Experiences running a production Puppet infrastructure @CERN

Ben Jones

HEPiX Bologna Spring 2013

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Infrastructure

- Foreman
  - kickstart, ENC, report visualisation
- Multiple puppet masters
  - load balanced (simple for now)
- PuppetDB
  - stored configs, complex manifests, data mining
- hiera data separation
- CERN CA
- git managed manifests
  - split into “modules”, “hostgroups”, “hieradata”

Production service

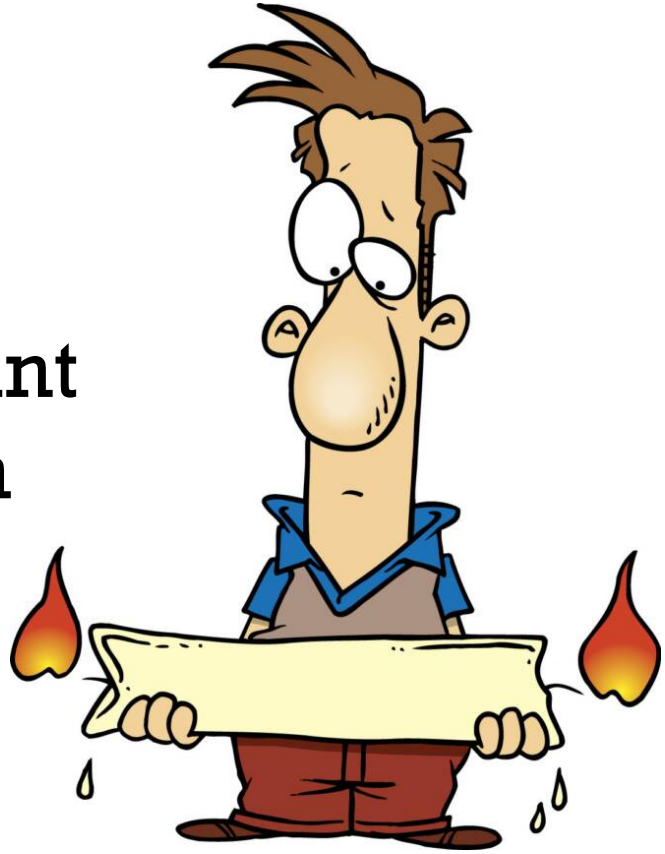
- Customers running production services using the infra
  - openstack, cvmfs, lxplus
  - modules appearing for bddi, MyProxy, glexec WN, argus, voms, castor
- Plant still modest but increasing
  - ~1350 clients, 2/3 physical
  - numbers will change soon as batch comes online
- Lots of changes in this year due to production experience

Don't turn your computer centre into a giant DDOS machine

Facter

- Facter a great tool to abstract common information for a machine
  - virtual fact good example of hidden complexity (180 loc)
- Unlike what we were used to in Quattor, machine facts can inform configuration
- Custom facts are simple to write… too simple!
  - LANDB integration for location, network info, owner etc
  - 1000s of clients * short runinterval == unhappy LANDB
- Solutions: cache or use masters
  - cache directly in facter code or wait for facter 2.0
  - use custom functions

Decide on one entry point to apply configuration

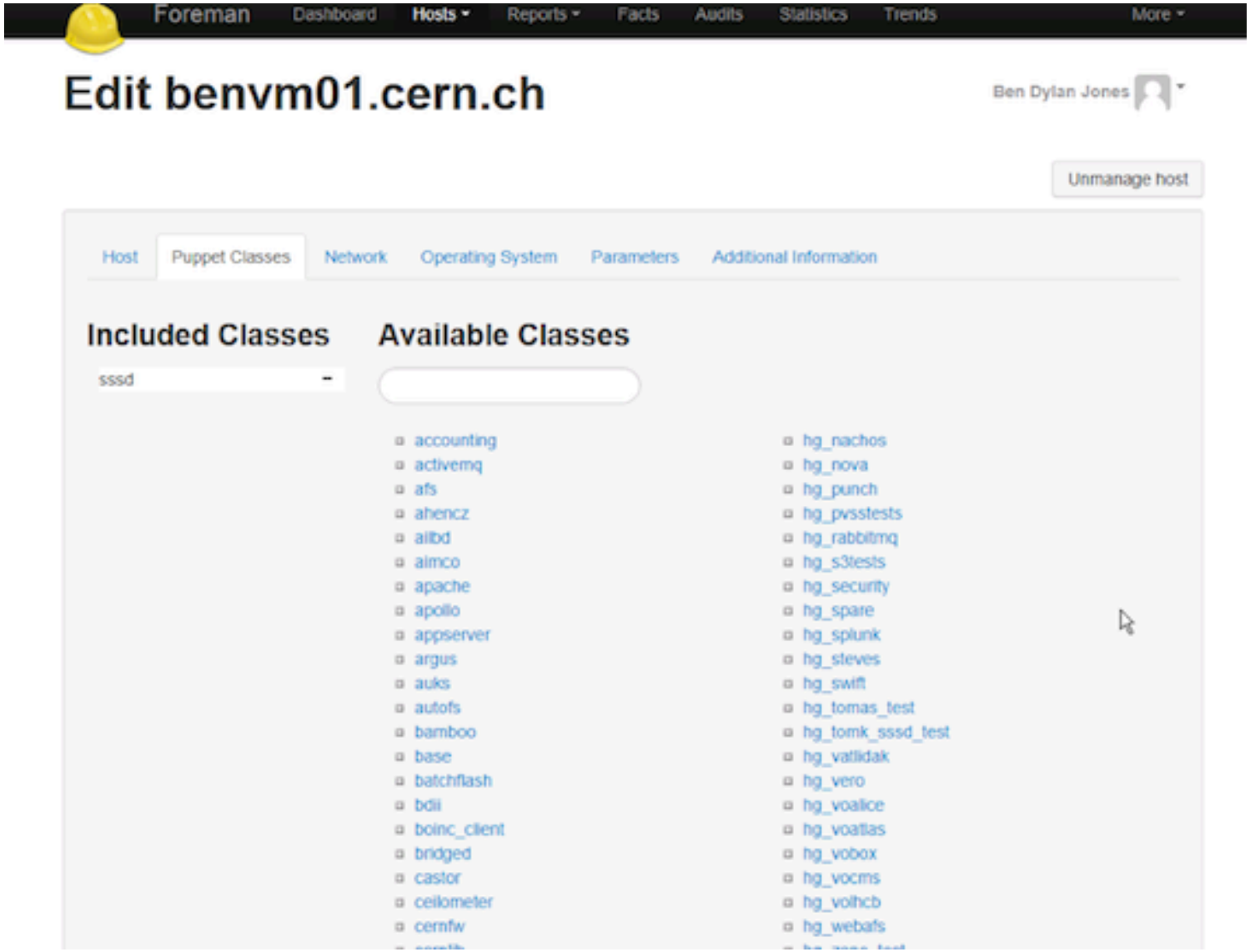


the foreman

- Foreman useful tool for many tasks
  - provisioning (physical and virtual)
  - configuration (ENC)
  - monitoring (reports)
- Important to consider extent to use the ENC
- Environment best defined by ENC
  - no choice anyway in puppet 3
- Hostgroups essential in ENC
  - hostgroups similar to quattor clusters
    - castor/server/diskserver/atlas/t0atlas
  - hiera data assigned to hostgroup, so machine cannot assert membership

foreman classes

CERN IT Department

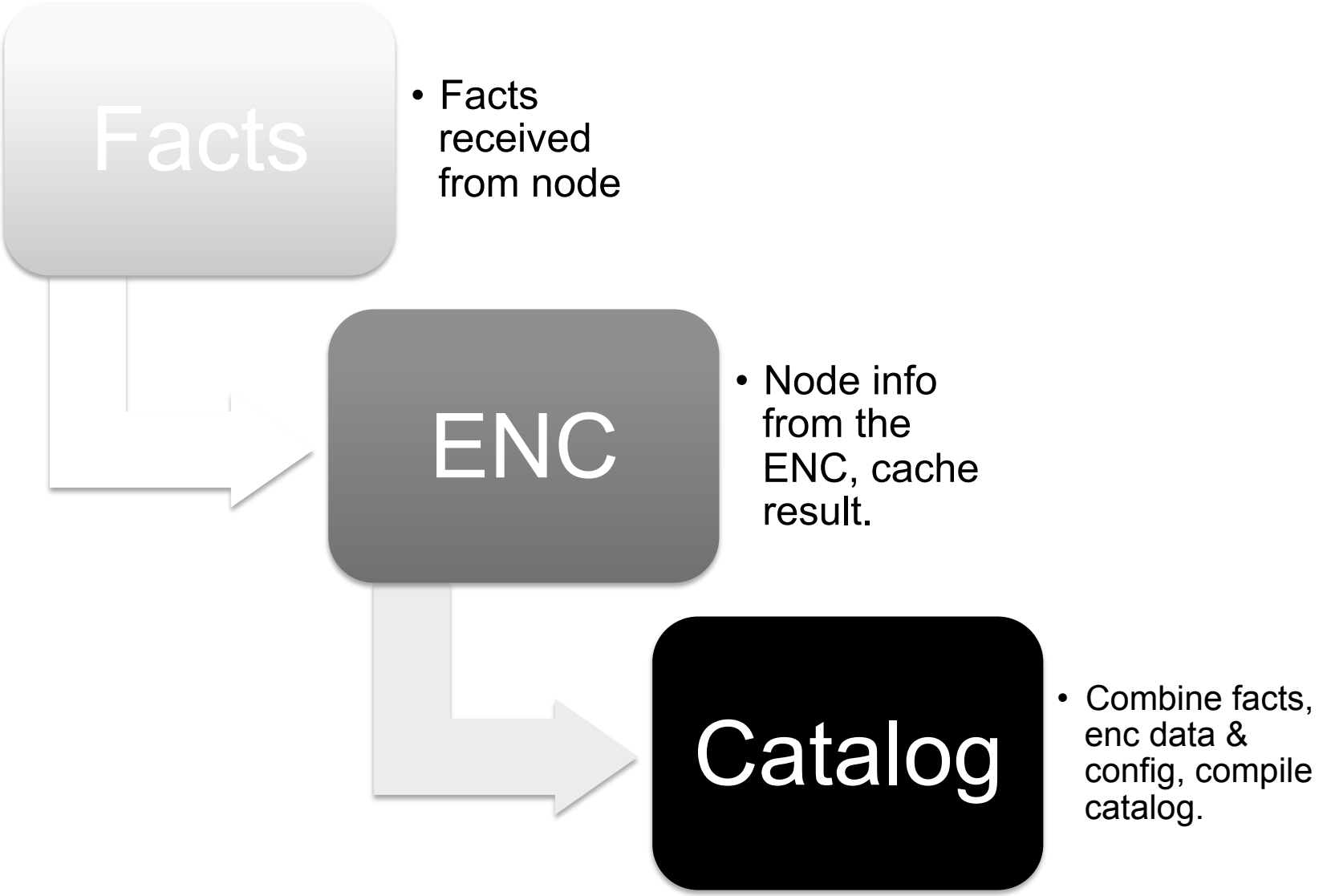


module imports

- We switched off foreman classes.
- History of changes not in git
- Either no parameters, or class{} declarations
  - most non-trivial modules require some parameters
- Class / environment import takes time
- Nested hostgroup module definition was useful
  - we implemented a puppet include_if_exists (on github)
  - hostgroup module tree included if in hierarchy
  - castor/headnode/c2repack hostgroup would load:
    - hostgroups/hg_castor/manifests/init.pp
    - hostgroups/hg_castor/manifests/headnode/c2repack.pp

Don't let your machines become time travellers.

Compile process



- Facts received from node
- Node info from the ENC, cache result.
- Combine facts, enc data & config, compile catalog.

Stale definitions

- With multiple puppet masters, if foreman is unavailable the cache can be stale depending on which master contacted.
- A change of hostgroup or environment can be painful
- Obvious answer to not cache, or to share / update cache
- Lesson is components are horizontally scalable, but not always case that this is robust

Try not to make all your secrets public

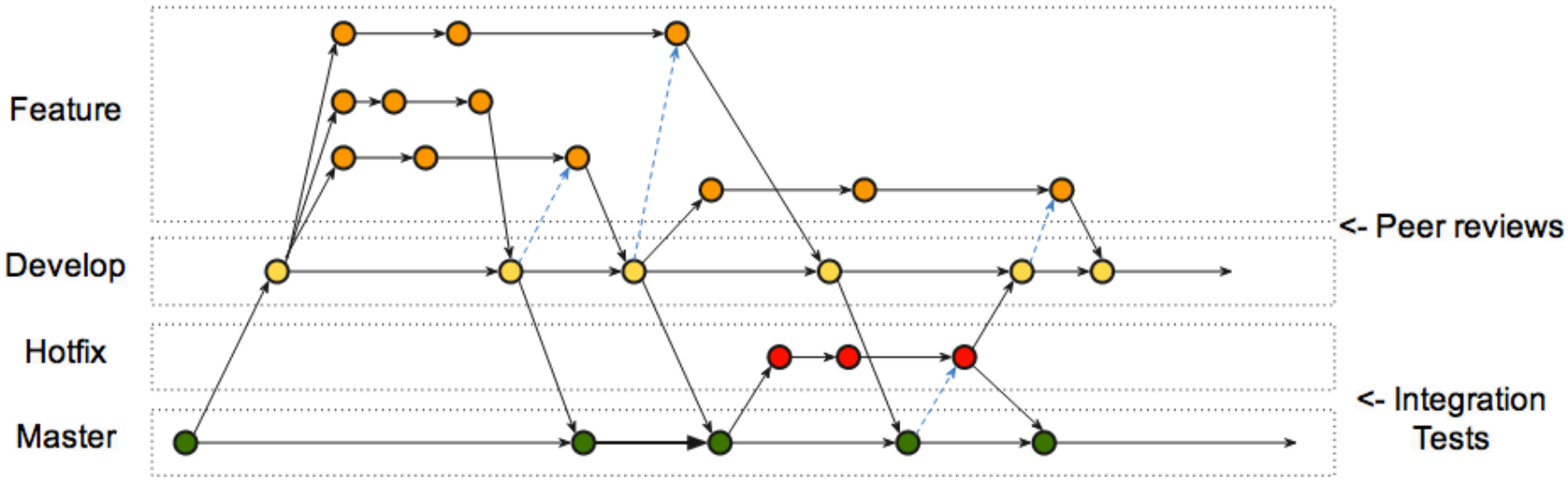
PuppetDB

- Collects facts & catalogs generated by puppet
- Supports exported resources & inventory service
- Can be directly queried in manifests
  - github.com/dalen/puppetdbquery
- Wider ad-hoc usage useful
  - git hooks
  - “who’s using X”
- Dangerous!
  - /commands
  - “secret” data

PuppetDB proxy

- Whitelist traffic from puppet masters & lb
- Refuse all /commands/ from load balancers
- Use resource tags to filter “secrets”


```
{
  "parameters" : {
  "entry" : "egroup",
  "k5file" : "/root/.k5login”,},
  "sourceline" : 54,
  "sourcefile" : "/etc/puppet/environments/bejones/modules/kerberos/
manifests/config.pp",
  "exported" : false,
  "tags" : [ "config", "k5entry", "ai-admins", "kerberos::k5entry", "base",
"class", "kerberos", "kerberos::config" ],
  "title" : "ai-admins",
  "type" : "Kerberos::K5entry”,
  "certname" : "benvm01.cern.ch"
}
```



git workflow is easy to get wrong

git workflow

- puppet “dynamic environments”
  - puppet masters pull git repositories, create environment per branch
  - foreman imports environment list
- Managed branches for majority
  - production (master)
  - devel
- Topic branches should be easy to create
- Don’t allow non fast-forward updates to managed branches
  - obviously, who would forget to do that?

git workflow

- Goal is to get safe changes applied as quickly as possible
- Only cherry-picked signed off changes to master
- Fast track for low impact one-module changes, hostgroup & hiera
- More involvement from core team for complex changes
- Future plans to use CI to streamline
  - bamboo (or similar) for system tests
  - canary machines

Community

- "We're not special"
- https://github.com/cernops
- Fork where necessary but always try to return to mainstream
- Would welcome more involvement from HEPiX sites on common problems