Core Infrastructure Initiative badging program for ONAP

LFN DDF & Plugfest in Stockholm

Pawel Pawlak, Amy Zwarico, Tony Hansen, Pierre Close

12th June 2019
• Core Infrastructure Initiative was created by the Linux Foundation Networking in response to previous security issues in open-source projects (Heartbleed in OpenSSL).

• The CII has created a badging program to recognize projects that follow a set of identified best practices that could be adopted.
  - There are three levels: Passing, Silver and Gold.

• The security sub-committee is reviewing multiple answers in several areas: Basics, Change control, Reporting, Quality, Security and Analysis, starting from the Beijing release
  - Source: https://wiki.onap.org/display/DW/CII+Badging+Program

• Tony has prepared a joint table that helps to identify areas where an effort should be made to improve scoring for projects:
  - Source: http://tlhansen.us/onap/cii.html
CII badging program

- Defines a set of best practices for Free/Libre and Open Source Software. It defines 3 levels: passing, silver and gold.
  - Encourage projects to follow the best practices
  - Help new projects discover what those practices are
  - Help users know which projects are following the best practices.
  - Passing criteria covers general project aspects with higher level of security practices described in silver and gold levels.

- Individual projects apply for the badging.
  - Include "ONAP" in the "human-readable name of the project" so our tools find it

- Basic introduction can be found here: https://github.com/coreinfrastructure/best-practices-badge/blob/master/doc/criteria.md

- Silver and gold criteria can be found here: https://github.com/coreinfrastructure/best-practices-badge/blob/master/doc/other.md
CII Badging Scope and Sample Requirement Areas

• **Basics**
  - Project description, OSS licensing, documentation, website, support TLS, change control, unique version numbering, release notes

• **Reporting**
  - Bug-reporting process, vulnerability report process

• **Quality**
  - Maintain golden source for rebuilding, use common tools, automate test suite, perform new-functionality testing, address compiler warning flags

• **Security**
  - Developers security knowledgeable, use good cryptographic practices, protection against man-in-the-middle (MITM) attacks, fix publicly known vulnerabilities, don’t leak valid private credential

• **Analysis**
  - Perform static code analysis, perform dynamic code analysis, fix vulnerabilities
Example criteria change between levels

• **Passing (sample):**
  - The default security mechanisms within the software produced by the project **SHOULD NOT** depend on cryptographic algorithms or modes with known serious weaknesses (e.g., the SHA-1 cryptographic hash algorithm or the CBC mode in SSH).
  - The project software **SHOULD** use hardening mechanisms so software defects are less likely to result in security vulnerabilities. If the project does not produce software, choose N/A

• **Silver (=Passing+1):** You must achieve the lower (passing) badge. In addition, some **SHOULD** will become **MUST**, and some **SUGGESTED** will become **SHOULD** or **MUST**.
  - Upgrade crypto_weaknesses from **SHOULD** to **MUST**. The default security mechanisms within the software produced by the project **MUST NOT** depend on cryptographic algorithms or modes with known serious weaknesses (e.g., the SHA-1 cryptographic hash algorithm or the CBC mode in SSH).

• **Gold (=Passing+2=Silver+1):** Upgrade of **SHOULD** and **SUGGESTED** (or not)
  - Upgrade hardening from **SHOULD** to **MUST**. "The project software **MUST** use hardening mechanisms so software defects are less likely to result in security vulnerabilities. If the project does not produce software, choose N/A."
  - Upgrade crypto_used_network from **SHOULD (NOT)** to **MUST (NOT)**. "The project **MUST NOT** use unencrypted network communication protocols (such as HTTP and telnet) if there is an encrypted equivalent (e.g., HTTPS/TLS and SSH), unless the user specifically requests or configures it. (N/A allowed).
  - Upgrade crypto_tls12 from **SHOULD** to **MUST**. The project **MUST**, if it supports TLS, support at least TLS version 1.2. Note that the predecessor of TLS was called SSL. (N/A allowed).
CII badging considerations

• Please remember to update your CII Badging answers across ONAP release lifecycle – discussion point M0 milestone?

• It is a good idea to have security delegate/champion in the projects.

• As we are working in a dynamic environment and committers are leaving projects, we have a process to add multiple editors for projects in CII Badging.
  - Make certain you have more than one editor on the CII page
    • [https://wiki.onap.org/display/DW/CII+Badging+Program#CIIBadgingProgram-Howtoaddmultipleeditorstoaprojectreport](https://wiki.onap.org/display/DW/CII+Badging+Program#CIIBadgingProgram-Howtoaddmultipleeditorstoaprojectreport)
  - e.g., your PTL and security SME should both have access to the page
  - Also please add Jim Baker (LFID: mtnskijier, bestpractices.coreinfrastructure.org id:3607) from the Linux Foundation – only 4 ONAP projects still pending…

• There are some general ONAP wide items (like Code of Conduct) that should be soon approved by ONAP TSC ([https://jira.onap.org/browse/TSC-130](https://jira.onap.org/browse/TSC-130)) – so ONAP projects would earn some extra points for passing.

• If some questions are not easy to understand, please use expand field and don’t hesitate to ask questions to SECCOM – we are here to help projects!

• JS to be covered in the scans.
El Alto
Regular PTL Reviews of CII Issues

• About ½ of the ONAP projects have started answering the Silver or Gold questions

• At PTL meetings
  - Discussion of an issue that we want a PTL to look at and answer
    • Something that the PTL should be able to answer in about 5 minutes
    • Explicit instructions would be provided on how to answer that question
  - After a week, kudos to the PTLs who answered
  - JIRA tickets then written against projects that had NOT been answered
  - One week later, escalate to higher importance

• PTLs should write project JIRA tickets against issues where they cannot answer with MET, assigned to either El Alto or Frankfurt
ONAP CII Badging Status and Clarifications
## ONAP CII Badging Status ½ as of 27th of April’19

<table>
<thead>
<tr>
<th>Project Prefix</th>
<th>Full Name</th>
<th>Passing</th>
<th>Requirements Not Met</th>
<th>Requirements Unanswered</th>
<th>Silver</th>
<th>Gold</th>
</tr>
</thead>
<tbody>
<tr>
<td>clamp</td>
<td>CLAMP (Closed Loop Automation</td>
<td>98</td>
<td>Vulnerabilities Fixed 60 days</td>
<td></td>
<td>93</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>Management Platform</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>appc</td>
<td>ONAP APMC (Application Controller)</td>
<td>98</td>
<td>Vulnerabilities Fixed 60 days</td>
<td></td>
<td>93</td>
<td>26</td>
</tr>
<tr>
<td>ccsdk</td>
<td>ONAP CSDK (Common Controller SDK)</td>
<td>98</td>
<td>Vulnerabilities Fixed 60 days</td>
<td></td>
<td>13</td>
<td>0</td>
</tr>
<tr>
<td>sdnc</td>
<td>ONAP SDNC (SDN Controller)</td>
<td>98</td>
<td>Vulnerabilities Fixed 60 days</td>
<td></td>
<td>13</td>
<td>9</td>
</tr>
<tr>
<td>vid</td>
<td>ONAP Virtual Infrastructure Deployment (VID)</td>
<td>98</td>
<td>Vulnerabilities Fixed 60 days</td>
<td></td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>logging-analytics</td>
<td>logging-analytics</td>
<td>98</td>
<td>Release Notes Vulns</td>
<td></td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>portal</td>
<td>Portal Platform</td>
<td>97</td>
<td>Static Analysis Fixed</td>
<td>Dynamic Analysis Fixed 60 days</td>
<td>87</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Dynamic Analysis Enable Assertions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>vvp</td>
<td>vvp</td>
<td>97</td>
<td>Static Analysis Fixed</td>
<td>Dynamic Analysis Fixed 60 days</td>
<td>11</td>
<td>13</td>
</tr>
<tr>
<td>usecase-ui</td>
<td>usecase-ui</td>
<td>97</td>
<td>Static Analysis Fixed</td>
<td></td>
<td>11</td>
<td>13</td>
</tr>
<tr>
<td>vfc</td>
<td>VF-C (Virtual Function Controller Project)</td>
<td>97</td>
<td>Static Analysis Fixed</td>
<td></td>
<td>11</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td>ONAP AAI UI (sparky-fe)</td>
<td>97</td>
<td>Static Analysis Fixed</td>
<td></td>
<td>84</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>ONAP ESR (External System Register)</td>
<td>97</td>
<td>Static Analysis Fixed</td>
<td></td>
<td>11</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>ONAP AAI UI</td>
<td>98</td>
<td>Static Analysis Fixed</td>
<td></td>
<td>87</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>ONAP Champ</td>
<td>95</td>
<td>Dynamic Analysis Fixed</td>
<td>Static Analysis Fixed</td>
<td>78</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>ONAP Gizmo</td>
<td>97</td>
<td>Static Analysis Fixed</td>
<td>Static Analysis Fixed</td>
<td>78</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>ONAP Spike</td>
<td>98</td>
<td>Static Analysis Fixed</td>
<td></td>
<td>78</td>
<td>17</td>
</tr>
<tr>
<td>externalapi</td>
<td>ONAP External API</td>
<td>95</td>
<td>Release Notes Vulns</td>
<td>Test</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Tests are Added</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sdc</td>
<td>ONAP SDC (Service Design and Creation)</td>
<td>92</td>
<td>Crypto Keylength</td>
<td>Warnings Strict</td>
<td>35</td>
<td>0</td>
</tr>
</tbody>
</table>
ONAP CII Badging Status 2/2

- As we do value honesty and transparency, so if we know that some project should refine its answers, it should be done even if it could negatively impact passing criteria – discussion point: when exactly this could be done?

- Losing CII badging level (and some scoring points) is a natural and an “acceptable” way forward, as long as there is a limit in the level score loss to prevent projects losing focus on the CII level which is, among other things, one of the ways to ensure projects follow security best practices.

- Score loss should be well documented.

- Getting higher, better score is a great proof of a project’s excellence (learning organization) and we do prefer honesty over high scoring…
ONAP CII Badging Clarifications 1/4

• [vulnerabilities_fixed_60_days] False positives only – answer is yes
  - There MUST be no unpatched vulnerabilities of medium or high severity that have been publicly known for more than 60 days.
  - Answers
    • Met: if (1) project can commit to fixing new vulnerabilities within 60 days or (2) no vulnerabilities or (3) if all vulnerabilities are false positives
    • Unmet: otherwise

• [release_notes]
  - The project MUST provide, in each release, release notes that are a human-readable summary of major changes in that release to help users determine if they should upgrade and what the upgrade impact will be. The release notes MUST NOT be the raw output of a version control log (e.g., the "git log" command results are not release notes). Projects whose results are not intended for reuse in multiple locations (such as the software for a single website or service) AND employ continuous delivery MAY select "N/A". (N/A allowed.) (Justification required for "N/A".) (URL required for "met".)
  - Answer: Met for all projects
[static_analysis]
- At least one static code analysis tool (beyond compiler warnings and "safe" language modes) MUST be applied to any proposed major production release of the software before its release, if there is at least one FLOSS tool that implements this criterion in the selected language. (N/A allowed.) (Justification required for "N/A").

- Answers
  - **Met**: Java, Python
  - **N/A**: javascript, Go, and other languages

[static_analysis_fixed]
- All medium and high severity exploitable vulnerabilities discovered with static code analysis MUST be fixed in a timely way after they are confirmed. (N/A allowed.)

- Answer:
  - **Met**: Java, Python
  - **N/A**: all other languages
The project MUST use at least one automated test suite that is publicly released as FLOSS (this test suite may be maintained as a separate FLOSS project).

- Answers
  - **Met**: Java
  - **N/A**: all other languages

It is SUGGESTED that at least one dynamic analysis tool be applied to any proposed major production release of the software before its release.

- Answer: **N/A** for all projects

- **[dynamic_analysis_fixed]** N/A
  - All medium and high severity exploitable vulnerabilities discovered with dynamic code analysis MUST be fixed in a timely way after they are confirmed. (N/A allowed.)
  - Answer: **N/A** for all projects
• [vulnerability_report_private]
  - If private vulnerability reports are supported, the project MUST include how to send the information in a way that is kept private (URL required).
  - We have a new process and private jira in place (OJSIs) -
  - Answer: MET All reported vulnerabilities are kept private in a private JIRA (OJSI)
How To

• How to display Silver and Gold Questions

• Where to find stock answers for questions

• Example session filling in some of those stock answers