Package Inventory
How to manage configuration drifts

Andrea Giardini
Master in Computer Engineering
University of Bologna

in collaboration with:
Configuration Management Team at CERN

April 28, 2015

1 Introduction

The following paper is part of a one year internship at CERN: the student joined the Configuration Team in order to solve the problem of configuration and package drifting.
The software described here is still under active development and it is not yet ready for a public release.

2 Problem description

Three years ago CERN decided to migrate from a traditional IT infrastructure to an Agile one: this migration improved dramatically the resources efficiency and the overall service uptime. Using Openstack and Puppet for machine provisioning and configuration sped up the development cycle and reduced significantly the deployment time.

This allowed also to have a more elastic and flexibile service for the users, that now are able to describe thair own service using a declarative language

(PuppetDSL) which allows them to control and configure a machine in all its aspects.

Moving to an Agile infrastructure, expecially in the CERN datacenter where the number of service managers is particularly high, lead to a management problem: since all the machines are managed automatically by Puppet, the service managers loose control over what is exactly installed in their machines. In particular it is complicated to have an overview of the installed systems and it is difficult to find out if some machines have outdated versions of packages, or configurations that may lead to security issues.

During the last year Puppet has been a great tool for configuring servers but, in some cases, his efficiency has been compromised by human or machine errors. Since service managers are able to freeze their configurations it is difficult to spot misalignment or, in general, out of sync machines. In other cases software errors prevent the machine itself from upgrading: we noticed that the package manager Yum under certain circumstances has problems upgrading or installing new packages due to RpmDB corruption.

Both those situations need to be monitored and fixed but it is not an easy task when dealing with thousands of machines managed by hundreds of administrators.

The idea is to have a centralized database to query in order to have the package status of all the machines and be able to resolve misalignment in a timely manner.

3 Package Inventory

The solution proposed aims to create a centralized database that stores the status of packages of the machines inside the datacenter.

We want to be able to:

- Get the list of packages for a selected machine
- Compare machines that are part of the same hostgroup, grouping them in hosts with same configuration
- Obtain the history of a package for a specified host
- Use puppetDb to query machines from a hostgroup

3.1 Softwares used

- Elasticseach

  It is a search server based on Apache Lucene for distributed, multitenant-capable full-text search engine with a RESTful web interface and schema-free JSON documents.

- Apache Flume

  Flume is a distributed, reliable, and available service for collecting, aggregating, and moving large amounts of log data. It is robust and fault tolerant with tunable reliability mechanisms and many failover and recovery mechanisms.

3.2 Components

Package Inventory has two main components:

3.2.1 Package Reporter

It is installed on every machine, it takes care of getting all the modiified packages by Yum and uses Flume to report them to a central elasticsearch cluster. Every time a new package is installed, removed, upgraded or downgraded using Yum, Package Reporter sends an update to keep the database up to date.

Package Reporter is designed as a standalone software plus a Yum plugin, using this tecnique we are able to report all the packages installed using:

- Yum

  Installing a Yum Plugin and using the post-transaction hook we are able to get the details of the transaction and communicate the new machine status after every package installation.

- Rpm

  Since occasionally it happens that some packages are installed without using Yum, we want to keep track of those packages as well. Using a cronjob we can run Package Reporter once a day and report them as well.

The file containing the packages installed is stored locally in a log file and, on every run, it compares the actual list of packages with the one contained inside the log file, looking for differences. After this operation all the modified packages are reported to elasticsearch using Flume, which provides an additional layer for redundancy and failover.

3.2.2 CLI

This interface to the database allows us to:

- Get list of packages installed in a machine
- Retrive the story of a package - When it has been installed/upgraded/ downgraded or removed
- Compare two machines or a set of machines and highlight the differences
- Compare machines that are part of the same hostgroup
- Verify if a machine has been upgraded after a package update

4 Results

Since the data volume might grow significantly we decided to have a fully customizable Reporter and CLI: the user is able to specify to which elasticsearch the machies must report their data and, in the same way, they can instruct the CLI to query a specified database.

Package Reporter has been already installed on a test evironment with 5 machines (2 Elasticseach + 3 test machines) and seems to perform well. In the following weeks it will be deployed in some QA machines and then, if no bugs are found, it will be installed in the production environment.

5 Conclusions and Future Plans

The plans for the future is to deploy Package Inventory in the machines of the datacenter, and also extend it to manage not only package drifts but also configuration drifts. An open source release is planned as well by the end of the internship.