



The Linux & Open Source Company

Patch reporting

When did I do what?

Jan Bundesmann

THE Linux & Open Source Company!



Consulting



Engineering



Support



Training



Problem

What's the question?

- ▶ Some people want to know when they patched which package on which host.
- ▶ They expect foreman to have this information.

What's the problem?

Foreman / katello only knows the current package status.

Agenda

- 1 My solution
 - Storing information externally
 - All currently installed packages

- 2 Improvements

Agenda

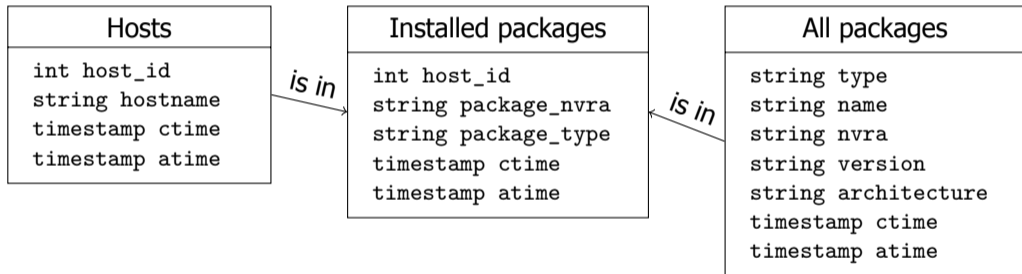
- 1 My solution
 - Storing information externally
 - All currently installed packages

- 2 Improvements

Agenda

- 1 My solution
 - Storing information externally
 - All currently installed packages
- 2 Improvements

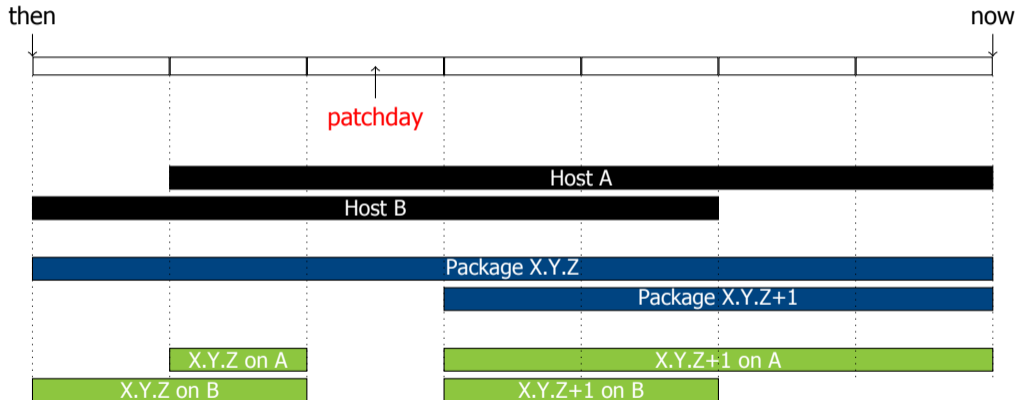
Storing information externally



How to work with this structure

- ▶ Postgres database with 3 tables
- ▶ Content updated regularly
- ▶ Dynamics stored in `ctime` and `atime`
 - ▶ `ctime`: entity appeared the first time
 - ▶ `atime`: entity appeared the last time

Visualize dynamics



Question that can be answered

- ▶ Which packages have been updated since the last patch day?
- ▶ Which packages have been updated within a given time slot?
- ▶ When has a package been update on host XY?
- ▶ Has a security issue on host XY been fixed?
- ▶ Which packages will be updated on the next patch day / upgrade slot?
- ▶ When did the last update on host XY happen?

Example query - packages newly installed after patchday

```
1 SELECT hosts.hostname, pkg_info_1.name, pkg_info_1.version as current_version
2
3 FROM hosts
4
5 JOIN installed_packages AS pkg_currently_installed ON (hosts.hostname = pkg_currently_installed.hostname)
6 JOIN all_packages AS pkg_info_1 ON (
7     pkg_info_1.nvra = pkg_currently_installed.package_nvra and
8     pkg_info_1.type = pkg_currently_installed.package_type
9 )
10
11 JOIN installed_packages AS pkg_ever_installed ON (hosts.hostname = pkg_ever_installed.hostname)
12 JOIN all_packages AS pkg_info_2 ON (
13     pkg_info_2.nvra = pkg_ever_installed.package_nvra and
14     pkg_info_2.type = pkg_ever_installed.package_type and
15     pkg_info_1.name = pkg_info_2.name
16 )
17
18 WHERE hosts.atime >= '{{ recent }}' /* host still exists */
19 AND hosts.ctime <= '{{ last_patchday }}' /* host already existed */
20 AND pkg_currently_installed.ctime >= '{{ last_patchday }}' /* installed after last patchday */
21
22 GROUP BY (
23     hosts.hostname,
24     pkg_info_1.name,
25     pkg_info_1.version
26 )
27 HAVING COUNT(*) = 1
28
29 ORDER BY hosts.hostname, pkg_info_1.name, pkg_info_1.version;
```

Agenda

- 1 My solution
 - Storing information externally
 - All currently installed packages
- 2 Improvements

First attempt: API

1. Query list of hosts
2. Query list of available packages
3. Query list installed on hosts

Second attempt: Report templates

```
1 <%#
2 name: All Hosts - Package Collector
3 model: ReportTemplate
4 %>
5 <% load_hosts(search: '*').each_record do |host| -%>
6 <%   host.installed_packages.each do |pkg| -%>
7 <%
8   architecture = pkg.nvra.gsub("#{pkg.name}-", "").match(/(?<architecture>[^\.]*)$/)
9   version = pkg.nvra.gsub("#{pkg.name}-", "").gsub(/\.#{architecture}/, "")
10 -%>
11 <%-   report_row(
12     'hostid': host.id, 'hostname': host.name, 'packagetype': 'rpm',
13     'packagename': pkg.name, 'packagenvra': pkg.nvra,
14     'packageversion': version, 'packagearchitecture': architecture
15   )-%>
16 <%   end -%>
17 <%   host.installed_debs.each do |pkg| -%>
18 <%-   report_row(
19     'hostid': host.id, 'hostname': host.name, 'packagetype': 'deb',
20     'packagename': pkg.name,
21     'packagenvra': "#{pkg.name}-#{pkg.version}.#{pkg.architecture}",
22     'packageversion': pkg.version, 'packagearchitecture': pkg.architecture
23   )-%>
24 <%   end -%>
25 <% end -%>
26 <%= report_render -%>
```

Problems and success

- ▶ Report templates much faster than direct API calls
- ▶ Doesn't work with safemode rendering enabled, currently
- ▶ Unify format for RPM and DEB content
- ▶ Works on my system, doesn't work for large number of hosts

Some nice implementation tweaks

- ▶ Streamline setup
 - ▶ Provision separate collector host using Ansible role, template, Application Centric Deployment
 - ▶ Define a single report template (in Ansible?)
- ▶ Easy usage
 - ▶ Playbook to collect data
 - ▶ Playbook to generate report

Agenda

- 1 My solution
 - Storing information externally
 - All currently installed packages

- 2 Improvements

Possible improvements

- ▶ Limit query to Host Collections or Host Groups
- ▶ Query PostgreSQL directly
- ▶ Time series database
- ▶ Metrics endpoint

Thank you for your attention!

Possible improvements

- ▶ Limit query to Host Collections or Host Groups
- ▶ Query PostgreSQL directly
- ▶ Time series database
- ▶ Metrics endpoint

Thank you for your attention!