# CII Badging Program

blocked URL



## What is the CII Badging program?

CII (Core Infrastructure Initiative) Badge may be achieved by the projects which follow the Best practices criteria for Free/Libre and Open Source Software (FLOSS).

CII has been created by the linux foundation in response to previous security issues in open-source projects (e.g. Heartbleed in openSSL).

The CII Badging is associated to the areas as follows:

> Basics, Change Control, Reporting, Quality, Security & Analysis

Projects in ONAP should be CII certified to an appropriate level in order to confirm with expectation of carrier grade.

### How to add multiple editors to a project report

- You need the numeric ID for the new editor you're adding to your project report.
  - If the new editor doesn't already have a login on the CII site, they need to create one by going to https://bestpractices.coreinfrastructure.org and clicking on Sign Up.
  - If you don't know the new editor's CII login ID, have them log in and click on Account -> Profile. At the bottom is a link called “JSON”. Click that and a JSON structure will be shown. We need the “id” value, which is numeric. For example, mine is 1597.
- An existing owner or editor of a project then needs to bring up the project page that they want to add another editor to. At the top is a menu item “Edit”. Click that, and search for “(Advanced) What other users have additional rights to edit this badge entry?”. In the field right below that, type in “+” and the numeric ID retrieved above. Click on one of the green “Save” buttons below.
- That new person is immediately added as an editor on the project.
- As an editor, you can quickly get a list of the projects that you have rights to by clicking on “Account -> Profile”.

## Do I Report For the Entire Project or Separately For Every Single Repo? Can I Group Repos Together?

This is your choice. You can do a single report for your entire project, or you can file a separate report for each repository, or even group multiple repositories together.

- If your repositories use different languages or different testing procedures, you probably would find it easier to do it per-repository.
- You may also group multiple repositories together, if that would help. For example, you might group all of the repositories together that use Java, or all of the repos together that use Erlang.
- The key to grouping repositories together is to list all of the repo URLs in the response to the question "What is the URL for the version control repository?" question.

It's your choice as to what makes it easiest for you to manage your project.

If you do file one report for the entire project, or group multiple repos together then you need to answer each Met/Not Met question based on the lowest-common denominator answer: if ANY of the repos don’t meet the requirement, you cannot select Met. You can use the text response for each question to keep track of your reasons for picking any particular answer.

## CII Badging Levels

There are 3 CII Badging levels which are as follows:

- Passing
- Silver
- Gold

When a new project starts the badging process they will begin at 0% completeness and as they progress the % will increase.

To see a list of all ONAP projects and their level of completions refer to link https://bestpractices.coreinfrastructure.org/projects?q=onap

# ONAP CII Compliance Levels

For ONAP, 4 levels(ONAP Compliance) of compliance have been defined:

For each ONAP compliance level, all the projects in ONAP should comply to certain standards when it comes to CII badging.

**ONAP Level 1**: 70 % of the code in gerrit must have an 100% completion in the CII badging towards passing level
with the non-passing projects reaching 80% completion in the CII badging towards passing level
Non-passing projects MUST pass specific cryptography criteria outlined by the Security Subcommittee*

**ONAP Level 2:** 70 % of the projects in gerrit must have an 100% completion in the CII badging for silver level
with non-silver projects completed passing level and 80% completion towards silver level

**ONAP Level 3:** 70% of the projects in gerrit must have an 100% completion in the CII badging for gold level
with non-gold projects achieving silver level and achieving 80% completion towards gold level

**ONAP Level 4:** 100 % passing gold.

**ONAP CII Badging Dashboard:** http://tlhansen.us/onap/cii.html

Some of the important high level *example* criteria associated to the various levels are listed as follows for quick reference:

| Level | Example Details/Criteria |
|---|---|
| Passing | The project website MUST succinctly describe what the software does (what problem does it solve?).<br>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). |
| Silver | The project MUST document what the user can and cannot expect in terms of security from the software produced by the project. The project MUST identify the security requirements that the software is intended to meet and an assurance case that justifies why these requirements are met.<br><br>The assurance case MUST include: a description of the threat model, clear identification of trust boundaries, and evidence that common security weaknesses have been countered |
| Gold | The project MUST have at least 50% of all proposed modifications reviewed before release by a person other than the author, to determine if it is a worthwhile modification and free of known issues which would argue against its inclusion. |

## Badge Specific Adherence requirements

Each of the Badging level is associated with compliance requirements which in turn may vary from being e.g. absolute to being as varied as recommendatory in nature.

The key words "MUST", "MUST NOT", "SHOULD", "SHOULD NOT", and "MAY" in the Badging guideline documents are are to be interpreted as below. (The terms are similar to what is described in RFC2119/RFC8174).

- The term MUST is an absolute requirement, and MUST NOT is an absolute prohibition.
- The term SHOULD indicates a criterion that is normally required, but there may exist valid reasons in particular circumstances to ignore it. However, the full implications must be understood and carefully weighed before choosing a different course.
- The term SUGGESTED is used instead of SHOULD when the criterion must be considered, but valid reasons to not do so are even more common than for SHOULD.
- Often a criterion is stated as something that SHOULD be done, or is SUGGESTED, because it may be difficult to implement or the costs to do so may be high.
- The term MAY provides one way something can be done, e.g., to make it clear that the described implementation is acceptable.

- To obtain a badge, all MUST and MUST NOT criteria must be met, all SHOULD criteria must be met OR the rationale for not implementing the criterion must be documented, and all SUGGESTED criteria have to be considered (rated as met or unmet). In some cases a URL may be required as part of the criterion's justification.

# R2 Beijing Requirements

For the Beijing release, the compliance requirement is ONAP Level 1 (at least 70% of the project are on passing level, and all non-passing projects at >80% towards passing).

## R2 Beijing Current Status Dashboard

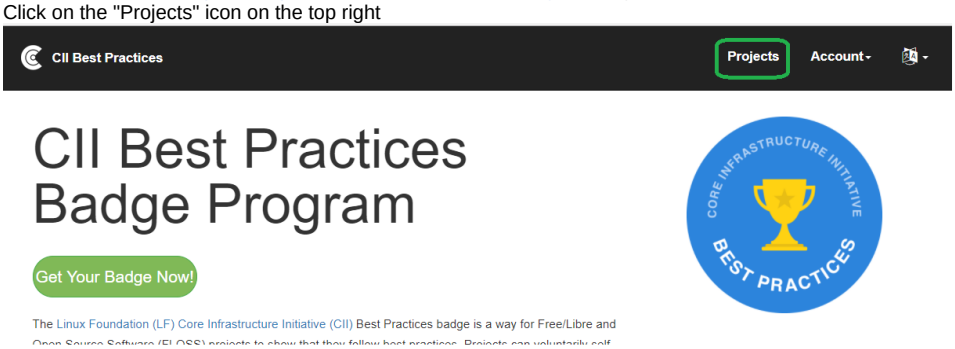
The dashboard gives a list of all onap projects that are undergoing the process and their % of completion.

<TODO insert a link to the dashboard table here>

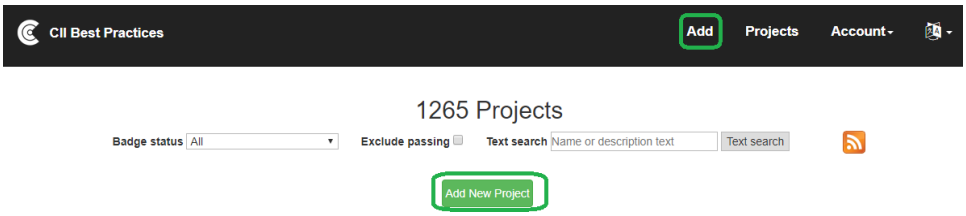
# Procedure to Fill Out the BestPractices.CoreInfrastructure.Org Form

First step is create a new project in bestpractices website

1. Create a account in https://bestpractices.coreinfrastructure.org/ and login
2. Click on the "Projects" icon on the top right




3. This page will list all the projects certified by CII not just the onap projects. Click on Add/Add new project button to add a new project.




4. Enter the details of your project in the new screen and click "Submit URL"

Now you will be prompted with a set of questions and most of them are straightforward. The following set of Sample Questions and Answers should help you fill it out. You may also wish to refer to one of the existing projects to get an idea of what has to be filled in. You can use this link and click on any project name to see the answers used for that project.

You should go through the questions for each of the levels. Some of the questions at Silver level change a "SHOULD" into a "MUST" from the Passing level, so if you have met some suggestion at Passing level, verify that is marked as Met at Silver level as well.

## Sample Passing Level Questions and Answers

This section will cover all the questions in each level and what it means and what a possible answer should be. A description of the question will be provided where needed.

| | Question | Description | Sample Answer |
|---|---|---|---|
| | **Basics: Identification** | | |
| | What is the human-readable name of the project? | This is a short name. It SHOULD include "ONAP" as part of it. | ONAP CLAMP (Closed Loop Automation Management Platform) |
| | What is a brief description of the project? | | ONAP CLAMP is a platform for designing and managing control loops. It is used to |

| | | | |
|---|---|---|---|
| | | Include a paragraph describing your project. You MUST include "ONAP" as part of it the description in order for our queries to work. | design a closed loop, configure it with specific parameters for a particular ... etc ... |
| | What is the URL for the project (as a whole)? | Use the wiki URL for your project. Only use the HTTPS version. | https://wiki.onap.org/display/DW/CLAMP+Project |
| | What is the URL for the version control repository (it may be the same as the project URL)? | This will be the Gerrit URL for your project. It MUST start with either <https://gerrit.onap.org/r/#/admin/projects> or <https://git.onap.org>.<br><br>You may list multiple URLs here if your report is covering multiple repositories. Separate them with whitespace. | https://gerrit.onap.org/r/#/admin/projects/clamp<br>https://git.onap.org/clamp |
| | What programming language(s) are used to implement the project? | | C++, Java, JavaScript, Python, etc. |
| | What is the Common Platform Enumeration (CPE) name for the project (if it has one)? | The textual answer is optional; you may leave it blank. | |
| | **Question** | **Description** | **Sample Answer** |
| | **Basics: Basic project website content** | | |
| | The project website MUST succinctly describe what the software does [description_good] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button.<br><br>Include the link to your readme file on onap.readthedocs.io | The description of the project can be found in http://onap.readthedocs.io/en/latest/submodules/aai/sparky-be.git/docs/index.html |
| | The project website MUST provide information on how to: obtain, provide feedback (as bug reports or enhancements), and contribute to the software [interact] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use the text in the sample answer. | The following URLs describe the process to join the community, developing the software and provide feekback: https://wiki.onap.org/display/DW/Joining+the+Community https://wiki.onap.org/display/DW/Tracking+Issues+with+JIRA https://wiki.onap.org/display/DW/Developing+ONAP |
| | The information on how to contribute MUST explain the contribution process (e.g., are pull requests used?) (URL required) [contribution] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use the text in the sample answer. | The process could be found in the following URL: https://wiki.onap.org/display/DW/Development+Procedures+and+Policies |
| | The information on how to contribute SHOULD include the requirements for acceptable contributions (e.g., a reference to any required coding standard). [contribution_requirements] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use the text in the sample answer. | The Javascript code should meet the requirements except for the number of characters in a line of code specified by the styleguide https://google.github.io/styleguide/jsguide.html<br>We avoid the restriction on the number of characters in one line of code to improve readability. |
| | **Question** | **Description** | **Sample Answer** |
| | **Basics: FLOSS license** | | |
| | What license(s) is the project released under? [license] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use the text in the sample answer. | Apache-2.0 |
| | The software produced by the project MUST be released as FLOSS. [floss_license] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use the text in the sample answer. | The Apache-2.0 license is approved by the Open Source Initiative (OSI). |
| | It is SUGGESTED that any required license(s) for the software produced by the project be approved by the Open Source Initiative (OSI) [floss_license_osi] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use the text in the sample answer. | The Apache-2.0 license is approved by the Open Source Initiative (OSI). |
| | The project MUST post the license(s) of its results in a standard location in their source repository. (URL required) [license_location] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use the text in the sample answer. | License can be found in: https://gerrit.onap.org/r/gitweb?p=aai/sparky-fe.git;a=blob;f=LICENSE;h=38a0459285f876f7cb07c931fe01d195b9122872;hb=refs/heads/amsterdam |
| | **Question** | **Description** | **Sample Answer** |
| | **Basics: Documentation** | | |
| | The project MUST provide basic documentation for the software produced by the project. [documentation_basics] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button.<br><br>Include a pointer to your project's readthedocs.io area. | The documentation describing the project can be found in http://onap.readthedocs.io/en/latest/submodules/aai/sparky-be.git/docs/index.html |
| | The project MUST provide reference documentation that describes the external interface (both input and output) of the software produced by the project. [documentation_interface] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | Full Documentation for ONAP can be found at: https://docs.onap.org/en/latest/ |
| | | | |

| | Question | Description | Sample Answer |
|---|---|---|---|
| | **Basics: Other** | | |
| | The project sites (website, repository, and download URLs) MUST support HTTPS using TLS. [sites_https] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | The project sites are all HTTPS:<br>Project site: https://wiki.onap.org/pages/viewpage.action?pageId=13599492<br>Repository: https://gerrit.onap.org/r/#/admin/projects/aai/sparky-fe |
| | The project MUST have one or more mechanisms for discussion (including proposed changes and issues) that are searchable, allow messages and topics to be addressed by URL, enable new people to participate in some of the discussions, and do not require client-side installation of proprietary software. [discussion] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use the text in the sample answer. | A mailing list is used for project related discussion. New users could also check, search the old discussion online at onap-discuss website. Joining the ONAP Technical Community |
| | The project SHOULD provide documentation in English and be able to accept bug reports and comments about code in English. [english] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use the text in the sample answer. | JIRA is used to track bugs. The whole website is in English. Tracking Issues with JIRA |
| | **Question** | **Description** | **Sample Answer** |
| | **Change Control: Public version-controlled source repository** | | |
| | The project MUST have a version-controlled source repository that is publicly readable and has a URL. [repo_public] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | Sparky's version controlled repository can be found in https://gerrit.onap.org/r/#/admin/projects/aai/sparky-fe |
| | The project's source repository MUST track what changes were made, who made the changes, and when the changes were made. [repo_track] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | Tracking is provided by using a combination of JIRA and git history. Every commit has an user and a Jira number attached to it.<br>Git history for sparky's master branch:https://gerrit.onap.org/r/gitweb?p=aai%2Fsparky-fe.git;a=shortlog;h=refs%2Fheads%2Fmaster<br>Jira for ONAP: https://jira.onap.org/secure/Dashboard.jspa |
| | To enable collaborative review, the project's source repository MUST include interim versions for review between releases; it MUST NOT include only final releases. [repo_interim] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | Gerrit provides an temporary branch for reviewing and providing comments. Once approved, the code will be merged and the temperate branch will be removed. |
| | It is SUGGESTED that common distributed version control software be used (e.g., git) for the project's source repository. [repo_distributed] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | Git and Gerrit are used. |
| | **Question** | **Description** | **Sample Answer** |
| | **Change Control: Unique version numbering** | | |
| | The project results MUST have a unique version identifier for each release intended to be used by users [version_unique] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | Release version is with format ${major}.${minor}.${patch} and will be updated accordingly for each release. |
| | It is SUGGESTED that the Semantic Versioning (SemVer) format be used for releases [version_semver] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | Release version is with format ${major}.${minor}.${patch} and will be updated accordingly for each release. |
| | It is SUGGESTED that projects identify each release within their version control system. For example, it is SUGGESTED that those using git identify each release using git tags. [version_tags] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | Each release is tagged within the Gerrit repository. |
| | **Question** | **Description** | **Sample Answer** |
| | **Change Control: 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". (URL required) [release_notes] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | Release notes can be found in http://onap.readthedocs.io/en/latest/submodules/aai/aai-common.git/docs/release-notes.html |
| | The release notes MUST identify every publicly known vulnerability with a CVE assignment or similar that is fixed in each new release, unless users typically cannot practically update the software themselves. If there are no release notes or there have been no publicly known vulnerabilities, choose "not applicable" (N/A). [release_notes_vulns] | If your project has had a vulnerability reported (e.g. identified in Nexus-IQ), verify that it is noted in the release notes with a CVE, CVSS, CWE, or CAPEC identifier, then select the Met radio button. (If not, select the Unmet radio button.)<br><br>If there have been no vulnerabilities yet reported, select N/A. | Release notes with identified vulnerabilities can be found in http://onap.readthedocs.io/en/latest/submodules/aai/aai-common.git/docs/release-notes.html<br><br>No vulnerabilities have yet been identified. |
| | **Question** | **Description** | **Sample Answer** |
| | **Reporting: Bug-reporting process** | | |

| | | | |
|---|---|---|---|
| | The project MUST provide a process for users to submit bug reports (e.g., using an issue tracker or a mailing list). (URL required) [report_process] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | The description of the process can be found in the following URL: https://wiki.onap.org/display/DW/Tracking+Issues+with+JIRA |
| | The project SHOULD use an issue tracker for tracking individual issues. [report_tracker] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | Jira is used to track issues. https://wiki.onap.org/display/DW/Tracking+Issues+with+JIRA |
| | The project MUST acknowledge a majority of bug reports submitted in the last 2-12 months (inclusive); the response need not include a fix. [report_responses] | *ONAP project common response*<br><br>ONAP requires that the PTLs review bug reports, so you can just select the Met radio button. Use text similar to the sample answer. | The reported issues are being handled as soon as possible. |
| | The project SHOULD respond to a majority (>50%) of enhancement requests in the last 2-12 months (inclusive).[enhancement_responses] | *ONAP project common response*<br><br>ONAP requires that the PTLs review enhancement requests, so you can just select the Met radio button. Use text similar to the sample answer. | The reported issues are being handled as soon as possible. |
| | The project MUST have a publicly available archive for reports and responses for later searching. (URL required) [report_archive] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | The ONAP Jira Ticketing System: https://jira.onap.org/secure/Dashboard.jspa<br>Additionally a concise list of open defects, epics, stories, and tasks can be found at: https://wiki.onap.org/display/DW/Issue+Reports |
| | **Question** | **Description** | **Sample Answer** |
| | **Change Control: Vulnerability report process** | | |
| | The project MUST publish the process for reporting vulnerabilities on the project site. (URL required) [vulnerability_report_process] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | The process on how to report a vulnerability can be found in ONAP Vulnerability Management <https://wiki.onap.org/display/DW/ONAP+Vulnerability+Management>. |
| | If private vulnerability reports are supported, the project MUST include how to send the information in a way that is kept private. (URL required) [vulnerability_report_private] | *ONAP project common response*<br><br>According to updated version of ONAP Vulnerability Management process vulnerabilities should be reported by creating OJSI tickets. Taken into account their default limited visibility and HTTPS only access to Jira, we consider them to be private reports. All details how to report the issue has been described in the process itself. Choose "Met". | The process on how to report a vulnerability can be found in ONAP Vulnerability Management <https://wiki.onap.org/display/DW/ONAP+Vulnerability+Management>.<br><br>By default all vulnerability reports are private unless modified by the reporter. |
| | The project's initial response time for any vulnerability report received in the last 6 months MUST be less than or equal to 14 days. [vulnerability_report_response] | For most new projects there are no vulnerability reported, so N/A would be a valid selection if that is the case for your project. For older projects that were in a previous ONAP release, the JIRA tickets should be reviewed for vulnerability response times. | There's no vulnerabilities reported so far. |
| | **Question** | **Description** | **Sample Answer** |
| | **Quality: Working build system** | | |
| | If the software produced by the project requires building for use, the project MUST provide a working build system that can automatically rebuild the software from source code. [build] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | Jenkins is used to build the war file. https://jenkins.onap.org/view/aai/job/aai-sparky-fe-master-release-version-java-daily-no-sonar/ |
| | It is SUGGESTED that common tools be used for building the software. [build_common_tools] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | Maven and npm are used to build the project |
| | The project SHOULD be buildable using only FLOSS tools [build_floss_tools] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | Maven is under Apache 2.0 liscense. And NPM is licensed under The Artistic License 2.0 |
| | **Question** | **Description** | **Sample Answer** |
| | **Quality: Automated test suite** | | |
| | 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). [test] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | Sparky uses Karma, mock-require and mocha to run the unit tests |
| | A test suite SHOULD be invocable in a standard way for that language. [test_invocation] | *ONAP project common response*<br><br>ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | Tests can be run, by running the command "npm test" |
| | It is SUGGESTED that the test suite cover most (or ideally all) the code branches, input fields, and functionality. [test_most] | ONAP has differing requirements for each release; depending on your coverage, select the appropriate radio button. Use text similar to the sample answer. | The combination of Karma, mock-require and mocha has the ability to cover all the branches and input fields |
| | It is SUGGESTED that the project implement continuous integration (where new or | *ONAP project common response* | For each pull request, the project needs |

| | | | |
|---|---|---|---|
| | changed code is frequently integrated into a central code repository and automated tests are run on the result). [test_continuous_integration] | ONAP requires this, so you can just select the Met radio button. Use text similar to the sample answer. | to be built successfully before the Merge option becomes activated. The test will be run automatically during the building process as well. Once build successfully and all tests has past, the Merge option will be activated. |
| | **Question** | **Description** | **Sample Answer** |
| | **Quality: New functionality testing**<br>**Quality: Warning flags** | **The remaining questions in the Quality section must be individually answered according to your project.** | |
| | . . . | | |
| | **Question** | **Description** | **Sample Answer** |
| | **Security: Secured delivery against man-in-the-middle (MITM) attacks** | | |
| | The project MUST use a delivery mechanism that counters MITM attacks. Using https or ssh+scp is acceptable. [delivery_mitm] | *ONAP project common response*<br><br>ONAP uses HTTPS for access to all ONAP artifacts, such as built items and source. In addition, some artifacts are signed by the Linux Foundation. | HTTPS is used to download all ONAP artifacts, and some are signed by the Linux Foundation. |
| | A cryptographic hash (e.g., a sha1sum) MUST NOT be retrieved over http and used without checking for a cryptographic signature. [delivery_unsigned] | *ONAP project common response*<br><br>Since all HTTP access is over HTTPS, if ONAP were to to publish checksums, they could only be retrieved using HTTPS. | TBD<br><br>Does ONAP publish any checksums?<br><br>If so, the answer is Met. If not, the answer is N/A. |
| | **Security: Secure development knowledge**<br>**Security: Use basic good cryptographic practices**<br>**Security: Publicly known vulnerabilities fixed**<br>**Security: Other security issues** | These questions in the Security section must be individually answered according to your project. | |
| | **Question** | **Description** | **Sample Answer** |
| | **Analysis: Static code analysis**<br>**Analysis: Dynamic code analysis** | The questions in the Analysis section must be individually answered according to your project. | |

---

## Sample Silver Level Questions and Answers

| | Question | Description | Sample Answer |
|---|---|---|---|
| | **Basics: Identification**<br>**Basics: Prerequisites** | The questions in these Basics sections will be filled in automatically. | |
| | **Question** | **Description** | **Sample Answer** |
| | **Basics: Basic project website content** | | |
| | The information on how to contribute MUST include the requirements for acceptable contributions (e.g., a reference to any required coding standard). (URL required) [contribution_requirements] | *ONAP project common response*<br><br>ONAP has Developer Best Practices, so click on Met and add a reference. | https://wiki.onap.org/display/DW/Developer+Best+Practices |
| | **Question** | **Description** | **Sample Answer** |
| | **Basics: Project oversight** | | |
| | The project SHOULD have a legal mechanism where all developers of non-trivial amounts of project software assert that they are legally authorized to make these contributions. The most common and easily-implemented approach for doing this is by using a Developer Certificate of Origin (DCO), where users add "signed-off-by" in their commits and the project links to the DCO website. However, this MAY be implemented as a Contributor License Agreement (CLA), or other legal mechanism. (URL required) [dco] | *ONAP project common response*<br><br>This question can be answered the same ONAP-wide. | ONAP requires both a **Developer Certificate of Origin** (DCO), and a **Contributor License Agreemen**t (CLA).<br><br>https://wiki.onap.org/display/DW/Contribution+Agreements |
| | The project MUST clearly define and document its project governance model (the way it makes decisions, including key roles). (URL required) [governance] | *ONAP project common response*<br><br>This question can be answered the same ONAP-wide. | The project governance is described at<br><br>https://wiki.onap.org/display/DW/Community+Offices+and+Governance |

| | | | |
|---|---|---|---|
| | | | Further information can be found at https://wiki.onap.org/display/DW/ONAP+Technical+Community+Document |
| | The project MUST adopt a code of conduct and post it in a standard location. (URL required) [code_of_conduct] | *ONAP project common response*<br><br>This question can be answered the same ONAP-wide.<br><br>ONAP adheres to the Linux Foundation Code of Conduct, so you can just select the Met radio button. | ONAP adheres to the Linux Foundation Code of Conduct, found at https://lfprojects.org/policies/code-of-conduct/ |
| | The project MUST clearly define and publicly document the key roles in the project and their responsibilities, including any tasks those roles must perform. It MUST be clear who has which role(s), though this might not be documented in the same way. (URL required) [roles_responsibilities] | *ONAP project common response*<br><br>This question can be answered the same ONAP-wide. | The key roles in the project and their responsibilities are described at<br><br>https://wiki.onap.org/display/DW/Community+Offices+and+Governance<br><br>Current members are listed at<br><br>https://wiki.onap.org/pages/viewpage.action?pageId=8226539 |
| | The project MUST be able to continue with minimal interruption if any one person is incapacitated or killed. In particular, the project MUST be able to create and close issues, accept proposed changes, and release versions of software, within a week of confirmation that an individual is incapacitated or killed. This MAY be done by ensuring someone else has any necessary keys, passwords, and legal rights to continue the project. Individuals who run a FLOSS project MAY do this by providing keys in a lockbox and a will providing any needed legal rights (e.g., for DNS names). (URL required) [access_continuity] | ONAP uses the Linux Foundation structure to support all projects, including all keys and passwords. Nothing, including all legal rights, is invested in any single person. | For AAI-UI we have 4 committers and multiple contributes who are listed in https://wiki.onap.org/display/DW/Resources+and+Repositories#ResourcesandRepositories-ActiveandAvailableInventory<br>All 4 committers have access and rights to maintain the code base, approve and review incoming changes and release a new version of the artifact. This will let the project continue with minimal to no interruption if one person is incapacitated. Also this project is controlled by the linux foundation so we can add more committers if needed |
| | The project SHOULD have a "bus factor" of 2 or more. (URL required) [bus_factor] | | All the projects covered in this report have more than 2 persons who actively contribute and maintain code.<br>The following link provides with list of commit owners for the project which list at least these 4 persons(Arul Nambi, Dave Adams, Francis Paquette, Richard von Dadelszen) in common in alphabetical order<br>https://gerrit.onap.org/r/#/q/project:aai/sparky-be+branch:master<br>https://gerrit.onap.org/r/#/q/project:aai/router-core+branch:master<br>https://gerrit.onap.org/r/#/q/project:aai/data-router+branch:master<br>https://gerrit.onap.org/r/#/q/project:aai/search-data-service+branch:master |
| | **Question** | **Description** | **Sample Answer** |
| | **Basics: Documentation** | | |
| | The project MUST have a documented roadmap that describes what the project intends to do and not do for at least the next year. (URL required) [documentation_roadmap] | This should be answered on a per-project basis. | Road map for AAI can be found in https://wiki.onap.org/display/DW/AAI+R3+Platform+Maturity?preview=%2F35523310%2F35523289%2FAAI+Roadmap+2018-01-05.pdf |
| | The project MUST include documentation of the architecture (aka high-level design) of the software produced by the project. If the project does not produce software, select "not applicable" (N/A). (URL required) [documentation_architecture] | This should be answered on a per-project basis. | Architecture of AAI can be found in https://wiki.onap.org/pages/viewpage.action?pageId=1015836 |
| | The project MUST document what the user can and cannot expect in terms of security from the software produced by the project (its "security requirements"). (URL required) [documentation_security] | This should be answered on a per-project basis. | Point to a security overview on wiki.onap.org. |
| | The project MUST provide a "quick start" guide for new users to help them quickly do something with the software. (URL required) [documentation_quick_start] | *ONAP project common response*<br><br>This question can be answered the same ONAP-wide. | Information on setting up ONAP can be found at https://onap.readthedocs.io/en/latest/guides/onap-developer/settingup/index.html |
| | The project MUST make an effort to keep the documentation consistent with the current version of the project results (including software produced by the project). Any *known* documentation defects making it inconsistent MUST be fixed. If the documentation is generally current, but erroneously includes some older information that is no longer true, just treat that as a defect, then track and fix as usual. [documentation_current] | *ONAP project common response* | Documentation is updated with each release. |
| | The project repository front page and/or website MUST identify and hyperlink to any achievements, including this best practices badge, within 48 hours of public recognition that the achievement has been attained. (URL required) [documentation_achievements] | **What is needed**<br><br>You will have to add the image and the link to the CII badging to your projects. Both you will need your project-idenfier information for that.<br><br>**How to retrieve your project identifier**<br><br>eg for the project AAI-UI the CII link is<br><br>https://bestpractices.coreinfrastructure.org/projects/1737<br><br>And the project identifier is 1737<br><br>**How to generate 2 urls**<br><br>To add the link and image you will need 2 URL's both are relative to your specific project<br><br>eg | The badge is visible on the project's readme.io page found at <project readme.io url> |

| | | | |
|---|---|---|---|
| | | Image: https://bestpractices.coreinfrastructure.org/projects/1737/badge<br><br>Link: https://bestpractices.coreinfrastructure.org/projects/1737<br><br>## Examples for diff types of readme<br><br>Once you have this information you will need to add them to your readme file. The readme files are in different formats. Examples for *.MD and *.rst are follows<br><br>### *.MD:<br><br>[![alt text](https://bestpractices.coreinfrastructure.org/projects/1737/badge)](https://bestpractices.coreinfrastructure.org/projects/1737)<br><br>### *.rst<br><br>.. image:: https://bestpractices.coreinfrastructure.org/projects/1737/badge<br>:height: 30px<br>:width: 200 px<br>:scale: 100 %<br>:alt: alternate text<br>:align: right<br>:target: https://bestpractices.coreinfrastructure.org/projects/1737<br><br>After doing so, click Met. | |
| | **Question** | **Description** | **Sample Answer** |
| | **Basics: Accessibility and internationalization** | | |
| | The project (both project sites and project results) SHOULD follow accessibility best practices so that persons with disabilities can still participate in the project and use the project results where it is reasonable to do so. [accessibility_best_practices] | Any UI based project should would need to consider Accessibility and internalization.<br><br>## Resources<br><br>http://brebru.com/css/accessibility.html<br><br>https://achecker.ca/checker/index.php | The following projects do not have any user interface. They all expose API's and fulfill a contract<br><br>https://gerrit.onap.org/r/gitweb?p=aai/router-core.git;a=blob;f=License.txt;h=c97ccb2ef72c2f18a5299fa8e8c380e01fcc109a;hb=refs/heads/master<br>https://gerrit.onap.org/r/gitweb?p=aai/data-router.git;a=blob;f=License.txt;h=5eb27145f1c3d60f9504291372c92b05ed7a144c;hb=refs/heads/master<br>https://gerrit.onap.org/r/gitweb?p=aai/search-data-service.git;a=blob;f=License.txt;h=df05b974d8a26b730d141f1945a8604ab683f4ae;hb=refs/heads/master<br>https://gerrit.onap.org/r/gitweb?p=aai/sparky-be.git;a=blob;f=LICENSE;h=c8636af62de206e3591b954c4317ad2a56a85cf9;hb=refs/heads/master |
| | The software produced by the project SHOULD be internationalized to enable easy localization for the target audience's culture, region, or language. If internationalization (i18n) does not apply (e.g., the software doesn't generate text intended for end-users and doesn't sort human-readable text), select "not applicable" (N/A). [internationalization] | | The following projects do not have any user interface. They all expose API's and fulfill a contract<br><br>https://gerrit.onap.org/r/gitweb?p=aai/router-core.git;a=blob;f=License.txt;h=c97ccb2ef72c2f18a5299fa8e8c380e01fcc109a;hb=refs/heads/master<br>https://gerrit.onap.org/r/gitweb?p=aai/data-router.git;a=blob;f=License.txt;h=5eb27145f1c3d60f9504291372c92b05ed7a144c;hb=refs/heads/master<br>https://gerrit.onap.org/r/gitweb?p=aai/search-data-service.git;a=blob;f=License.txt;h=df05b974d8a26b730d141f1945a8604ab683f4ae;hb=refs/heads/master<br>https://gerrit.onap.org/r/gitweb?p=aai/sparky-be.git;a=blob;f=LICENSE;h=c8636af62de206e3591b954c4317ad2a56a85cf9;hb=refs/heads/master |
| | **Question** | **Description** | **Sample Answer** |
| | **Basics: Other** | | |
| | If the project sites (website, repository, and download URLs) store passwords for authentication of external users, the passwords MUST be stored as iterated hashes with a per-user salt by using a key stretching (iterated) algorithm (e.g., PBKDF2, Bcrypt or Scrypt). If the project sites do not store passwords for this purpose, select "not applicable" (N/A). [sites_password_security] | *ONAP project common response* | Mark this as N/A.<br><br>The project does not store passwords in the website, repository or downloads, but uses the Linux Foundation common login service. |
| | **Question** | **Description** | **Sample Answer** |
| | **Change Control: Previous versions** | | |
| | The project MUST maintain the most often used older versions of the product *or* provide an upgrade path to newer versions. If the upgrade path is difficult, the project MUST document how to perform the upgrade (e.g., the interfaces that have changed and detailed suggested steps to help upgrade). [maintenance_or_update] | *ONAP project common response* | All major releases are tagged in gerrit and the artifacts are stored with the release information on onap.nexus. So we can access all old versions of the artifact. If and when a upgrade requires certain steps to be followed they are being added to the release documents as needed |

| | Question | Description | Sample Answer |
|---|---|---|---|
| | **Change Control: Vulnerability report process** | | |
| | The project MUST give credit to the reporter(s) of all vulnerability reports resolved in the last 12 months, except for the reporter(s) who request anonymity. If there have been no vulnerabilities resolved in the last 12 months, select "not applicable" (N/A). (URL required) [vulnerability_report_credit] | *ONAP project common response*<br><br>ONAP-Wide | Vulnerabilities can be reported using the link https://wiki.onap.org/pages/viewpage.action?pageId=6591711 Currently we dont have any vulnerabilities reported, but the wiki page explains on how to report a vulnerability and how to report anonymously if you do not want the credit for it. |
| | The project MUST have a documented process for responding to vulnerability reports. (URL required) [vulnerability_response_process] | *ONAP project common response*<br><br>ONAP-WIDE | Vulnerabilities handling is documented in https://wiki.onap.org/pages/viewpage.action?pageId=6591711 |
| | **Question** | **Description** | **Sample Answer** |
| | **Quality: Coding standards** | | |
| | The project MUST identify the specific coding style guides for the primary languages it uses, and require that contributions generally comply with it. (URL required) [coding_standards] | *ONAP project common response*<br><br>ONAP-WIDE | Coding style is defined in https://wiki.onap.org/display/DW/Java+code+style |
| | The project MUST automatically enforce its selected coding style(s) if there is at least one FLOSS tool that can do so in the selected language(s). [coding_standards_enforced] | | For sparky-fe we use eslint to force code styling. |
| | **Question** | **Description** | **Sample Answer** |
| | **Quality: Working build system** | | |
| | Build systems for native binaries MUST honor the relevant compiler and linker (environment) variables passed in to them (e.g., CC, CFLAGS, CXX, CXXFLAGS, and LDFLAGS) and pass them to compiler and linker invocations. A build system MAY extend them with additional flags; it MUST NOT simply replace provided values with its own. If no native binaries are being generated, select "not applicable" (N/A). [build_standard_variables] | | The application does not create native binaries. (Some of the libraries it depends on do, but those are external.) |
| | The build and installation system SHOULD preserve debugging information if they are requested in the relevant flags (e.g., "install -s" is not used). If there is no build or installation system (e.g., typical JavaScript libraries), select "not applicable" (N/A).[build_preserve_debug] | | The application does not create native binaries. (Some of the libraries it depends on do, but those are external.) |
| | The build system for the software produced by the project MUST NOT recursively build subdirectories if there are cross-dependencies in the subdirectories. If there is no build or installation system (e.g., typical JavaScript libraries), select "not applicable" (N/A). [build_non_recursive] | | The application does not create native binaries. (Some of the libraries it depends on do, but those are external.) |
| | The project MUST be able to repeat the process of generating information from source files and get exactly the same bit-for-bit result. If no building occurs (e.g., scripting languages where the source code is used directly instead of being compiled), select "not applicable" (N/A). [build_repeatable] | | All releases are tagged in gerrit(git), and the builds are controlled using jenkins. By providing the git tag information the same image can be build over and over again with same bit-for-bit result. |
| | **Question** | **Description** | **Sample Answer** |
| | **Quality: Installation system** | | |
| | The project MUST provide a way to easily install and uninstall the software produced by the project using a commonly-used convention. [installation_common] | | All packages are delivered either as an jar artifact or a docker image.<br>In case of maven artifacts, they can be removed using the pom file.<br>In case of docker container. We can delete the container we dont want.<br>Also control the orchestration in Kubernetes if you want to exclude certain docker images. |
| | The installation system for end-users MUST honor standard conventions for selecting the location where built artifacts are written to at installation time. For example, if it installs files on a POSIX system it MUST honor the DESTDIR environment variable. If there is no installation system or no standard convention, select "not applicable" (N/A). [installation_standard_variables] | | The compiled docker images and jar files can be installed/used as the user sees fit. Both run on JVM or docker. So there is no reason to selecting locations etc. |
| | The project MUST provide a way for potential developers to quickly install all the project results and support environment necessary to make changes, including the tests and test environment. This MUST be performed with a commonly-used convention. [installation_development_quick] | *ONAP project common response* | All the components require only java and maven to begin with for a developer to quickly install and test it.<br>Even for deployment using OOM and the right amount of resources, we can deploy the full AAI/ONAP suite in less than a day. The steps are documented in https://onap.readthedocs.io/en/latest/submodules/oom.git/docs/oom_quickstart_guide.html |
| | **Question** | **Description** | **Sample Answer** |
| | **Quality: Externally-maintained components** | | |
| | The project MUST list external dependencies in a computer-processable way. (URL required) [external_dependencies] | | External dependencies are controlled using the pom file, which can be found in the root folder for the projects<br><br>https://gerrit.onap.org/r/gitweb?p=aai/sparky-be.git;a=tree;h=refs/heads/master;hb=refs/heads/master<br>https://gerrit.onap.org/r/gitweb?p=aai/data-router.git;a=tree;h=refs/heads/master;hb=refs/heads/master<br>https://gerrit.onap.org/r/gitweb?p=aai/router-core.git;a=tree;h=refs/heads/master;hb=refs/heads/master<br>https://gerrit.onap.org/r/gitweb?p=aai/search-data-service.git;a=tree;h=refs/heads/master;hb=refs/heads/master |
| | Projects MUST monitor or periodically check their external dependencies (including convenience copies) to detect known vulnerabilities, and fix exploitable vulnerabilities or verify them as unexploitable. [dependency_monitoring] | *ONAP project common response* | NexusIQ sonar scan is run on all the projects on a weekly basis |
| | The project MUST either: | *ONAP project common response* | External components are maintained |

| | | | |
|---|---|---|---|
| | 1. make it easy to identify and update reused externally-maintained components; **or**<br>2. use the standard components provided by the system or programming language.<br><br>Then, if a vulnerability is found in a reused component, it will be easy to update that component. [updateable_reused_components] | | through Maven. The user can get a list of all included components using the maven dependency tree and can update or reuse as they see fit |
| | The project SHOULD avoid using deprecated or obsolete functions and APIs where FLOSS alternatives are available in the set of technology it uses (its "technology stack") and to a supermajority of the users the project supports (so that users have ready access to the alternative). [interfaces_current] | | We avoid depending on deprecated /obsolete functions. |
| | **Question** | **Description** | **Sample Answer** |
| | **Quality: Automated test suite** | | |
| | An automated test suite MUST be applied on each check-in to a shared repository for at least one branch. This test suite MUST produce a report on test success or failure. [automated_integration_testing] | *ONAP project common response* | Automatic test suites are run every time before merging the code. The code check in cannot pass with out jenkins posting a +1 on the review. |
| | The project MUST add regression tests to an automated test suite for at least 50% of the bugs fixed within the last six months. [regression_tests_added50] | | When regressions occur, we add tests for them. |
| | The project MUST have FLOSS automated test suite(s) that provide at least 80% statement coverage if there is at least one FLOSS tool that can measure this criterion in the selected language. [test_statement_coverage80] | | We use sonar to measure the code coverage.<br>https://sonar.onap.org/about<br>Code coverage at the date of filling this report(2018-09-19) is<br>Sparky-be: 48.7<br>Data-router: 51.5<br>Router-core: 69.2<br>search-data-service: 54 |
| | **Question** | **Description** | **Sample Answer** |
| | **Quality: New functionality testing** | | |
| | The project MUST have a formal written policy that as major new functionality is added, tests for the new functionality MUST be added to an automated test suite. [test_policy_mandated] | *ONAP project common response* | Contributing guide lines for development is recorded in https://wiki.onap.org/display/DW/Development+Procedures+and+Policies |
| | The project MUST include, in its documented instructions for change proposals, the policy that tests are to be added for major new functionality. [tests_documented_added] | *ONAP project common response* | This is documented on our wiki: Code Coverage and Static Code Analysis |
| | **Question** | **Description** | **Sample Answer** |
| | **Quality: Warning flags** | | |
| | Projects MUST be maximally strict with warnings in the software produced by the project, where practical. | | Build systems run the compile with test flag enabled by default. So any failure in test cases will fail the ci and the merge request. |
| | **Question** | **Description** | **Sample Answer** |
| | **Secure development knowledge** | | |
| | The project MUST implement secure design principles (from "know_secure_design"), where applicable. If the project is not producing software, select "not applicable" (N/A). [implement_secure_design | Todo<br><br>Even though we can currently say that we implement secure design to the best of our knowledge there is no documentation on the accepted standards so all projects should either have their own guidelines and principles or we need to create a onap-wide page<br><br>Sample pages<br><br>https://docs.zephyrproject.org/latest/security/secure-coding.html<br><br>https://github.com/coreinfrastructure/best-practices-badge/blob/master/doc/security.md | The project strives to implement secure design principles. |
| | **Question** | **Description** | **Sample Answer** |
| | **Security: Use basic good cryptographic practices** | **In general if your product does not deal with any cryptographic mechanism you can select N/A for all the questions in this section. Also on the CII badging page you can see that this option has a button with text "Press here if the software produced by the project does not use cryptographic mechanisms"; clicking on this will select N/A for all the questions in this section** | |
| | 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).[crypto_weaknesses] | | All generated certificates are generated with sha256, and furthermore only JWE approved ciphers are used. |
| | The project SHOULD support multiple cryptographic algorithms, so users can quickly switch if one is broken. Common symmetric key algorithms include AES, Twofish, and Serpent. Common cryptographic hash algorithm alternatives include SHA-2 (including SHA-224, SHA-256, SHA-384 AND SHA-512) and SHA-3. | | Certificates are managed through AAF micro-service which will be deployed with ONAP suite |
| | The project MUST support storing authentication credentials (such as passwords and dynamic tokens) and private cryptographic keys in files that are separate from other information (such as configuration files, databases, and logs), and permit users to update and replacement them without code recompilation. If the project never processes authentication credentials and private cryptographic keys, select "not applicable" (N/A). | | |
| | The software produced by the project SHOULD support secure protocols for all of its network communications, such as SSHv2 or later, TLS1.2 or later (HTTPS), IPsec, SFTP, and SNMPv3. Insecure protocols such as FTP, HTTP, telnet, SSLv3 or earlier, and SSHv1 SHOULD be disabled by default, and only enabled if the user specifically configures it. If the software produced by the project does not support network comunications, select "not applicable" (N/A). [crypto_used_network] | | The projects supports secure TLS and HTTPS and Insecure protocols are disabled by default in these applications, they cab be over-ridden by user configuration |
| | The software produced by the project SHOULD, if it supports or uses TLS, support at least TLS version 1.2. Note that the predecessor of TLS was called SSL. If the software does not use TLS, select "not applicable" (N/A). [crypto_tls12] | | The products support TLS version 1.2 |

| | | | |
|---|---|---|---|
| | The software produced by the project MUST, if it supports TLS, perform TLS certificate verification by default when using TLS, including on subresources. If the software does not use TLS, select "not applicable" (N/A). [crypto_certificate_verification] | | Certificate validation is done before answering any calls. |
| | The software produced by the project MUST, if it supports TLS, perform certificate verification before sending HTTP headers with private information (such as secure cookies). If the software does not use TLS, select "not applicable" (N/A).[crypto_verification_private] | | The certificate is validated before sending http headers or private information. |
| | **Question** | **Description** | **Sample Answer** |
| | **Security: Secure release** | | |
| | The project MUST cryptographically sign releases of the project results intended for widespread use, and there MUST be a documented process explaining to users how they can obtain the public signing keys and verify the signature(s). The private key for these signature(s) MUST NOT be on site(s) used to directly distribute the software to the public. If releases are not intended for widespread use, select "not applicable" (N/A). [signed_releases] | *ONAP project common response*<br><br>TDB:<br><br>I think we will have to discuss with LF to see how we can sign out releases and also create a page on onap wiki may be on a per project basis? | All release artifacts are signed by the Linux Foundation prior to release. |
| | It is SUGGESTED that in the version control system, each important version tag (a tag that is part of a major release, minor release, or fixes publicly noted vulnerabilities) be cryptographically signed and verifiable as described in signed_releases. [version_tags_signed] | TBD:<br><br>Need to talk to LF to see how we can sign our tagged releases | |
| | **Question** | **Description** | **Sample Answer** |
| | **Other Security Issues** | | |
| | The project results MUST check all inputs from potentially untrusted sources to ensure they are valid (a *whitelist*), and reject invalid inputs, if there are any restrictions on the data at all. [input_validation] | | The project strives to validate all input to functions. The inputs that are provided to the services are checked against existing models such as OXM or search-abstraction layer and only valid inputs are allowed to be pass through |
| | Hardening mechanisms SHOULD be used in the software produced by the project so that software defects are less likely to result in security vulnerabilities. [hardening] | | The project tries to use hardening mechanism whenever possible. Eg we use transaction id for tracking transactions through multiple services and also we use http headers to identify the application where possible |
| | The project MUST provide an assurance case that justifies why its security requirements are met. The assurance case MUST include: a description of the threat model, clear identification of trust boundaries, an argument that secure design principles have been applied, and an argument that common implementation security weaknesses have been countered. (URL required)[assurance_case] | This builds on the document prepared for documentation_security by documenting how those security goals have been met. This can be done either within the documentation_security document itself or in a separate document.<br><br>This should be answered on a per-project basis. | Point to a security assurance case on wiki.onap.org. |
| | **Question** | **Description** | **Sample Answer** |
| | **Analysis: Static code analysis** | | |
| | The project MUST use at least one static analysis tool with rules or approaches to look for common vulnerabilities in the analyzed language or environment, if there is at least one FLOSS tool that can implement this criterion in the selected language. [static_analysis_common_vulnerabilities] | | https://sonar.onap.org |
| | **Question** | **Description** | **Sample Answer** |
| | **Analysis: Dynamic code analysis** | | |
| | If the software produced by the project includes software written using a memory-unsafe language (e.g., C or C++), then at least one dynamic tool (e.g., a fuzzer or web application scanner) MUST be routinely used in combination with a mechanism to detect memory safety problems such as buffer overwrites. If the project does not produce software written in a memory-unsafe language, choose "not applicable" (N/A). [dynamic_analysis_unsafe] | | All the projects use Java which are memory safe that run on JVM. Also the end product runs on a docker container which is run on docker. |

---

## Sample Gold Level Questions and Answers

This section needs to be filled in.

| | Question | Description | Sample Answer |
|---|---|---|---|
| | **Basics: Identification**<br><br>**Basics: Prerequisites**<br><br>**Basics: Project oversight**<br><br>**Basics: Other** | The questions in these Basics sections will be filled in automatically.<br><br>Some questions change SHOULDs from previous levels to MUSTs. | |
| | The project MUST include a copyright statement in each source file, identifying at least one relevant year and copyright holder. [copyright_per_file] | *ONAP project common response* | all source and documentation files are required to have copyright notices |
| | The project MUST include a license statement in each source file. This MAY be done by including the following inside a comment near the | *ONAP project common response* | all source and documentation files are required to have license statements |

| | | | |
|---|---|---|---|
| | beginning of each file: SPDX-License-Identifier: [SPDX license expression for project]. [license_per_file] | | |
| | **Question** | **Description** | **Sample Answer** |
| | **Change Control: Public version-controlled source repository** | | |
| | The project's source repository MUST use a common distributed version control software (e.g., git or mercurial). [repo_distributed] | This question will be filled in automatically from previous levels. | |
| | The project MUST clearly identify small tasks that can be performed by new or casual contributors. (URL required) [small_tasks] | TBD<br><br>DO WE HAVE POLICIES ON THIS? | |
| | The project MUST require two-factor authentication (2FA) for developers for changing a central repository or accessing sensitive data (such as private vulnerability reports). This 2FA mechanism MAY use mechanisms without cryptographic mechanisms such as SMS, though that is not recommended. [require_2FA] | *ONAP project common response*<br><br>This will need to be resolved on an ONAP project basis. We cannot currently answer MET on this item. | Capability is not currently supported. |
| | The project's two-factor authentication (2FA) SHOULD use cryptographic mechanisms to prevent impersonation. Short Message Service (SMS) based 2FA, by itself, does NOT meet this criterion, since it is not encrypted. [secure_2FA] | *ONAP project common response*<br><br>This will need to be resolved on an ONAP project basis. We cannot currently answer MET on this item. | Capability is not currently supported. |
| | **Question** | **Description** | **Sample Answer** |
| | **Quality: Coding standards** | | |
| | The project MUST document its code review requirements, including how code review is conducted, what must be checked, and what is required to be acceptable. (URL required) [code_review_standards] | TBD<br><br>DO WE HAVE POLICIES ON THIS? | |
| | The project MUST have at least 50% of all proposed modifications reviewed before release by a person other than the author, to determine if it is a worthwhile modification and free of known issues which would argue against its inclusion [two_person_review] | *ONAP project common response*<br><br>ONAP requires a committer other than the submitter to review each proposed modification.<br><br>https://wiki.onap.org/display/DW/Code+Review | Per https://wiki.onap.org/display/DW/Code+Review, self-commits are not allowed. |
| | **Question** | **Description** | **Sample Answer** |
| | **Quality: Working build system** | | |
| | The project MUST have a reproducible build. If no building occurs (e.g., scripting languages where the source code is used directly instead of being compiled), select "not applicable" (N/A). (URL required) [build_reproducible] | TBD<br><br>AFAIK, WE DO NOT CURRENTLY HAVE A POLICY ON THIS | |
| | **Question** | **Description** | **Sample Answer** |
| | **Quality: Automated test suite** | These questions will be filled in automatically from previous levels. | |
| | The project MUST implement continuous integration, where new or changed code is frequently integrated into a central code repository and automated tests are run on the result. (URL required) [test_continuous_integration] | *ONAP project common response*<br><br>ONAP uses continuous integration and unit tests run automatically during CI. | Junit tests are invoked from mvn. Pytest tests are invoked by running pytest from command line. Rebar3 tests are invoked from command line by running rebarr3. All are included as part of Jenkin builds. All are standard testing tools invoked in standard way.<br>Robot Framework tests are invoked by standard Robot methodology, also triggered by Jenkins build jobs.<br>https://wiki.onap.org/display/DW/Continuous+Integration<br>https://wiki.onap.org/pages/viewpage.action?pageId=4718718 |
| | A test suite MUST be invocable in a standard way for that language. (URL required) [test_invocation] | TBD<br><br>WHERE IS THIS DOCUMENTED? A URL IS REQUIRED. | |
| | **Question** | **Description** | **Sample Answer** |
| | **Security: Use basic good cryptographic practices** | These questions will be filled in automatically from previous levels. | |

| | | | |
|---|---|---|---|
| | **Security: Secured delivery against man-in-the-middle (MITM) attacks**<br><br>**Security: Publicly known vulnerabilities fixed** | | |
| | The project website, repository (if accessible via the web), and download site (if separate) MUST include key hardening headers with nonpermissive values. (URL required) [hardened_site] | *ONAP project common response*<br><br>This will need to be resolved on an ONAP project basis. We cannot currently answer MET on this item. | Cannot be met yet. Select unmet. For a reason, type:<br><br>// X-Content-Type-Options was not set to "nosniff".<br><br>Details on why this cannot be set to Met:<br><br>The project website, wiki.onap.org, has these headers. https://securityheaders.io/ gives an A grade. content-security-policy: frame-ancestors 'self' strict-transport-security: max-age=15552000 x-content-type-options: nosniff x-frame-options: SAMEORIGIN x-xss-protection: 1; mode=block<br><br>The code repository, gerrit.onap.org, has some of the headers. https://securityheaders.io/ gives a D grade. content-security-policy: NOT PRESENT Strict-Transport-Security: max-age=15552000 X-Content-Type-Options: NOT PRESENT X-Frame-Options: X-XSS-protection: NOT PRESENT<br><br>The download site, nexus.onap.org, has some of the headers. https://securityheaders.io/ gives a C grade. content-security-policy: NOT PRESENT strict-transport-security: max-age=15552000 x-content-type-options: nosniff x-frame-options: SAMEORIGIN x-xss-protection: NOT PRESENT // X-Content-Type-Options was not set to "nosniff". |
| | **Question** | **Description** | **Sample Answer** |
| | **Analysis: Dynamic code analysis** | Some questions in the Analysis section will be automatically filled in from previous levels.<br><br>The remaining questions in the Analysis section must be individually answered according to your project. | |

# Sample Testing tools

The following is a list of some of the testing tools available. These may be considered for evaluation vis-a-vis the specific project requirements and open source license requirements.

| S.No. | Tool | Website | Java | Javascript | Ruby | Python | C++ |
|---|---|---|---|---|---|---|---|
| 1 | Mockito | http://site.mockito.org/ | X | | | | |
| 2 | Powermock | https://github.com/powermock/powermock | X | | | | |
| 3 | Junit | http://junit.org/junit4/ | X | | | | |
| 4 | TestNG | http://testng.org/doc/ | X | | | | |
| 5 | Robot Framework | http://robotframework.org/ | | | | X | |
| 6 | Mocha | https://mochajs.org/ | | X | | | |
| 7 | Jasmine | https://jasmine.github.io/ | | X | | | |
| 8 | Nemo.js | http://nemo.js.org/ | | X | | | |
| 9 | CasperJS | http://casperjs.org/ | | X | | | |
| 10 | cucumber | https://cucumber.io/ | | | X | | |
| 11 | Watir | https://watir.com/ | | | X | | |
| 12 | Google Test | https://github.com/google/googletest | | | | | X |
| 13 | Boost | http://www.boost.org/ | | | | | X |
| 14 | cppUnit | https://sourceforge.net/projects/cppunit/ | | | | | X |
| 15 | Cacth | https://github.com/catchorg/Catch2 | | | | | X |
| 16 | TUT | https://cpputest.github.io/ | | | | | X |

# Resources

The following resources may be useful source of information about CII badging:

•CII Badging overview: https://bestpractices.coreinfrastructure.org/
•Basic Criteria: https://github.com/coreinfrastructure/best-practices-badge/blob/master/doc/criteria.md
•Higher Level Criteria: CII Badging overview : https://github.com/coreinfrastructure/best-practices-badge/blob/master/doc/other.md
•Example : https://bestpractices.coreinfrastructure.org/projects/1/0
•Further reading: https://wiki.onap.org/display/DW/ONAP+Beijing+Release+Developer+Forum%2C+Dec.+11-13%2C+2017%2C+Santa+Clara%2C+CA+US?preview=/16002054/20874916/ONAP-Security%20Sub-committee-pa2.pdf
•CLAMP project CII: https://bestpractices.coreinfrastructure.org/projects/1197
•http://tlhansen.us/onap/cii.html [ONAP Project Status dashboard]

The project MUST support storing authentication credentials (such as passwords and dynamic tokens) and private cryptographic keys in files that are separate from other information (such as configuration files, databases, and logs), and permit users to update and replacement them without code recompilation. If the project never processes authentication credentials and private cryptographic keys, select "not applicable" (N/A).

# Status of Security Application Quality Requirements on Nov 13, 2019

There are six Security requirements affecting Application Quality at the **Passing** level, and all but two projects have Met these or marked them N/A.

Below are the Security requirements affecting Application Quality for the **Silver** and **Gold** badging levels. Most projects have already answered a large percentage of these, with the majority answering Met or N/A. The project counts are given. There are three requirements that have a low answer rate, and those are shown in red.

## Silver / Gold

### assurance_case (5 of 38 projects are Met or N/A)

The project MUST provide an assurance case that justifies why its security requirements are met. The assurance case MUST include: a description of the threat model, clear identification of trust boundaries, an argument that secure design principles have been applied, and an argument that common implementation security weaknesses have been countered. (URL required)

### hardening (9 of 38 projects are Met or N/A)

Hardening mechanisms MUST be used in the software produced by the project so that software defects are less likely to result in security vulnerabilities.

### crypto_algorithm_agility (24 of 38 projects are Met or N/A)

The project MUST support multiple cryptographic algorithms, so users can quickly switch if one is broken. Common symmetric key algorithms include AES, Twofish, and Serpent. Common cryptographic hash algorithm alternatives include SHA-2 (including SHA-224, SHA-256, SHA-384 AND SHA-512) and SHA-3.

### crypto_certificate_verification (30 of 38 projects are Met or N/A)

The software produced by the project MUST, if it supports TLS, perform TLS certificate verification by default when using TLS, including on subresources. If the software does not use TLS, select 'not applicable' (N/A).

### crypto_credential_agility (20 of 38 projects are Met or N/A)

The project MUST support storing authentication credentials (such as passwords and dynamic tokens) and private cryptographic keys in files that are separate from other information (such as configuration files, databases, and logs), and permit users to update and replace them without code recompilation. If the project never processes authentication credentials and private cryptographic keys, select 'not applicable' (N/A).

### crypto_tls12 (27 of 38 projects are Met or N/A)

The software produced by the project MUST, if it supports or uses TLS, support at least TLS version 1.2. Note that the predecessor of TLS was called SSL. If the software does not use TLS, select 'not applicable' (N/A).

### crypto_used_network (26 of 38 projects are Met or N/A)

The software produced by the project MUST support secure protocols for all of its network communications, such as SSHv2 or later, TLS1.2 or later (HTTPS), IPsec, SFTP, and SNMPv3. Insecure protocols such as FTP, HTTP, telnet, SSLv3 or earlier, and SSHv1 SHOULD be disabled by default, and only enabled if the user specifically configures it. If the software produced by the project does not support network communications, select 'not applicable' (N/A).

### crypto_verification_private  (26 of 38 projects are Met or N/A)

The software produced by the project MUST, if it supports TLS, perform certificate verification before sending HTTP headers with private information (such as secure cookies). If the software does not use TLS, select 'not applicable' (N/A).

### crypto_weaknesses (35 of 38 projects are Met or N/A)

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).

### implement_secure_design (22 of 38 projects are Met or N/A)

The project MUST implement secure design principles (from 'know_secure_design'), where applicable. If the project is not producing software, select 'not applicable' (N/A).

### input_validation (21 of 38 projects are Met or N/A)

The project results MUST check all inputs from potentially untrusted sources to ensure they are valid (a *whitelist*), and reject invalid inputs, if there are any restrictions on the data at all.

## Gold Only

### security_review  (2 of 38 projects are Met or N/A)

The project MUST have performed a security review within the last 5 years. This review MUST consider the security requirements and security boundary.