients in terms of metadata
  - Hostname, image, profile, classes
- Includes all information about configuration inventory and contents (Software versions, configuration file contents)
- Describes network
  - As it ***should*** be
- Consists of a filesystem hierarchy on disk

Metadata

- Describe clients in terms of base image and role (profile)
- Profiles consist of classses providing functionality
- Classes can be reused in a “cookbook” fashion
- Leads to a highly modular setup
- Implies the meaning of the file repository
- Provides all differentiation of configuration

Generators

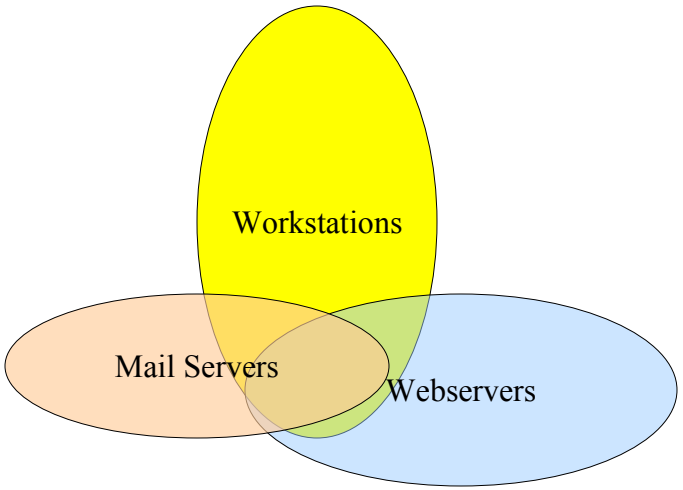
- Plug in interface that allow custom generation of client configuration
- Allows for representation of rule-based configurations
- Implements “smart” configurations
  - Automatically updating
- SSHbase is the most interesting one
  - SSH key management
  - Coordinates known_hosts file
  - Automatically generates new keys for new clients
  - Propagates updated known_hosts files when needed

The MCS Deployment

- 3 Servers
  - Core - ~ 100 machines
  - Chiba - ~250 machines
  - Teragrid - ~200 machines
  - Blue Gene/L and Jazz in progress
- Run independently due to differing administrative groups
- Each uses System Imager as a bootstrap to perform base system setup
- 10 full-time administrators plus 2-4 students work together on the specification, across the three installations

Core Overview

- All profiles share a common base
- Multiple profiles can share the same functionality
  - NIS common to many
  - NFS home directories only needed on some
  - Less services are active on servers than on workstations
  - X installed on workstations
  - Conduits only point at servers
  - Some machines only allow ssh-key based authentication
- Central decisions allows audits and coordination of policy



Client Build Overview

- System Imager installs base system
- Bcfg2 invoked with a profile specification
  - Can build a machine of any profile from build floppy menu
- All new client registration occurs automatically
  - Metadata setup
  - SSH Key generation
  - More possible with improved generators
- All software is installed and patched to the latest rev before initial system boot – systems come up secure the first time
- Systems are configured to periodically update and reconfigure

Care and Feeding

- Bcfg2 server operation is largely automatic
  - Specification changes can occur automatically, if desired
  - Security updates can be automatically deployed, if desired
- Daily interactions occur through reports
- Clients can run daily in dry-run mode
  - While no changes are performed, statistics updates occur
  - Increase the chances of client misconfiguration

Reports

- Reports provide bird's eye view of entire network
- Can describe individual machine configuration states
  - Clean
  - Dirty
  - Not recently updates
- In the case of dirty clients, a list of incorrect configuration elements is provided
- A list of updated configuration elements can also be provided
- Reports can be published in a variety of ways
  - Email
  - Web
  - RSS

Misc

- Once a set of configurations have been represented in the Bcfg2 repository, large scale changes can be efficiently performed
  - Building a SLES9 image based on a SLES8 one on teragrid took an afternoon, and was largely correct.
- Bcfg2 currently supports Redhat and Debian

Future Improvements

- Automatic generation of audit reports
- Solaris support
- OSX support
- Templating generator
- Host management generator
- User management generator

Questions?

- Well?