Security Automation Challenges
Open Source

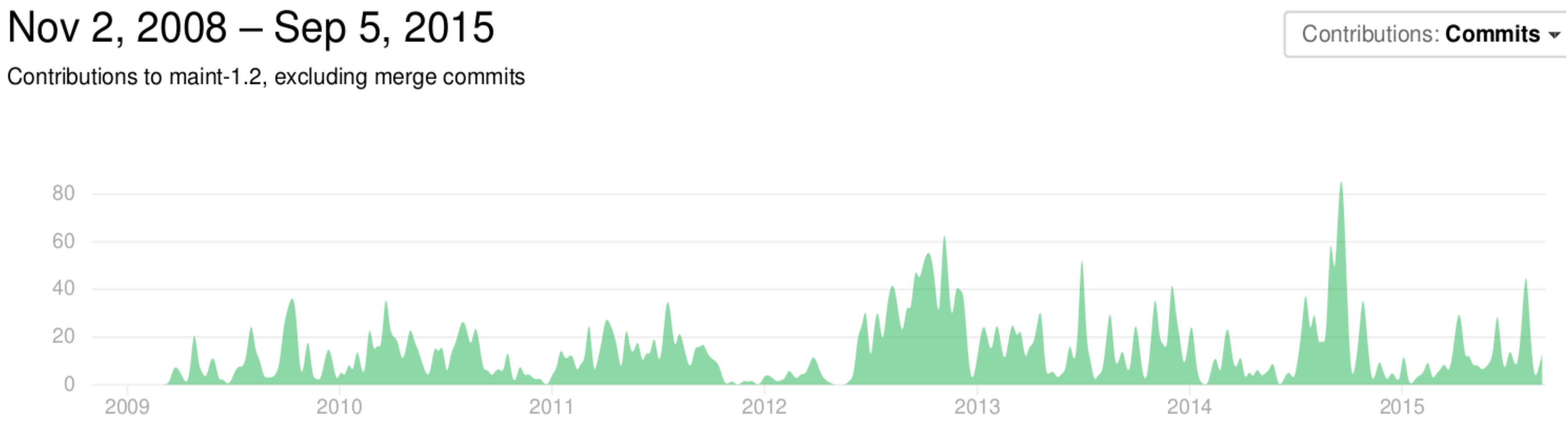
Steve Grubb
Red Hat

Open Source Tools

- OpenSCAP
  - Certified on RHEL5
  - Integrated with Satellite
  - Available on all major linux distributions and Solaris
  - Community working at github
    - 5840 code commits
    - 28 contributors world wide

https://github.com/OpenSCAP/openscap

Code contributions



Nov 2, 2008 – Sep 5, 2015

Contributions to maint-1.2, excluding merge commits

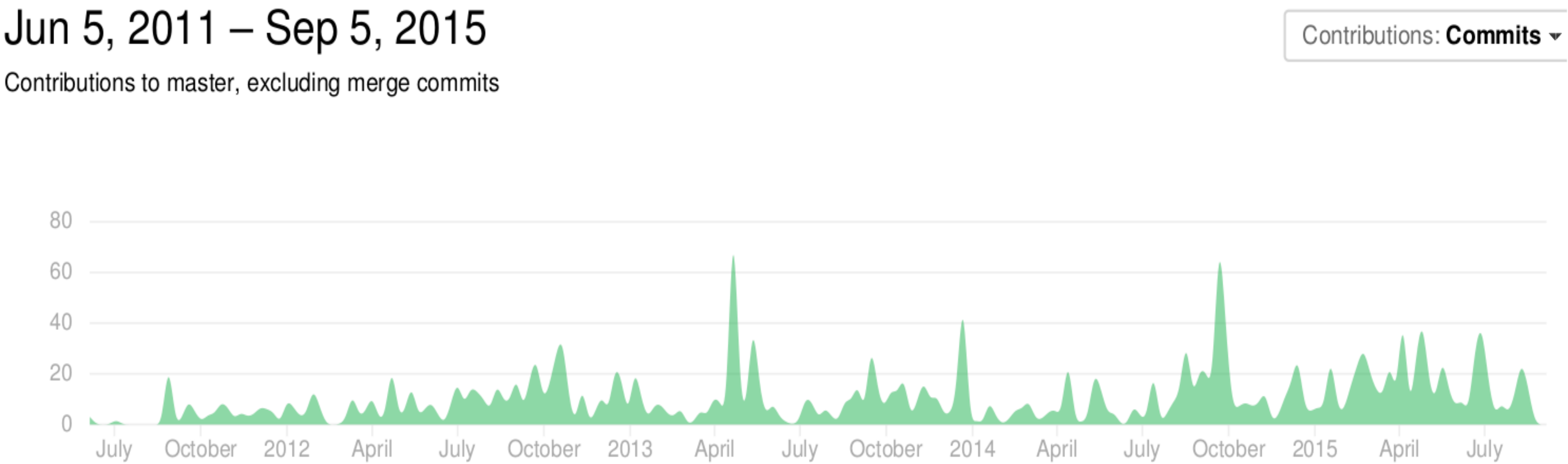
Contributions: **Commits**

Open Source Content

- SCAP Security Guide
  - 3852 contributions
  - 31 contributors
  - 10 guides
    - Chromium, Fedora, Firefox, JBossEAP5, JBossFuse6, JRE, OpenStack, RHEL, RHEVM, Webmin

https://github.com/OpenSCAP/scap-security-guide

Content Contributions



Jun 5, 2011 – Sep 5, 2015

Contributions to master, excluding merge commits

Issues

- Open Source starved of content
  - Creating it is hard
  - Limited to Operating System and few major apps
  - Hard to learn
    - Online tutorial, book, class?
  - How do new users go about finding content?
  - How do new users even locate resources available?
  - People want more content
    - Cisco routers, Juniper routers, Microsoft, application servers, Tomcat, splunk, ...

Issues

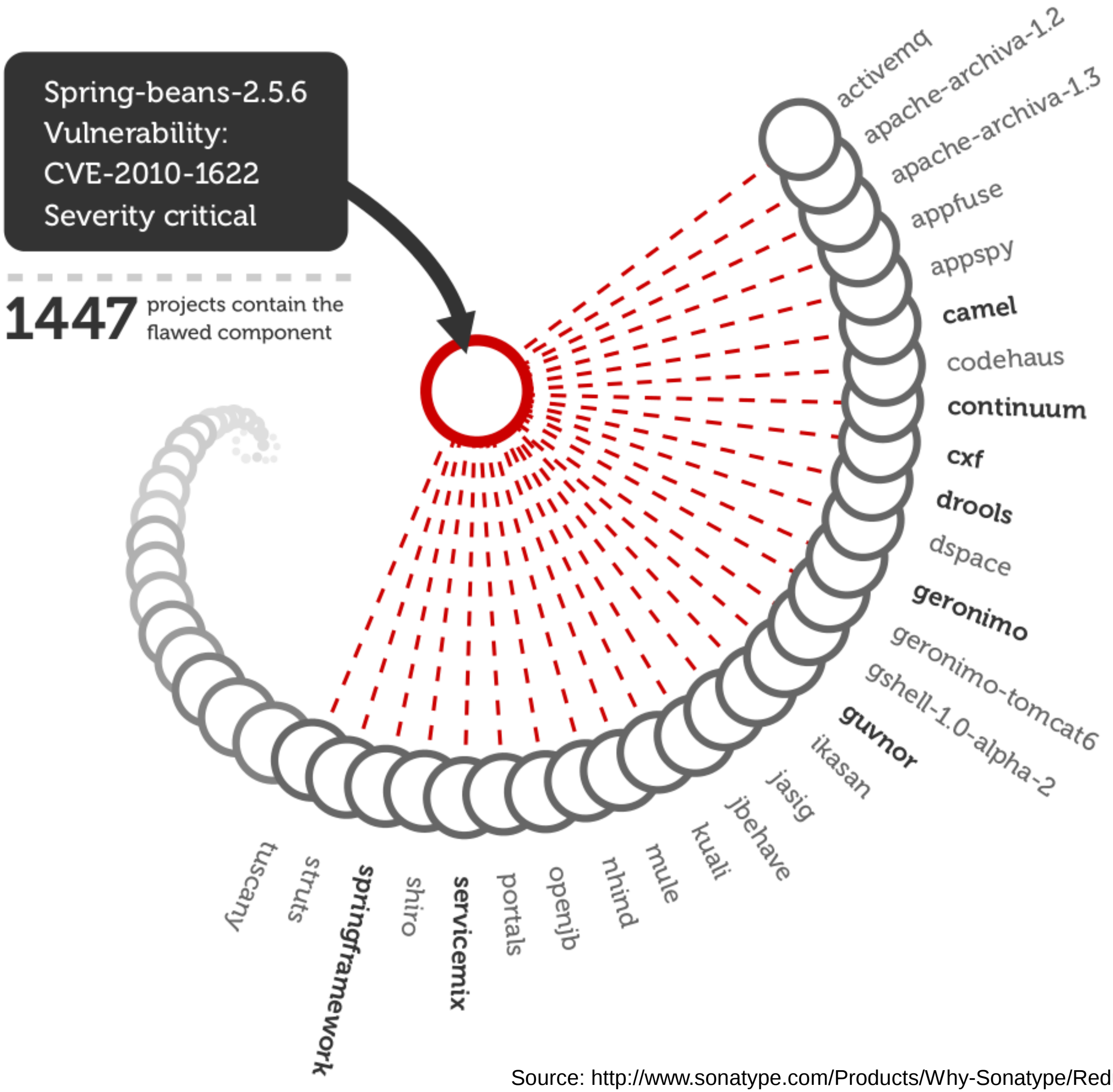
- Open source is about sharing
  - People grab SSG which has CCE's for Red Hat
    - Now the RH CCE's are used everywhere
    - Distributions don't know the rules
    - CCE's should be assigned to just a control and its number re-used on any OS even different releases of RHEL
  - Proprietary extensions
    - Not open source friendly

SWID

SWID

- Idea is gaining supporters in open source world
- Initially hard to sell in the open source world
  - Have to pay to see the ISO standard
  - Cuts off competition
  - Best ideas come from competition
- NIST IR 8060 finally opens things for the open source community
  - Seeing an uptick in discussions around supporting it now

Example: CVE-2010-1622 in Spring



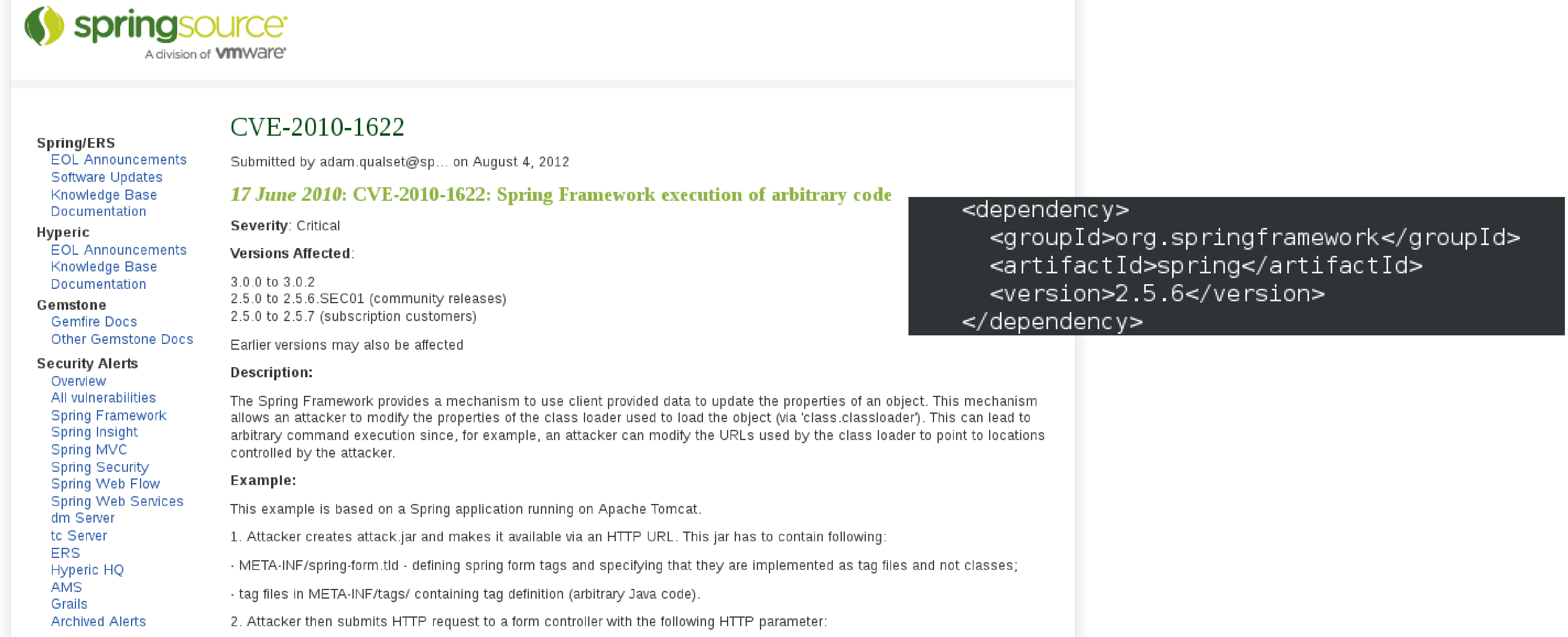
Spring-beans-2.5.6
Vulnerability:
CVE-2010-1622
Severity critical

1447 projects contain the flawed component

Source: http://www.sonatype.com/Products/Why-Sonatype/Reduce-Security-Risk/Security-Brief

Example: CVE-2010-1622

- How do you know your application is vulnerable and needs to be recompiled?
- How do you know to update your dependencies?



springsource
A division of vmware

Spring/ERS
- EOL Announcements
- Software Updates
- Knowledge Base
- Documentation

Hyperic
- EOL Announcements
- Knowledge Base
- Documentation

Gemstone
- Gemfire Docs
- Other Gemstone Docs

Security Alerts
- Overview
- All vulnerabilities
- Spring Framework
- Spring Insight
- Spring MVC
- Spring Security
- Spring Web Flow
- Spring Web Services
- dm Server
- tc Server
- ERS
- Hyperic HQ
- AMS
- Grails
- Archived Alerts

CVE-2010-1622

Submitted by adam.qualset@sp… on August 4, 2012

17 June 2010: CVE-2010-1622: Spring Framework execution of arbitrary code

Severity: Critical

Versions Affected:

3.0.0 to 3.0.2
2.5.0 to 2.5.6.SEC01 (community releases)
2.5.0 to 2.5.7 (subscription customers)

Earlier versions may also be affected

Description:

The Spring Framework provides a mechanism to use client provided data to update the properties of an object. This mechanism allows an attacker to modify the properties of the class loader used to load the object (via 'class.classloader'). This can lead to arbitrary command execution since, for example, an attacker can modify the URLs used by the class loader to point to locations controlled by the attacker.

Example:

This example is based on a Spring application running on Apache Tomcat.

1. Attacker creates attack.jar and makes it available via an HTTP URL. This jar has to contain following:

- META-INF/spring-form.tld - defining spring form tags and specifying that they are implemented as tag files and not classes;

- tag files in META-INF/tags/ containing tag definition (arbitrary Java code).

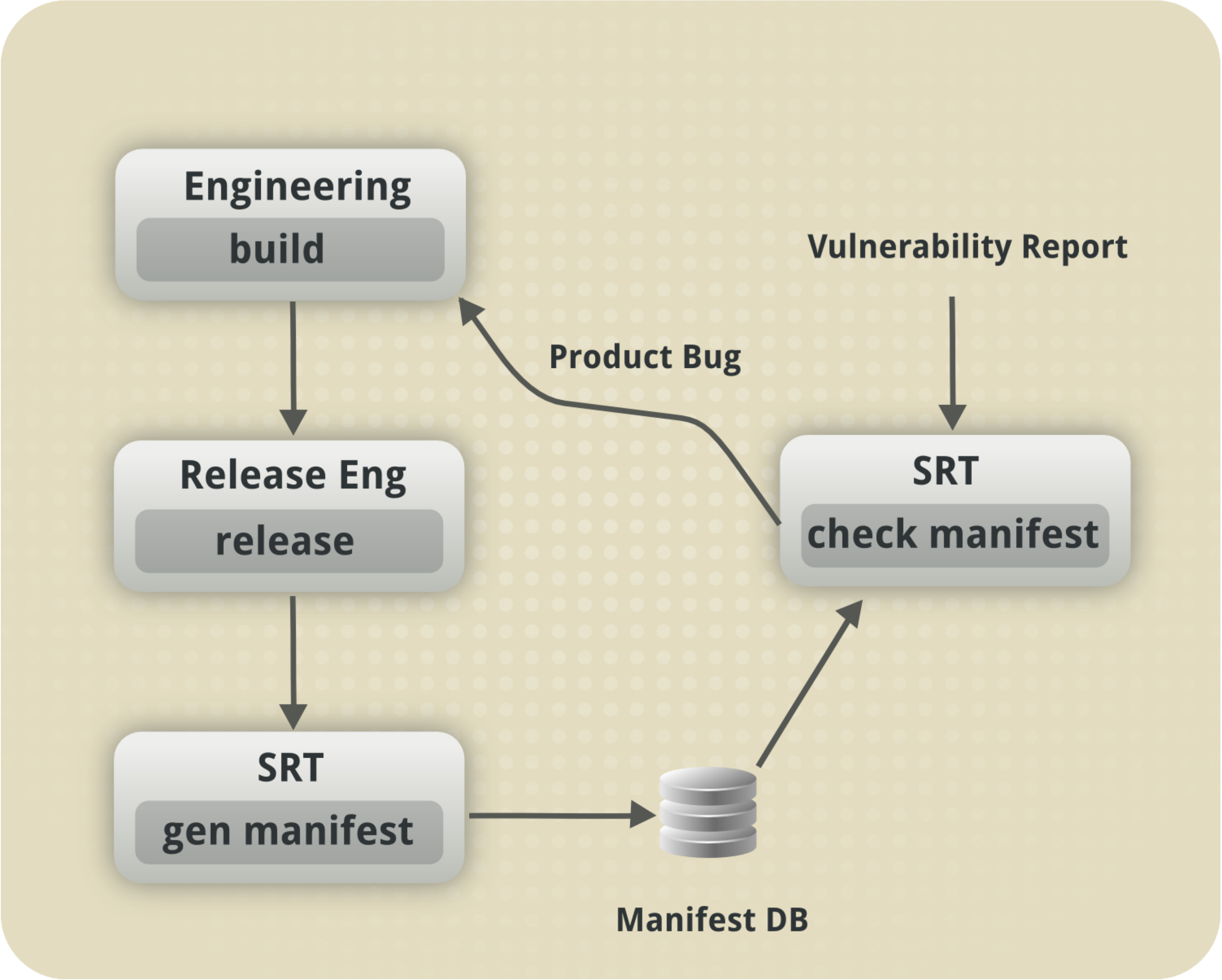
2. Attacker then submits HTTP request to a form controller with the following HTTP parameter:

```
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring</artifactId>
  <version>2.5.6</version>
</dependency>
```

Solution (Java)

- Collate all released/engineering product builds
- Recursively unpack them and generate a complete manifest database cataloging the JARs used by each build
- Match the manifest database against a database of known vulnerable JARS
- Perform a check against the database at build time

Solution (Java)



SWID

- Should this process be redesigned around SWID?
  - Currently using victims.db and a maven plugin
- The prior issues around JAR files relate to the state of containers today
  - During image creation check vulnerabilities
  - During container startup check vulnerabilities
  - On demand CVE scans
  - Same issues around appliances and Virtual Machines

Future

- Need much more documentation around content creation
  - To help with this we are creating tutorials and a portal to gather a community around
- Need to have a place for the collaboration of like minded people to work together on guidance and content
  - SSG could be the place...maybe not
  - We'd like to see automated test suites for checkin
- Drive SCAP scanning and SWID into management tools
  - Cockpit, Satellite, CloudForms, Atomic
  - Make content discovery easier

Future

- People need a cohesive story on CPE to SWID conversion
  - Security bulletins need to use it
  - Build systems need to create it
  - Need tools to make SWID useful
- Need OVAL to address JAR files
  - This trickles into container and VM techniques
- Need to address image creation in SCAP be it JAR, container, or VM
- Introspection is another area that might need some committee work
  - OVAL has active and passive attributes sometimes mixed together

Questions?
sgrubb @redhat.com