Moving Compliance to the Left

Open Source Compliance and Product Planning
Craig Northway - Director, Engineering
Mark Matyas - Sr Staff Engineer
From 5G to IoT, innovation starts with Qualcomm

$53+ billion cumulative investment in R&D

Source: Qualcomm data, as of Q3 FY18
Moving Compliance to the Left

• What do we mean by “to the left”?
  • In software development that typically means earlier in the product lifecycle

• Our historical compliance processes have been discovery-based

• What’s the problem with this? And what are we doing about it?
Background

Leaving it until the last minute
Background

Compliance coverage

https://pixabay.com/photos/matuschka-matroschka-babuschka-1029539/
Background
Planning and Guidance

Background

Security Vulnerabilities
Background

“Embedded” Dependencies

• C/C++ Embedded software
  • No culture of dependency management
  • Lack of tooling to manage dependencies
  • Memory/performance requirements
Initial Solutions

Tried to solve compliance via education and starting the review early

- **Educate Product and Program Managers on scan and review process**
  - Give us enough notice to scan and review the codebase before the product ships

- **Plan of Record → Master Scan Ticket**
  - Know about all products shipping as soon as POR entered into the system
  - Ensure we have resources and a plan for each product’s compliance needs

- **Component/Proactive/Code-Vetting scanning**
  - Scan “components” or modules at release time before they are integrated into products:
    - Maximize re-use
    - Get more direct feedback with the developers
    - Code-vetting
Problems

- Process is still retroactive
- Inefficient
  - Duplication of scanning and review efforts
- Discovery
  - Process is still a discovery-based process, less than ideal
Moving Compliance to the Left

Goals

• **Simpler, Declarative Workflow**: Review and set legal guidance earlier in the development lifecycle

• **Efficiency Gains**: Reduce manual efforts and minimize impact to engineering in collecting Third Party Software (TPS) metadata for legal review

• **System of Record**: Improved search/discovery, TPS meta data, and reporting
Focus Areas
Moving Compliance to the Left

• **Workflow**: Standardization of the Inbound Third Party Software (TPS) Review
• **Tooling**: Upfront reusable legal guidance stored in “Software Dossiers”
• **Integration**: Deeper integration with product systems/nomenclature
Inbound Third Party Software (TPS) Process

Engineer declares TPS for a product component → Legal reviews package details against usage & distro context → Code scan looks for unexpected licenses or copyrights → TPS Approval is recorded and source code is stored.

Product TPS Declaration

S/W “Dossier”

PRODUCT_FOO_2.X

TPS

❖ Boost 1.55.0
❖ Zlib 1.2.8
❖ OpenBSD
❖ Foobar 2.3
❖ Acme 1.2

TPS Catalog

Boost 1.55.0

Project Metadata

QCOM Metadata

Code Scan

Scan Boost 1.55.0

Project Vetting

❖ Concluded licenses
❖ Copyrights

Pristine Storage

Source Code
Use Cases

TPS usage can vary ...

- Full OSS Package 😄
- Partial OSS Package 😊
- Only a snippet of code is used 😐
- Binary form 🥴
- OSS Package modified by engineering 😞
- Unclear or un-versioned package, e.g. Public Domain Software 😱
- Code generated by OSS tool 😐
- OSS toolchains e.g. used by build process 🔧👨‍🔧
- Superset projects (Android, Chromium OS, Yocto) 😊
- Etc... 😫
Software Product Registry (SPR)

Tooling

- Upfront reusable legal guidance stored in “Software Dossiers”
  - Record TPS vetting and associated information, including legal guidance, distribution models, etc.
  - Can be reused across multiple products or components, e.g. a software product family
  - Existing tooling didn’t work for our particular use cases and didn’t integrate easily with our systems

- Validate declared Third Party Software (TPS) vs discovered TPS

- Reports
Software Dossiers

TPS Legal Guidance is Reusable & Composable

New Product

1.X   2.X

Product
Foo_1.X

Component
Bar_2.X

Component
Baz_4.X

Product
Foo_2.X

Component
Bar_2.X

Component
Baz_5.X
**Basic Information**

<table>
<thead>
<tr>
<th>Name</th>
<th>Product FOO_2.X</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Approved</td>
</tr>
<tr>
<td>Description</td>
<td>Software Family for Foo Product Line 2.X</td>
</tr>
<tr>
<td>Product Documents</td>
<td><a href="http://foo.com">http://foo.com</a></td>
</tr>
<tr>
<td>Product Ticket</td>
<td>OSCR-44312</td>
</tr>
<tr>
<td>Trusted Advisor(s)</td>
<td>ahamilton, aburr</td>
</tr>
<tr>
<td>Product Manager(s)</td>
<td>tjjefferson</td>
</tr>
<tr>
<td>Created</td>
<td>8/6/2019, 1:39:47 PM</td>
</tr>
<tr>
<td>Modified</td>
<td>8/6/2019, 1:43:14 PM</td>
</tr>
</tbody>
</table>

**Distribution Model(s)**

- 2 items

**Architecture**

- 1 item

**Scan Plan**

- 1 item

**Guidance**

**Legal Guidance**

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
# Module Quux_2.3.X

## Third Party Open Source

Include TPS from all Sub-Dossiers

<table>
<thead>
<tr>
<th>Name</th>
<th>Version</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Material Design Iconic Font</td>
<td>nv</td>
<td>Material design icons are the official icon set from Google that are designed under the material design...</td>
</tr>
<tr>
<td>Stylus Toolbox</td>
<td>0.2.5</td>
<td>Stylus Toolbox is an Epson inkjet printer utility. It shows ink levels, allows you to clean and align...</td>
</tr>
<tr>
<td>handlebars</td>
<td>2.0.0</td>
<td>Handlebars.js is an extension to the Mustache templating language created by Chris Wanstrath. Handlebars...</td>
</tr>
<tr>
<td>React Native</td>
<td>0.48.2</td>
<td>Build native mobile apps using JavaScript and React. React Native lets you build mobile apps using o...</td>
</tr>
<tr>
<td>jQuery</td>
<td>1.11.1</td>
<td>jQuery is a cross-platform JavaScript library designed to simplify the client-side scripting of HTML...</td>
</tr>
</tbody>
</table>
Software Product Registry (SPR)

Tooling

• Upfront reusable legal guidance stored in “Software Dossiers”

• Validate declared Third Party Software (TPS) vs discovered TPS
  • Store source code for vetted TPS in a “Pristine Storage” repository
  • Validation: compare TPS discovered in scans vs TPS declared in SPR
    • Reduce manual scanning efforts, shorten approval times
    • Focus on new/undeclared TPS

• Reports
Third Party Software (TPS) Validation

SOFTWARE REGISTRY
“S/W Dossier”
SW_FOO_2.X
  · Declared TPS
SW_FOO_2.1-8362
  · Product Build

BUILD TRIGGERS SCAN
Scan Ticket
SW_FOO_2.1-8362

CODE SCAN
SW_FOO_2.1-8362
Scan
  · Perform release scan
  · Declared TPS passed in as input
Validation
  · TPS identified in the scan compared against declared TPS (using Pristine Storage)

RESULTS
  · Validated list of Declared TPS
  · Report of unexpected TPS for engineering review
  · Missing TPS declarations added to Dossier in SPR
Software Product Registry (SPR)

Tooling

• Upfront reusable legal guidance stored in “Software Dossiers”

• Validate declared (Third Party Software) TPS vs TPS discovered in scans

• Reports
  • Portal to generate reports on demand for any release
  • E.g. Notice File, TPS Disclosure Report, etc.
  • Lower cost to generate reports and related artifacts
WIMP

Where Is My Pizza Product?

- Tighter integration into product definition systems
- Automate as much as possible
- Provide indicators to program management and drive any enforcement as early as possible
- Integrate Software Dossiers into product lifecycle and planning dashboards
  - Has the TPS been declared and approved as part of the product plan?
<table>
<thead>
<tr>
<th>Software Product</th>
<th>JIRA</th>
<th>Status</th>
<th>Milestone</th>
<th>SAST Phase</th>
<th>SAST Health</th>
<th>Enter</th>
<th>Leave</th>
<th>Task Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>QCTSAST-120576</td>
<td>Closed</td>
<td>CS</td>
<td>No Review</td>
<td>No Planned Review</td>
<td></td>
<td>6/26/2019</td>
<td>7/9/2019</td>
<td>13 days 20 hours</td>
</tr>
<tr>
<td>QCTSAST-115532</td>
<td>Closed</td>
<td>CS</td>
<td>Review Complete</td>
<td>Complete</td>
<td></td>
<td>5/6/2019</td>
<td>6/26/2019</td>
<td>50 days 19 hours</td>
</tr>
<tr>
<td>QCTSAST-100224</td>
<td>Closed</td>
<td>CS</td>
<td>Review Complete</td>
<td>Complete</td>
<td></td>
<td>5/6/2019</td>
<td>8/6/2019</td>
<td>91 days 16 hours</td>
</tr>
<tr>
<td>QCTSAST-99578</td>
<td>Closed</td>
<td>CS</td>
<td>Review Complete</td>
<td>Complete</td>
<td></td>
<td>5/6/2019</td>
<td>8/6/2019</td>
<td>91 days 16 hours</td>
</tr>
<tr>
<td>QCTSAST-92573</td>
<td>Closed</td>
<td>NA</td>
<td>Review Complete</td>
<td>Complete</td>
<td></td>
<td>5/6/2019</td>
<td>5/6/2019</td>
<td>0 days 0 hours</td>
</tr>
</tbody>
</table>

- **3/27/2019**: Software Product Created
- **7/9/2019**: 7 days 21 hours
- **5/6/2019**: Scanning master ticket(s) created. Hover over node to see details.
- **4/29/2019**: Distro created in
- **3/5/2019**: Product Kit Created in
- **6/13/2019**: 7 days 17 hours
- **8/6/2019**: 49 days 14 hours
- **8/2/2019**: Pending

**Task Duration:**
- 13 days 20 hours
- 50 days 19 hours
- 91 days 16 hours
- 0 days 0 hours
Built with OSS!

- **SPR**
  - TypeScript, React,
  - Python, Django, DRF, Postgres

- **Codescan**
  - Python, Flask, Postgres, Elasticsearch
  - FOSSology, LiD (License Identifier)

- **Notice tool**
  - Python, Django

- **TPS Workflow**
  - Jenkins
  - Git
Future + Ecosystem

• Why isn’t SPR open source?
  • It’s currently too tightly coupled with our process, integrated into our product tools and specific to Qualcomm’s Open Source legal guidance

• Alternatives
  • Amazon’s oss-contribution-tracker + oss-attribution-builder
  • nexB’s DejaCode + ScanCode Toolkit
  • OSS Review Toolkit
  • Eclipse SW360

• ClearlyDefined
  • Harvest data from OSS projects, allows community to curate missing information