



Sandia  
National  
Laboratories

# The OpenSSF Best Practices Badge Program

Dr. Roscoe A. Bartlett, PhD  
Department of Software Engineering and Research  
Sandia National Laboratories

June 14, 2023

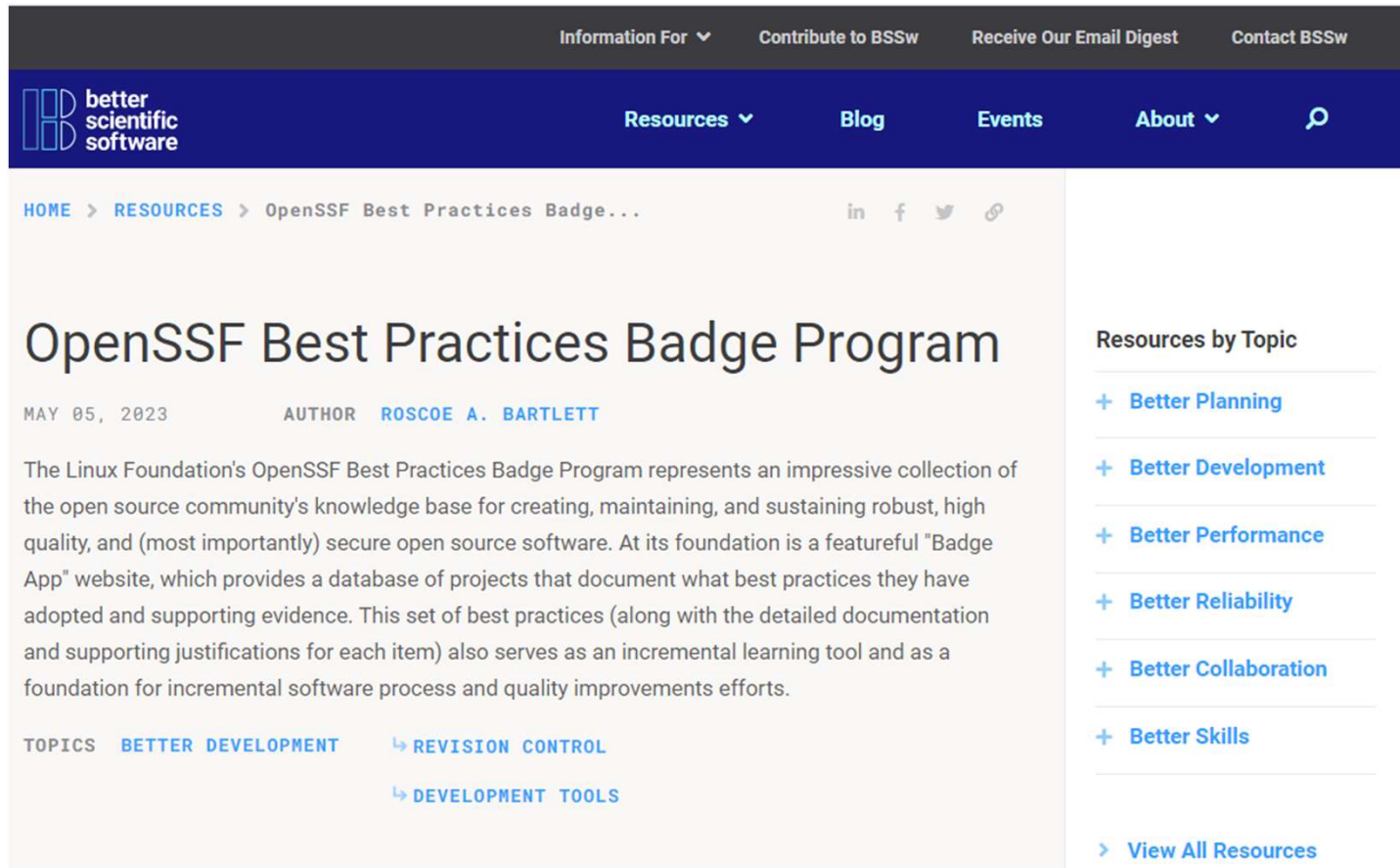
HPC Best Practices Webinar Series



Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

SAND2023-048190

# OpenSSF Best Practices Badge Program: BSSW.io Article



The screenshot shows the BSSW.io website interface. At the top, there is a dark blue navigation bar with the BSSW logo on the left and links for 'Information For', 'Contribute to BSSW', 'Receive Our Email Digest', and 'Contact BSSW' on the right. Below this is a secondary dark blue bar with 'Resources', 'Blog', 'Events', and 'About' menus, along with a search icon. The main content area has a light beige background. The article title 'OpenSSF Best Practices Badge Program' is prominently displayed. Below the title, the date 'MAY 05, 2023' and author 'ROSCOE A. BARTLETT' are listed. The article text describes the Linux Foundation's OpenSSF Best Practices Badge Program as a collection of open source community knowledge for creating, maintaining, and sustaining robust, high-quality, and secure open source software. It mentions a 'Badge App' website that provides a database of projects and their best practices. At the bottom of the article, there are topic tags: 'TOPICS BETTER DEVELOPMENT', 'REVISION CONTROL', and 'DEVELOPMENT TOOLS'. On the right side of the page, there is a sidebar titled 'Resources by Topic' with a list of categories: 'Better Planning', 'Better Development', 'Better Performance', 'Better Reliability', 'Better Collaboration', and 'Better Skills'. A 'View All Resources' link is at the bottom of the sidebar.

<https://bssw.io/items/openssf-best-practices-badge-program>

# White House Executive Order 14028



THE WHITE HOUSE

Administration Priorities The Record Briefing Room Español MENU

MAY 12, 2021

## Executive Order on Improving the Nation's Cybersecurity

BRIEFING ROOM PRESIDENTIAL ACTIONS

The screenshot shows the top portion of the White House website. It includes the 'THE WHITE HOUSE' logo on the left, a navigation menu with links for 'Administration', 'Priorities', 'The Record', 'Briefing Room', and 'Español', and a 'MENU' button on the right. The date 'MAY 12, 2021' is centered above the main title. The title 'Executive Order on Improving the Nation's Cybersecurity' is prominently displayed in a large, dark blue serif font. Below the title is a breadcrumb trail: 'BRIEFING ROOM' followed by 'PRESIDENTIAL ACTIONS'. A search icon is visible in the top right corner.

- *“Incremental improvements will not give us the security we need”*
- *“The Federal Government must bring to bear the full scope of its authorities and resources to protect and secure its computer systems, whether they are cloud-based, on-premises, or hybrid.”*
- *“All Federal Information Systems should meet or exceed the standards and requirements for cybersecurity set forth in and issued pursuant to this order.”*

**Source:** <https://www.whitehouse.gov/briefing-room/presidential-actions/2021/05/12/executive-order-on-improving-the-nations-cybersecurity/>

# Security Issues in Open Source Software



 Dark Reading

## Open Source Vulnerabilities Still Pose a Big Challenge for Security Teams

Open source software continues to pose a challenge for companies. With the proper security practices, you can reduce your open source risk...

Mar 23, 2023

 The New Stack

## Why So Much Open Source Software Is Vulnerable to Hackers

A recent Open Source Security and Risk Analysis (OSSRA) study indicates that 84% of codebases contained at least one known open source...

Mar 9, 2023

 Lawfare Blog

## Open-Source Security: How Digital Infrastructure Is Built on a House of Cards

Log4Shell remains a national concern because the open-source community cannot continue to shoulder the responsibility of securing this...

Jul 25, 2022

 CSO Online

## The Heartbleed bug: How a flaw in OpenSSL caused a security crisis

Heartbleed can be traced to a single line of code in OpenSSL, an open source code library. Here's how Heartbleed works and how to fix it.

Sep 6, 2022



## How does software security impact Computational Science & Engineering (CSE) / High Performance Computing (HPC) Communities?

- Incautious usage of software systems can create problems for our institutions.
- Software installs can create vulnerabilities on our systems and customer systems.
- Some of our CSE/HPC software may be run in environments that can open up security vulnerabilities.
- Workflows, pipelines, and containers increasingly used for CSE/HPC bring with them possible security vulnerabilities.

**NOTE:** OpenSSL Heartbleed bug was exposed in **2014** and is still generating news articles 8 years later!

# Open Source Security Foundation (OpenSSF)



## Open Source Security Foundation (OpenSSF)

- A Linux Foundation Project
- Cross-industry organization
- Bringing together the industries open source security initiatives & individuals
- Vision: “... a future where participants in the open source ecosystem use and share high quality software, with security handled proactively, by default, and as a matter of course”
- Technical vision:
  - Developers learn secure practices, guided by tools.
  - Security policies created, distributed, enforced automatically.
  - Security issues identified, flow back to chain for rapid response.
  - Community members provide info, notifications, flow forward to users.



OpenSSF Members - Premier (22)  
[[Provide funding for OpenSSF](#)]



Source: <https://openssf.org/about/>

**Questions?**





# OpenSSF Best Practices Badge Program: Overview



- Set of [best practices](#) curated from open-source community that have specific actionable criteria which require supporting evidence,
- Particularly strong focus on [software security](#) which addresses the White House Executive Order 14028 "Improving the Nation's Cybersecurity"<sup>8</sup> (of particular importance to U.S. laboratory and government agencies and contractors),

**NOTE: Most of the OpenSSF best practices are applicable to all software, not just security-critical software!**

- Featureful ["Badge App" site](#) that enhances the display of each practice, expanded descriptions of the practices, and fields to enter URL and text descriptions of the status of each practice in the project,
- [Badge](#) that can be displayed on a project's own hosting site to show that a project follows accepted best practices,
- [Learning tool](#) for best practices for developers and projects,
- [Roadmap for continual improvement](#) for a project as it incrementally adopts more practices and improves its scores in different areas,
- [Standard index](#) into the parts of the projects and how it handles different types of processes, and
- [Website template](#) and database implementation that can be forked and customized for more targeted communities.



# OpenSSF Best Practices Badge Program Overview



# OpenSSF Best Practices Structure



The OpenSSF Best Practices are broken down and organized in several different ways:

- **Required or optional practices:**
  - MUST: Required/not optional (unless 'N/A' is allowed)
  - SHOULD: Required unless a strong argument against can be made
  - SUGGESTED: Not required but suggested
- **Three different badge levels:**
  - Passing: 43 MUST, 10 SHOULD, 14 SUGGESTED
  - Silver: +44 MUST, +10 SHOULD, +1 SUGGESTED
  - Gold: +21 MUST, +2 SHOULD
- **Six different categories in each badge level:**
  - Basic
  - Change Control
  - Reporting
  - Quality
  - **Security**
  - Analysis
- **Each category broken down into subcategories**

## Passing

### Basics

#### Basic project website content

- The project website **MUST** provide information on how to: obtain, provide feedback (as bug reports or enhancements), and contribute to the software. [\[interact\]](#)
- The information on how to contribute **SHOULD** include the requirements for acceptable contributions (...). {**Met URL**} [\[contribution\\_requirements\]](#)

**NOTE:** Each practice has a unique [\[short\\_link\\_name\]](#)

# OpenSSF Best Practices: Passing-Level: Samples 1



## Basics

### Basic project website content

- The project website **MUST** succinctly describe what the software does (what problem does it solve?). [\[description\\_good\]](#)
- The project website **MUST** provide information on how to: obtain, provide feedback (as bug reports or enhancements), and contribute to the software. [\[interact\]](#)
- The information on how to contribute **MUST** explain the contribution process (e.g., are pull requests used?) **{Met URL}** [\[contribution\]](#)
- The information on how to contribute **SHOULD** include the requirements for acceptable contributions (e.g., a reference to any required coding standard). **{Met URL}** [\[contribution\\_requirements\]](#)

### FLOSS license ...

### Documentation ...

### Other ...

## Change Control

### Public version-controlled source repository

- The project **MUST** have a version-controlled source repository that is publicly readable and has a URL. [\[repo\\_public\]](#)
- The project's source repository **MUST** track what changes were made, who made the changes, and when the changes were made. [\[repo\\_track\]](#)
- To enable collaborative review, the project's source repository **MUST** include interim versions for review between releases; it **MUST NOT** include only final releases. [\[repo\\_interim\]](#)
- It is **SUGGESTED** that common distributed version control software be used (e.g., git) for the project's source repository. [\[repo\\_distributed\]](#)

### Unique version numbering ...

### Release notes ...

Source: <https://bestpractices.coreinfrastructure.org/en/criteria>

# OpenSSF Best Practices: Passing-Level: Samples 2



## 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). {Met URL} [\[report\\_process\]](#)
- The project **SHOULD** use an issue tracker for tracking individual issues. [\[report\\_tracker\]](#)
- The project **MUST** acknowledge a majority of bug reports submitted in the last 2-12 months (inclusive); the response need not include a fix. [\[report\\_responses\]](#)
- The project **SHOULD** respond to a majority (>50%) of enhancement requests in the last 2-12 months (inclusive). [\[enhancement\\_responses\]](#)
- The project **MUST** have a publicly available archive for reports and responses for later searching. {Met URL} [\[report\\_archive\]](#)

### Vulnerability report process

- The project **MUST** publish the process for reporting vulnerabilities on the project site. {Met URL} [\[vulnerability\\_report\\_process\]](#)
- ...

## 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. {N/A allowed} [\[build\]](#)
- ...

### Automated test suite

- The project **MUST** use at least one automated test suite that is publicly released as FLOSS <...> [\[test\]](#)

### New functionality testing

- The project **MUST** have a general policy (formal or not) that as major new functionality is added to the software produced by the project, tests of that functionality should be added to an automated test suite. [\[test\\_policy\]](#)
- ...

### Warning flags ...

# OpenSSF Best Practices: Passing-Level: Samples 3



## Security

### Secure development knowledge

- The project MUST have at least one primary developer who knows how to design secure software. (See 'details' for the exact requirements.) [\[know\\_secure\\_design\]](#)
- At least one of the project's primary developers MUST know of common kinds of errors that lead to vulnerabilities in this kind of software, as well as at least one method to counter or mitigate each of them. [\[know\\_common\\_errors\]](#)

### Use basic good cryptographic practices ...

### Secured delivery against man-in-the-middle (MITM) attacks ...

### Publicly known vulnerabilities fixed ...

### Other security issues ...

**NOTE: Most of the OpenSSF best practices are applicable to all software!**

## Analysis

### Static code 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, ... {N/A justification} {Met justification} [\[static\\_analysis\]](#)
- It is SUGGESTED that at least one of the static analysis tools used for the static\_analysis criterion include rules or approaches to look for common vulnerabilities in the analyzed language or environment. {N/A allowed} [\[static\\_analysis\\_common\\_vulnerabilities\]](#)
- All medium and higher severity exploitable vulnerabilities discovered with static code analysis MUST be fixed in a timely way after they are confirmed. {N/A allowed} [\[static\\_analysis\\_fixed\]](#)
- It is SUGGESTED that static source code analysis occur on every commit or at least daily. {N/A allowed} [\[static\\_analysis\\_often\]](#)

### Dynamic code analysis ...

# OpenSSF Best Practices: Criteria Statistics



Table: OpenSSF Best Practice Breakdown

Level	Total active	MUST	SHOULD	SUGGESTED	Security specific	Allow N/A	Met justification or URL required	Require URL
<b>Passing</b>	67	43	10	14	16	27	1	8
<b>Silver</b>	+55	+44	+10	1	+18	40	38	17
<b>Gold</b>	+23	+21	+2	0	+5	9	13	9

## Notes/Observations:

- Some practices are relisted in higher levels going from SUGGESTED to SHOULD or SHOULD to REQUIRED
  - Example **SHOULD ... [bus\_factor]** at Silver-level is relisted as **MUST ... [bus\_factor]** at Gold-level
- **Most of the practices are NOT specific to security!**
- Also, **most of the security-specific practices are "N/A"** or are easily met **for most CSE/HPC software.**



# Questions?

# OpenSSF Best Practices “Badge App” Site





# OpenSSF Best Practices: "Badge App" Site Overview



OpenSSF Best Practices

Projects Sign Up Login

5823 Projects

Badge status: All Exclude passing  Text search: Name or description text

Add New Project

Id	Name	Description	Website	License	Owner	Last achieved at	Tiered %	Badge
1	BadgeApp	BadgeApp is the web application that allows developers to provide information about their project and (hopefully) get an Open Source Security Foundation...	<a href="https://github.com/coreinfrastructure/best-practices-badge">https://github.com/coreinfrastructure/best-practices-badge</a>	MIT	David A. Wheeler	2016-01-12 22:55:00	300%	openssf best practices gold
24	OWASP Zed Attack Proxy (ZAP)	OWASP Zed Attack Proxy (ZAP) is an easy to use integrated penetration testing tool for finding vulnerabilities in web applications. It is designed to be...	<a href="https://www.zaproxy.org">https://www.zaproxy.org</a>	Apache-2.0	Simon Bennetts	2016-08-10 07:03:00	198%	openssf best practices passing
26	TrouSerS	A software stack that provides a programmatic API to the computer's Trusted Platform...	<a href="http://trousers.sourceforge.net">http://trousers.sourceforge.net</a>	CPL-1.0	Charlemange		85%	openssf best practices in progress 85%

## Some Projects Earning Badges:



Source: <https://bestpractices.coreinfrastructure.org/en/projects>

# OpenSSF Best Practices: “Badge App” Site: Gold-Level Projects



OpenSSF Best Practices Projects Sign Up Login

19 Projects

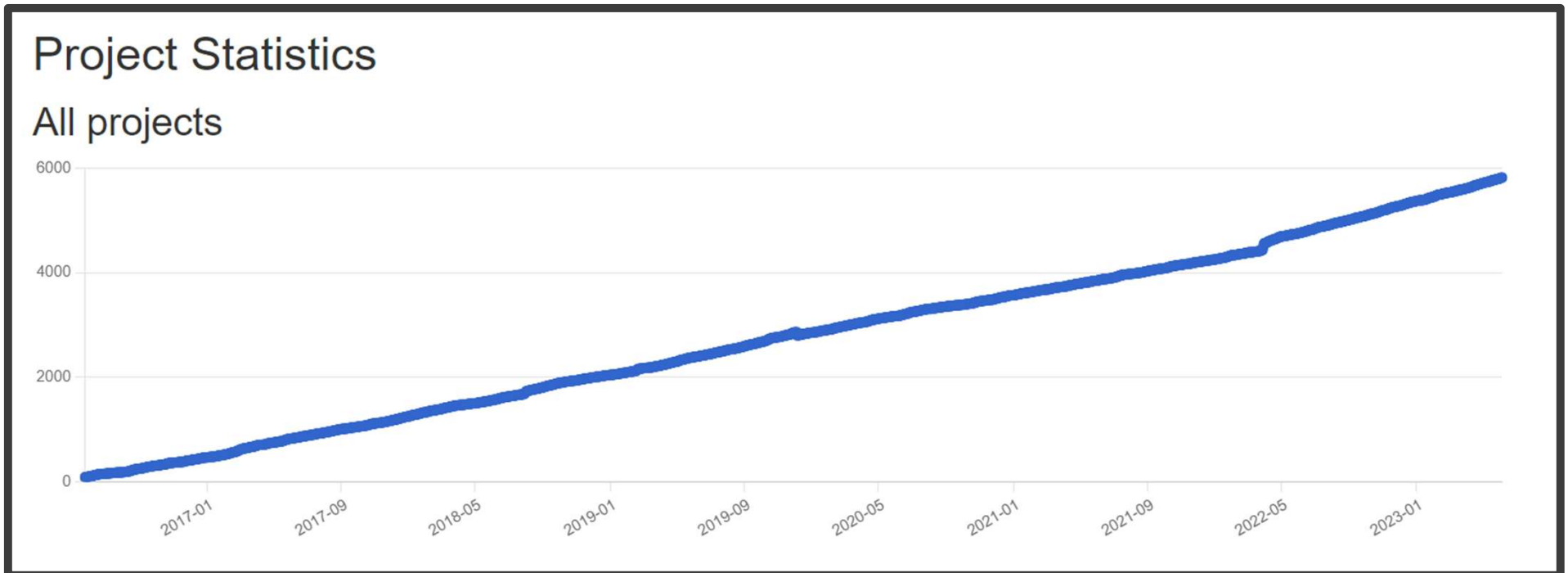
Badge status: Gold (300%)  Exclude passing

Add New Project

Id	Name	Description	Website	License	Owner	Last achieved at	Tiered %	Badge
1	<a href="#">BadgeApp</a>	BadgeApp is the web application that allows developers to provide information about their project and (hopefully) get an Open Source Security Foundation...	<a href="https://github.com/coreinfrastructure/best-practices-badge">https://github.com/coreinfrastructure/best-practices-badge</a>	MIT	David A. Wheeler	2016-01-12 22:55:00	300%	
34	<a href="#">Linux Kernel</a>	The Linux kernel.	<a href="https://www.kernel.org">https://www.kernel.org</a>	GPL-2.0	Greg Kroah-Hartman	2018-06-14 16:10:57	300%	
63	<a href="#">curl</a>	curl is a command line tool and library for internet transfers	<a href="https://curl.se">https://curl.se</a>	MIT	Daniel Stenberg	2016-03-24 20:14:00	300%	
74	<a href="#">Zephyr Project</a>	The Zephyr Project is a small, scalable real-time operating system.	<a href="https://www.zephyrproject.org">https://www.zephyrproject.org</a>	Apache-2.0	Brett Preston	2018-03-10	300%	

Source: <https://bestpractices.coreinfrastructure.org/en/projects?gteq=300>

## OpenSSF Best Practices: Growth in adoption



- Number of registered projects growing steadily
- Number of projects registering is accelerating!
- **Conclusion: Well accepted and adopted badge program and site!**

**Source:** [https://bestpractices.coreinfrastructure.org/en/project\\_stats](https://bestpractices.coreinfrastructure.org/en/project_stats)

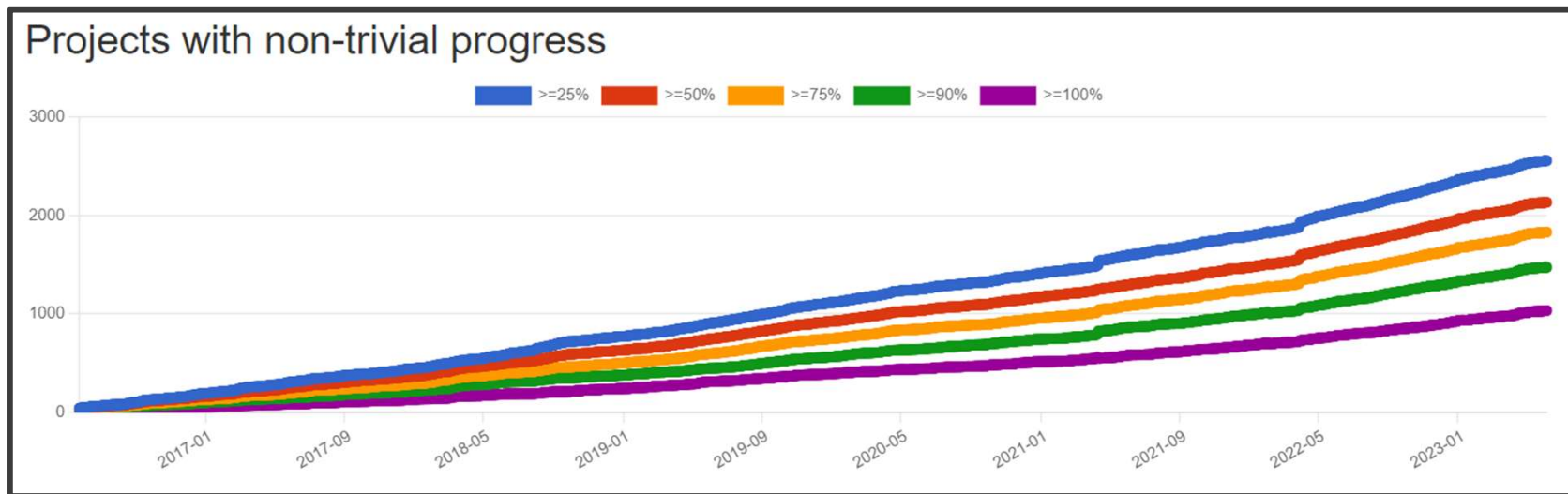
# OpenSSF Best Practices: Badge level and adoption stats

Of 5823 total registered projects (as of 6/5/2023)

- 1033 **Passing-level** projects: **17%**
- 50 **Silver-level** projects: **0.8%**
- 19 **Gold-level** projects: **0.3%**

**SIDENOTE:** A number of the listed projects are not official entries for those projects

**Source:** <https://bestpractices.coreinfrastructure.org/en/projects>



**Source:** [https://bestpractices.coreinfrastructure.org/en/project\\_stats](https://bestpractices.coreinfrastructure.org/en/project_stats)



# OpenSSF Best Practices: Project Page

OpenSSF Best Practices 99% [Projects](#) [Sign Up](#) [Login](#)



## TriBITS Core

[Expand panels](#) [Show all details](#) [Hide met & N/A](#)

Projects that follow the best practices below can voluntarily self-certify and show that they've achieved an Open Source Security Foundation (OpenSSF) best practices badge. [Show details](#)

If this is your project, please show your badge status on your project page! The badge status looks like this: `openssf best practices in progress 99%` Here is how to embed it: [Show details](#)

These are the `passing` level criteria. You can also view the `silver` or `gold` level criteria.

Basics	13/13
Change Control	9/9
Reporting	8/8
Quality	13/13
Security	15/16
Analysis	8/8

Source: <https://bestpractices.coreinfrastructure.org/en/projects/4839#>



# OpenSSF Best Practices: Project Practice Entry



Basics

13/13

Change Control

9/9



## Release notes



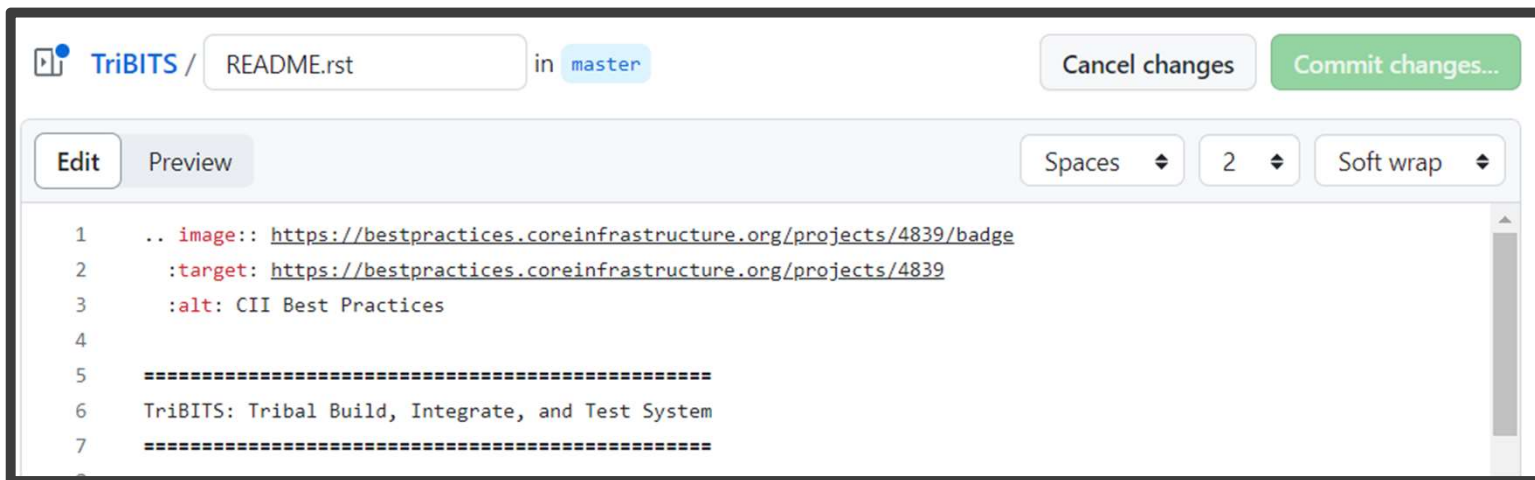
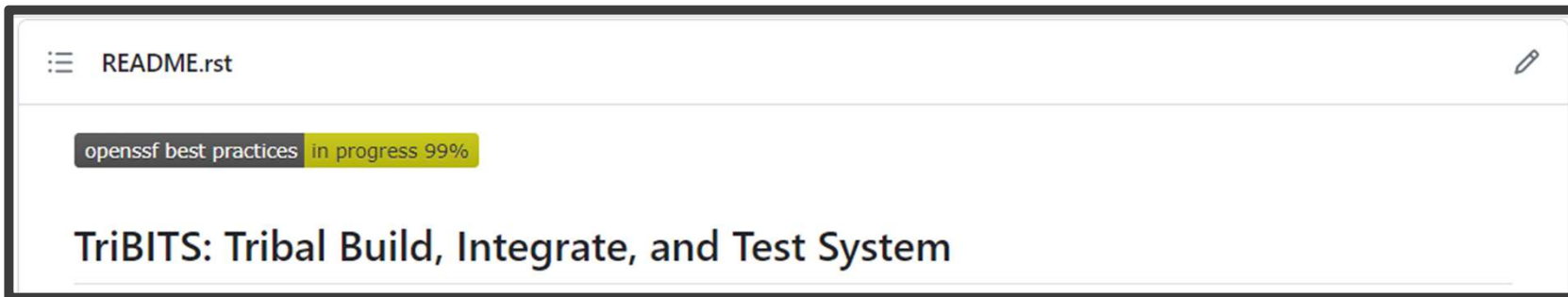
- Met
- Unmet
- N/A
- ?

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) [\[release\\_notes\]](#) [Show details](#)

<https://github.com/TriBITSPub/TriBITS/blob/master/tribits/CHANGELOG.md>

**Source:** <https://bestpractices.coreinfrastructure.org/en/projects/4839#changecontrol>

# OpenSSF Best Practices: Project Badge Display



Source: <https://github.com/TriBITSPub/TriBITS#readme>



**Questions?**





# **Other Benefits of the OpenSSF Best Practices Badge Program**

# OpenSSF Best Practices as a Learning Tool



- Listing of practices includes “additional information”
  - <https://bestpractices.coreinfrastructure.org/en/criteria?details=true&rationale=true>
- Each practice on Badge App Project page has a “Show details” button and there is a “Show all details button” at the top of the page.
  - <https://bestpractices.coreinfrastructure.org/en/projects/4839#all>
- Reading though all 129 best practices with detail (and clicking on the links for more info) can take more than ½ a day (or much longer depending on the level of familiarity with each practices).

# OpenSSF Best Practices: Practices with additional information



## FLOSS license

- The software produced by the project MUST be released as FLOSS. [\[floss license\]](#)

### **Details:**

FLOSS is software released in a way that meets the [Open Source Definition](#) or [Free Software Definition](#). Examples of such licenses include the [CC0](#), [MIT](#), [BSD 2-clause](#), [BSD 3-clause revised](#), [Apache 2.0](#), [Lesser GNU General Public License \(LGPL\)](#), and the [GNU General Public License \(GPL\)](#). For our purposes, this means that the license MUST be:

- [an approved license by the Open Source Initiative \(OSI\)](#), or
- [a free license as approved by the Free Software Foundation \(FSF\)](#), or
- [a free license acceptable to Debian main](#), or
- [a "good" license according to Fedora](#).

The software MAY also be licensed other ways (e.g., "GPLv2 or proprietary" is acceptable).

### **Rationale:**

These criteria are designed for FLOSS projects, so we need to ensure that they're only used where they apply. Some projects may be mistakenly considered FLOSS even though they are not (e.g., they might not have any license, in which case the defaults of the country's legal system apply, or they might use a non-FLOSS license). We've added "produced by the project" as a clarification - many projects use non-FLOSS software/services in the process of creating software, or depend on them to run, and that is allowed.

**Source:** <https://bestpractices.coreinfrastructure.org/en/criteria?details=true&rationale=true>

# OpenSSF Best Practices: Badge App Project Page “Show Details”



## FLOSS license

What license(s) is the project released under? [Show details](#)

BSD-3-Clause



- Met
- Unmet
- ?

The software produced by the project MUST be released as FLOSS. [\[floss\\_license\] Hide details](#)

FLOSS is software released in a way that meets the [Open Source Definition](#) or [Free Software Definition](#). Examples of such licenses include the [CC0](#), [MIT](#), [BSD 2-clause](#), [BSD 3-clause revised](#), [Apache 2.0](#), [Lesser GNU General Public License \(LGPL\)](#), and the [GNU General Public License \(GPL\)](#). For our purposes, this means that the license MUST be:

- an approved license by the [Open Source Initiative \(OSI\)](#), or
- a free license as approved by the [Free Software Foundation \(FSF\)](#), or
- a free license acceptable to [Debian main](#), or
- a "good" license according to [Fedora](#).

The software MAY also be licensed other ways (e.g., "GPLv2 or proprietary" is acceptable).

<https://github.com/TriBITSPub/TriBITS/blob/master/tribits/Copyright.txt> The BSD-3-Clause license is approved by the Open Source Initiative (OSI).

**Source:** <https://bestpractices.coreinfrastructure.org/en/projects/4839#basics>

# OpenSSF Best Practices as Continuous Process Improvement tool



- Badge App Project page **Show unmet criteria mode** ( i.e. “Expand panels” and “Hide met & N/A”) :
  - <https://bestpractices.coreinfrastructure.org/en/projects/4839#all>

**TriBITS Core**

Expand panels Show all details Hide met & N/A

Projects that follow the best practices below can voluntarily self-certify and show that they've achieved an Open Source Security Foundation (OpenSSF) best practices badge. [Show details](#)

If this is your project, please show your badge status on your project page! The badge status looks like this: `openssf best practices in progress 99%` Here is how to embed it: [Show details](#)

These are the `passing` level criteria. You can also view the `silver` or `gold` level criteria.

- Badge App **sends out periodic email reminders** about the status for your projects and where to look for improvements.

# OpenSSF Best Practices Project Page: Showing unmet criteria



OpenSSF Best Practices 99% [Projects](#) [Sign Up](#) [Login](#)



## TriBITS Core

[Collapse panels](#) [Show all details](#) [Show met & N/A](#)

Projects that follow the best practices below can voluntarily self-certify and show that they've achieved an Open Source Security Foundation (OpenSSF) best practices badge. [Show details](#)

If this is your project, please show your badge status on your project page! The badge status looks like this: `openssf best practices in progress 99%` Here is how to embed it: [Show details](#)

These are the `passing` level criteria. You can also view the `silver` or `gold` level criteria.



Change Control 9/9

### Public version-controlled source repository

### Unique version numbering

Met  Unmet  ?

It is SUGGESTED that the [Semantic Versioning \(SemVer\)](#) or [Calendar Versioning \(CalVer\)](#) version numbering format be used for releases. It is SUGGESTED that those who use CalVer include a micro level value. `[version_semver]` [Show details](#)

TriBITS has not put out any official releases. Customers instead use almost-continuous integration to get updates of TriBITS. In the future, official releases may be put out.

Source: <https://bestpractices.coreinfrastructure.org/en/projects/4839#all>



# OpenSSF Best Practices Project Page: Showing unmet criteria



Quality 13/13

Working build system


Automated test suite

New functionality testing

Warning flags

Security 15/16

Secure development knowledge

  Met  Unmet  ?

The project MUST have at least one primary developer who knows how to design secure software. (See 'details' for the exact requirements.) [know\_secure\_design] [Show details](#)

The main developer does not claim to "know how to design secure software" so I can't check this. But it is hard to see how a framework for CMake build systems could open up these types of security vulnerabilities.

# OpenSSF Best Practices Badge App: Email Reminders



**From:** badgeapp@bestpractices.coreinfrastructure.org <badgeapp@bestpractices.coreinfrastructure.org>

**Sent:** Wednesday, April 26, 2023 7:02 PM

**To:** Bartlett, Roscoe A <rabartl@sandia.gov>

**Subject:** [EXTERNAL] Your project does not yet have the "best practices" badge

## Automated Best Practices Badge Reminder for TriBITS Core

This is an automated reminder that your project "TriBITS Core" does not currently have a "best practices" badge, and its badge entry has not been updated in a while.

Your best practices badge entry is at <https://bestpractices.coreinfrastructure.org/en/projects/4839> and was last updated on 2022-05-26 16:11:01 UTC. It is currently at 99% (out of 100%).

We encourage you to keep making progress. Please visit your badge entry at <https://bestpractices.coreinfrastructure.org/en/projects/4839> to complete the information and get your badge!

If you want to see *only* what you're missing, visit your badge entry, select the button near the top labelled "Expand all panels", and then select the button near the top labelled "Hide met or N/A criteria". If you have questions, or need help, please contact [cii-badges-questions-owner@lists.coreinfrastructure.org](mailto:cii-badges-questions-owner@lists.coreinfrastructure.org) or file an issue at <https://github.com/coreinfrastructure/best-practices-badge/issues> (click on "new issue").

We don't send reminders if you continue to update your badge entry, and we only send reminders approximately every 30-60 days. However, if you want to disable these reminder messages, edit your badge entry at [https://bestpractices.coreinfrastructure.org/en/projects/4839#project\\_disabled\\_reminders](https://bestpractices.coreinfrastructure.org/en/projects/4839#project_disabled_reminders) to turn on "disable inactivity reminder". Be sure to edit the project entry, don't just display it, if you want to change whether or not you'll receive an inactivity reminder. We hope you'll instead keep working at it and eventually earn the badge.

Thank you for your time.


--- David A. Wheeler, OpenSSF Best Practices Badge Lead

# OpenSSF Best Practices Badge App: Standard Index into projects



- Badge App Project Page for a given project provides a standard list of practices and make it easy to find how a given project addresses various issues and how to access those.
- Example: **How to report an issue for the project?**
  - [[report\\_process](#)] Passing-level, Reporting, Bug reporting process

### Bug-reporting process


 Met      The project MUST provide a process for users to submit bug reports (e.g., using an issue tracker or a mailing list). (URL required) <sup>[report\_process]</sup>

Unmet

?

Yes, either GitHub issue tracker or mailing list. See: <https://github.com/linuxfoundation/cii-best-practices-badge/blob/master/README.md>

- Example: **Documentation of the internal implementation?**
  - [[documentation\\_architecture](#)] Silver-level, Basics, Documentation

 Met      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) <sup>[documentation\_architecture]</sup> [Show details](#)

Unmet

N/A

?

The design is documented in <doc/implementation.md>

**Questions?**





# Software Security Practices

# OpenSSF Best Practices: OpenSSL (Heartbleed bug)



Id	Name	Description	Website	License	Owner	Last achieved at	Tiered %	Badge
54	OpenSSL	OpenSSL is an open source project that provides a robust, commercial-grade, and full-featured toolkit for the Transport Layer Security (TLS) and Secure...	<a href="https://www.openssl.org">https://www.openssl.org</a>	OpenSSL	Rich Salz	2016-02-12 22:09:00	105%	
87	OpenSSL before Heartbleed	This is a <i>historical</i> badge entry for the OpenSSL project before the Heartbleed vulnerability was reported, circa February 2014. Please note that the...	<a href="http://www.openssl.org">http://www.openssl.org</a>	OpenSSL	Rich Salz		63%	

**Source:** <https://bestpractices.coreinfrastructure.org/en/projects?q=openssl>

**NOTE:** OpenSSL Heartbleed bug was exposed and fixed in **2014** and was **only 63% of a passing** OpenSSF best practices badge!

**NOTE:** Even today, OpenSSL only achieves a passing badge!

**Source:** <https://en.wikipedia.org/wiki/Heartbleed>

# OpenSSF Best Practices: Security Focus?



## Software Security Practices:

- 16 of 67 **Passing-level** practices (9 of 16 allow N/A)
- +18 of +55 **Silver-level** practices (11 of 18 allow N/A)
- +5 of +23 **Gold-level** practices (3 of 5 allow N/A)

## Until recently, software security is barely mentioned in most software engineering books:

**Classic Example:** "Code Complete: 2nd Edition", 800+ pages, 2004: **Exactly one paragraph** is devoted to the area of software security in section 3.5 "Architecture Prerequisite":

*The architecture should describe the approach to design level and code level security. If a thread model has not previously been built, it should be built at architecture time. Coding guidelines should be developed with security implications in mind, including approaches to handling buffers, rules for handling untrusted data (data input from users, cookies, configuration data, and other external interfaces), encryption, level of detail contained in error messages, protected secret data that's in memory, and other issues.*

**More Recent Example:** "The Pragmatic Programmer: 20th Anniversary Edition", 300+ pages, 2020: **Devotes 7 pages** to software security in Topic 42 "Stay Safe Out There".

- Awareness and visibility of software security is increasing
- Is this good timing for discussing security? => **White House Executive Order 14028**



# OpenSSF Best Practices: The Most Difficult Passing-level Items?



## Passing-level

### Security

#### Secure development knowledge

- The project MUST have at least one primary developer who knows how to design secure software. (See 'details' for the exact requirements.) [\[know\\_secure\\_design\]](#)
- At least one of the project's primary developers MUST know of common kinds of errors that lead to vulnerabilities in this kind of software, as well as at least one method to counter or mitigate each of them. [\[know\\_common\\_errors\]](#)

---

What does it mean to **“knows how to design secure software”** and **“know of common kinds of errors that lead to vulnerabilities in this kind of software”**?

**NOTE:** These are not even practices!

=> These are asking the question **“Do you know what you are doing?”**

# OpenSSF Best Practices: Know secure design?



## Security

### Secure development knowledge

- The project MUST have at least one primary developer who knows how to design secure software. (See 'details' for the exact requirements.) [\[know\\_secure\\_design\]](#)

#### **Details:**

This requires understanding the following design principles, including the 8 principles from [Saltzer and Schroeder](#):

- economy of mechanism (<...>), fail-safe defaults (<...>), complete mediation (<...>), open design (<...>), separation of privilege (<...>), least privilege (<...>), least common mechanism (<...>), psychological acceptability (<...>), limited attack surface (<...>), input validation with allowlists (<...>)

A "primary developer" in a project is anyone who is familiar with the project's code base, <...> If there is only one developer, that individual is the primary developer. Many books and courses are available to help you understand how to develop more secure software and discuss design. For example, the [Secure Software Development Fundamentals](#) course is a free set of three courses that explain how to develop more secure software (it's free if you audit it; for an extra fee you can earn a certificate to prove you learned the material).

---

### What is needed to check this box? [OpenSSF: Concise Guide for Developing More Secure Software](#):

- **Learn about secure software development.** Take, e.g., the [free OpenSSF course](#) or the hands-on [Security Knowledge Framework](#) course. [SAFECode's Fundamental Practices for Secure Software Development](#) provides a helpful summary.
  - [Free OpenSSF course](#) : 18 hours every 2 years (least time consuming option?)

# OpenSSF Best Practices: Know common kinds of errors?



## Security

### Secure development knowledge

- At least one of the project's primary developers **MUST** know of common kinds of errors that lead to vulnerabilities in this kind of software, as well as at least one method to counter or mitigate each of them. [\[know common errors\]](#)

#### *Details:*

- Examples (depending on the type of software) include SQL injection, OS injection, **classic buffer overflow**, cross-site scripting, missing authentication, and missing authorization. See the [CWE/SANS top 25](#) or [OWASP Top 10](#) for commonly used lists. Many books and courses are available to help you understand how to develop more secure software and discuss common implementation errors that lead to vulnerabilities. For example, the [Secure Software Development Fundamentals](#) course is a free set of three courses that explain how to develop more secure software (it's free if you audit it; for an extra fee you can earn a certificate to prove you learned the material).

# Common Security Vulnerabilities? [2022 CWE/SANS top 25](#)



## Security vulnerabilities related to CSE/HPC Software (9 of 25)

Rank	ID	Name	Score	KEV Count (CVEs)	Rank Change vs. 2021
1	<a href="#">CWE-787</a>	Out-of-bounds Write	64.2	62	0
4	<a href="#">CWE-20</a>	Improper Input Validation	20.63	20	0
5	<a href="#">CWE-125</a>	Out-of-bounds Read	17.67	1	-2
7	<a href="#">CWE-416</a>	Use After Free	15.5	28	0
11	<a href="#">CWE-476</a>	NULL Pointer Dereference	7.15	0	4
12	<a href="#">CWE-502</a>	Deserialization of Untrusted Data	6.68	7	1
13	<a href="#">CWE-190</a>	Integer Overflow or Wraparound	6.53	2	-1
23	<a href="#">CWE-400</a>	Uncontrolled Resource Consumption	3.56	2	4
25	<a href="#">CWE-94</a>	Improper Control of Generation of Code ('Code Injection')	3.32	4	3

- 5 of the top 11 are errors can occur an very common in all CSE/HPC codes
- Many of these are major causes of software bugs impacting software correctness in CSE/HPC codes

**Conclusion: Software correctness & quality and software security are best friends** 🤝 😊

**Source:** [https://cwe.mitre.org/top25/archive/2022/2022\\_cwe\\_top25.html](https://cwe.mitre.org/top25/archive/2022/2022_cwe_top25.html)

## OpenSSF Best Practices Badge Program: Summary



- OpenSSF Best Practices provides actionable collection of open source communities best practices.
- OpenSSF Best Practices Badge App site codifies best practices and provides blanks for projects to fill in.
- Badge can be displayed on project's hosting site when a given badge level is earned (or showing progress towards a badge).
- Badge App site contains features to identify areas of improvement.
- Badge App site can send out regular reminders to keep making progress.
- Badge App site has the potential to provide a ready implementation foundation for a customized best practices site.
- OpenSSF Best Practices Badge Program elevates security to a first-level concern for everyone developing software? **(What the minimum sufficient level of knowledge?)**

**Suggestion: Create an OpenSSF Best Practices Badge App site entry for one of your software projects?**

<https://bestpractices.coreinfrastructure.org/en/projects>



# Comments or Questions?