THE OPENSSF SUPER BEST FRIENDS

AMAZING tales of ADVENTURE in SECURITY!!!

ACTION-PACKED **FIRST** ISSUE!!

TABLE OF CONTENTS

01 WHO IS THIS GUY?

CRob, n, adj, and v

Pronunciation: U.S. (K-rowb)

Chief Security Architect, OpenSSF- Linux Foundation

43rd level Dungeon Master, 26th level Securityologist

Pirate-enthusiast, hat-owner, & penguin fancier

02

OPENSSF

The Open Source Security Foundation

http://www.quickmeme.com/meme/36exe7

EXCELCIOR!

We are assembling a team of experienced security-ologists to share the wisdom and insights they've earned over the ‹cough cough› years they have been practitioners within upstream open source communities

WHAT IS THE OPENSSF?

- The OpenSSF is a cross-industry collaboration that brings together leaders to improve the security of open source software (OSS) by building a broader community, targeted initiatives, and best practices

- The OpenSSF brings together open source security initiatives under one foundation to accelerate work through cross-industry support. This is beginning with the Core Infrastructure Initiative and the Open Source Security Coalition, and will include new working groups that address vulnerability disclosures, security tooling and more.

- OpenSSF is committed to collaboration and working both upstream and with existing communities to advance open source security for all.

OpenSSF

OPEN SOURCE SECURITY FOUNDATION

[What's up with all the Geese?]

OpenSSF
OPEN SOURCE SECURITY FOUNDATION

WORKING GROUPS, PROJECTS, & SIGS

1. Inform

Vulnerability Disclosures
Efficient vulnerability reporting and remediation

- V1. CVD Guides SIGs
- V2. Open Source Vuln Schema (OSV) project
- V3. OpenVEX project
  - OpenVEX SIG

Securing Critical Projects
Identification of critical open source projects

- CP1. criticality_score project
- CP2. Package Analysis project

DevRel
Develop Use Cases and help others learn about security

AI/ML Security
AI/ML Security at the Intersection of Artificial Intelligence and Cybersecurity

- A1. Model Signing SIG

2. Equip

Best Practices
Identification, awareness, and education of security best practices

- B1. OpenSSF Best Practices Badge project
- B2. OpenSSF Scorecard project
- B3. Education SIG
- B4. Memory Safety SIG
- B5. C/C++ Compiler Options SIG
- B6. Python Hardening SIG
- B7. Security Baseline SIG

Security Tooling
State of the art security tools

- T1. SBOM Everywhere SIG
- T2. OSS Fuzzing SIG
- T3. SBOMit project
- T4. Protobom project
- T5. bomctl project
- T6. Fuzz Introspector project
- T7. Minder project

Diversity, Equity, & Inclusion
Increase representation and strengthen the overall effectiveness of the cybersecurity workforce

3. Engage

Securing Software Repositories
collaboration between repository operators

- R1. RSTUF Project

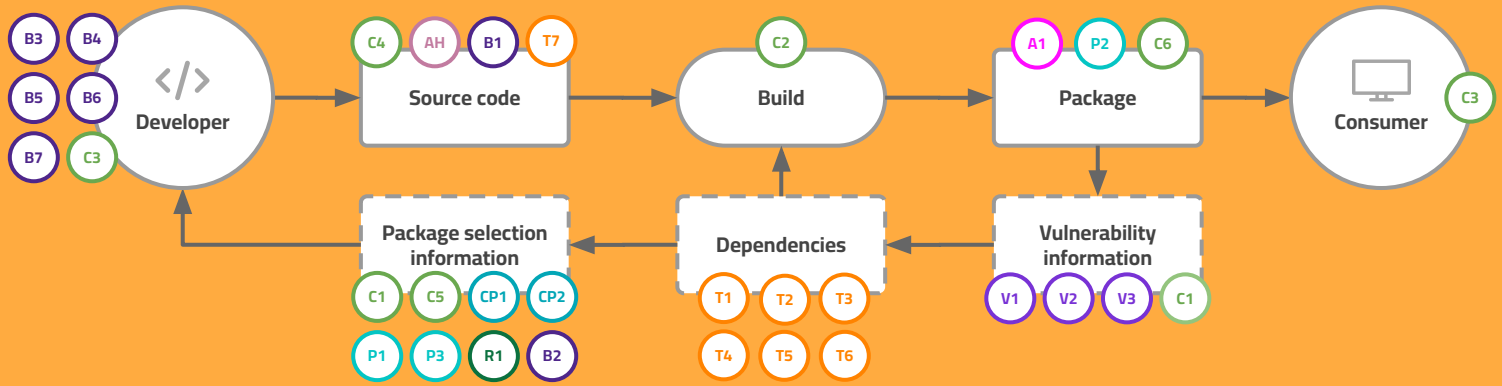
Supply Chain Integrity
Ensuring the provenance of open source code

- C1. Security Insights project
- C2. Supply-chain Levels for Software Artifacts (SLSA) project
- C3. Secure Supply Chain Consumpt Framework (S2C2F) project
- C4. gittuf project
- C5. GUAC project
- C6. Zarf project

Projects
Category-leading software initiatives

- P1. Alpha Omega
- P2. Sigstore
- P3. Core Toolchain Infrastructure (CTI)

OPENSSF TECHNICAL INITIATIVES LANDSCAPE



Best Practices
- B1. OpenSSF Best Practices Badge project
- B2. OpenSSF Scorecard project
- B3. Education SIG
- B4. Memory Safety SIG
- B5. C/C++ Compiler Options SIG
- B6. Python Hardening SIG
- B7. Security Baseline SIG

Supply Chain Integrity
- C1. Security Insights project
- C2. SLSA project
- C3. S2C2F project
- C4. Gittuf project
- C5. GUAC project
- C6. Zarf project M

Security Tooling
- T1. SBOM Everywhere SIG
- T2. OSS Fuzzing SIG
- T3. SBOMit project
- T4. Protobom project
- T5. bomctl project
- T6. Fuzz Introspector project
- T7. Minder project

Securing Critical Projects
- CP1. criticality_score project
- CP2. Package Analysis project

Projects
- P1. Alpha & Omega project
- P2. Sigstore
- P3. Core Toolchain Infrastructure (CTI)

DevRel Community

Diversity, Equity, & Inclusion

Vulnerability Disclosures
- V1. CVD Guides SIGs
- V2. OSV Schema project
- V3. OpenVEX SIG

OpenVEX Project

AI/ML Security
- A1. Model Signing SIG

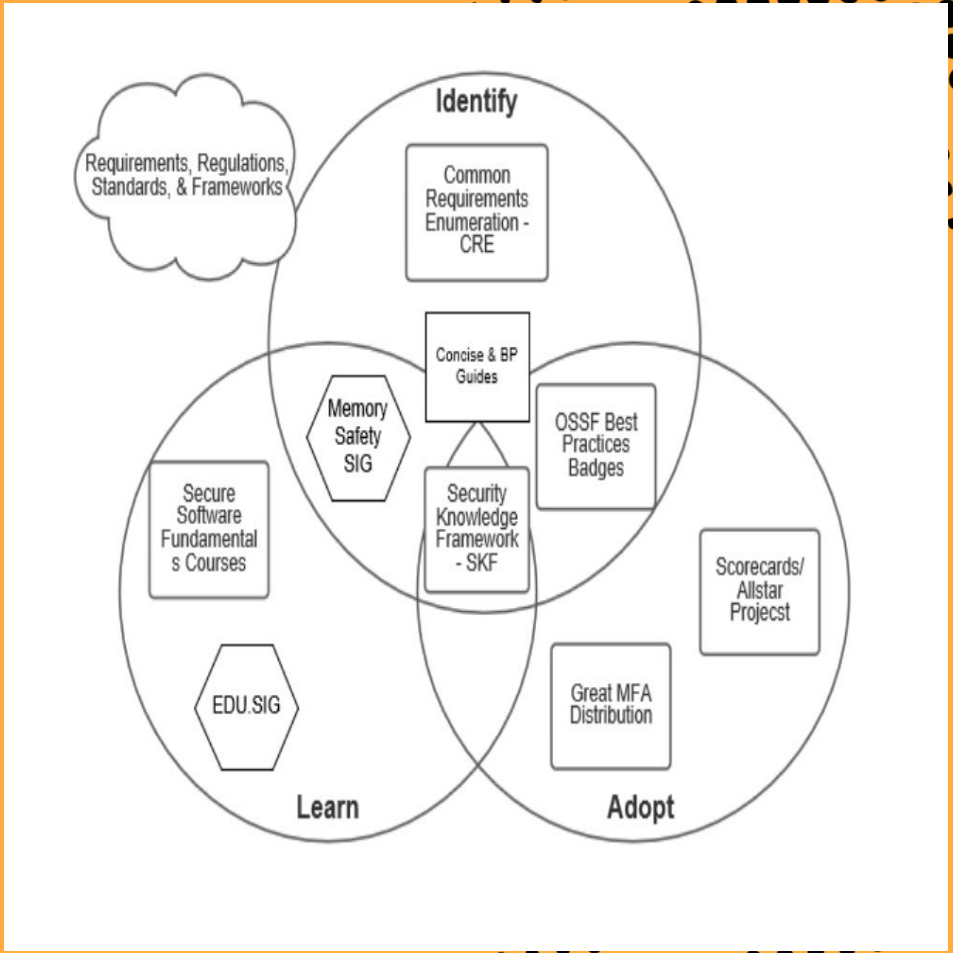
Securing Software Repositories
- R1. RSTUF Project

OPENSSF DEVELOPER BEST PRACTICES WORKING GROUP

Our objective is to provide open source developers with best practices recommendations, and with an easy way to learn and apply them.

Our vision is to make it easy for developers to adopt these best practices, thanks to:

- **Identifying** good practices, requirements, and tools that help open source developers create and maintain more secure software
- Helping maintainers **Learn** to write secure software
- Provide tools to help developers **Adopt** these good practices into their daily work



BEST EFFORTS

SECURITY FUNDATMENTALS CLASS

Free course teaching AppSec and Dev basics

BEST PRACTICES BADGES

Interview-based badging proof of good practices

EDUCATION & MEMORY SAFETY SIGS

Groups focused on BEST practices & training

SCORECARD

Automated review & scoring of project good practices

CONCISE GUIDES

Checklists & Guides to quickly learn how to follow BEST practices. Includes C/C++ Complier hardening, SCM, and Python security guides!

SECURITY BASELINE

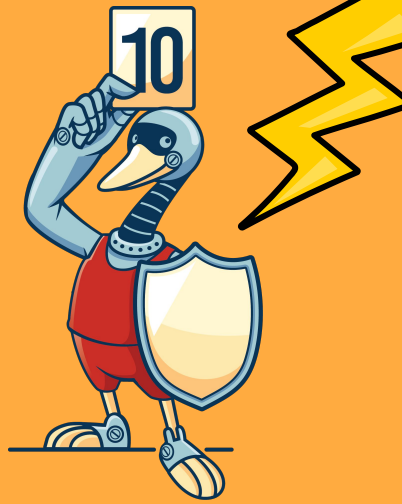
Criteria projects should follow to align with common industry frameworks

OPENSSF BEST PRACTICES BADGE

- Identifies best practices for OSS projects
  - Goal: Increase likelihood of better quality & security. E.g.:
    - "The project sites… MUST support HTTPS using TLS."
    - "The project MUST use at least one automated test suite…"
    - "At least one static code analysis tool MUST be applied…"
    - "The project MUST publish the process for reporting vulnerabilities on the project site."
  - Based on practices of well-run OSS projects
- If OSS project meets best practice criteria, it earns a badge
  - Enables projects & potential users know current status & where it can improve
  - Combination of self-certification, automated checks, spot checks, public accountability
  - Three badge levels: passing, silver, gold
- Participation widespread & continuing to grow
  - >5,000 participating projects, >850 passing+ projects in 2022-10
  - Current statistics: https://bestpractices.coreinfrastructure.org/en/project_stats
- Supports many languages, including English, Simplified Chinese, & Japanese
- For more, see: https://bestpractices.coreinfrastructure.org

OPENSSF SCORECARD

Scorecard takes a different approach, leaning into automation and externally-observable facts it can determine, as opposed to the interview-style approach the **Best Practices Badges** takes.



OPENSSF SCORECARD

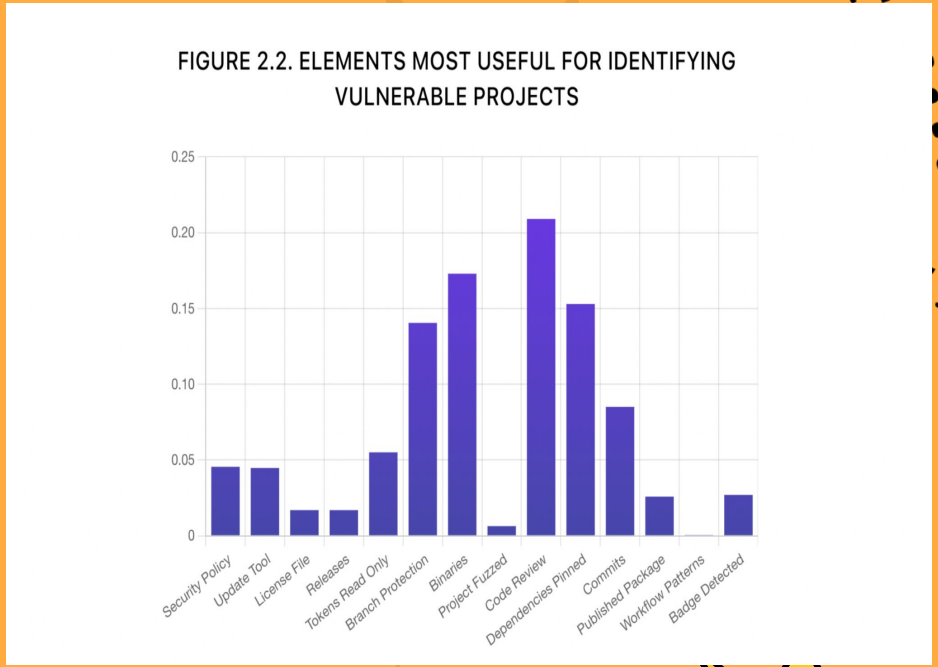
- *Automatically* scores OSS projects on security heuristics ("checks")
  - Each related to security, scored 0-10, weighted average computed
  - Can use to evaluate your own or others' projects (they don't need to cooperate)
  - Currently only works on projects hosted on GitHub (not fundamental)
- Gives:
  - Aggregate (weighted) score
  - Scores for different areas
  - Ideas for improvement
- Newer features: scorecards badge, REST API
- Officially it's ("OpenSSF Scorecard") but we often use the plural
- https://github.com/ossf/scorecard

SONATYPE ANALYSIS OF SCORECARD HEURISTICS

Sonatype's eighth annual "***State of the Software Supply Chain Report***" examined Scorecard

Found some Scorecard heuristics especially helpful at predicting projects with vulnerabilities:

1. Code Review (lack of) - peer review is a *powerful* against vulnerabilities
2. Checked in Binaries - an attack path, decrease transparency, and reduce auditability
3. Pinning dependencies - forces correct ones
4. Branch protection - enables review process



https://www.sonatype.com/state-of-the-software-supply-chain/project-quality-metrics
https://openssf.org/blog/2022/10/20/report-finds-openssf-scorecards-are-highly-effective-measures-to-assess-project-security/

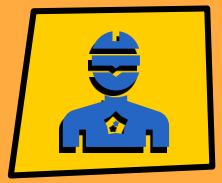
03

ALL YOUR BASE

Are belong to us.

Image Source

WHAT IS A BASELINE, AND WHY DOES IT MATTER TO ME?



DEFINITION

Often seen in enterprise operations, a Baseline is defined by NIST SP 800-37 Rev. 2 as

"The set of controls that are applicable to information or an information system to meet legal, regulatory, or policy requirements, as well as address protection needs for the purpose of managing risk."

TL/DR

"A set of security requirements that help an organization or project meet specific security objectives".

Enter the Open Source Project Security (OSPS) Baseline

<insert dramatic music>

STANDARDS + FRAMEWORKS = BASED CRITERIA

ACCESS CONTROL

How is access and changes to the code and pipeline managed?

DOCUMENTATION

What does the project state about how they manage the code and react to vulnerability reports?

LEGAL

What licenses are in place for the project?

BUILD + RELEASE

How are the CI/CD pipelines configured and managed?

QUALITY

What testing and change management are in place?

ACCESS CONTROL

How is access and changes to the code and pipeline managed?

Phase I

- MFA
- Least Privilege Access
- Push prevention
- Branch protection

Phase II

- Least privilege on CI/CD
- Code reviewers reviewed

Phase III

- MFA excluding SMS

BUILD + RELEASE

How are the CI/CD pipelines configured and managed?

Phase I

- No arbitrary code execution
- Tagged releases
- Encryption in transit/at rest

Phase II

- Automated builds
- Standard dependency tooling

Phase III

- Address SCA findings

DOCUMENTATION

What does the project state about how they manage the code and react to vulnerability reports?

Phase I

- Documented change process
- Documented contributor ladder
- Basic documentation

Phase II

- Documented vulnerability reporting
- Contributor guide
- Design documentation

Phase III

- Documented vulnerability disclosure

QUALITY

What testing and change management are in place?

Phase I

- Source code available
- Version Control System

Phase II

- SBOM or other dependency list
- Status checks must block
- All project reports align to baseline
- No executables in VCS

LEGAL

What licenses are in place for the project?

Phase I

- DCO (contribution assertion)
- Standard OSS license
- Discoverable license
- License in released assets

BASELINE IS GROUNDED IN INTERNATIONAL SECURITY FRAMEWORKS

So "BASELINING" helps you be compliant!

GLOBAL FRAMEWORKS

ISO 27001
ISO 27002

International InfoSec frameworks

EU CRA + PLD

The Cyber Resilience Ace and Product Liability Directive

NIST SSDF 1.1

NIST's Secure Software Development Framework

CSA CCM V4

Cloud Security Alliance's Cloud Controls Matrix

NIST CSF 2.0

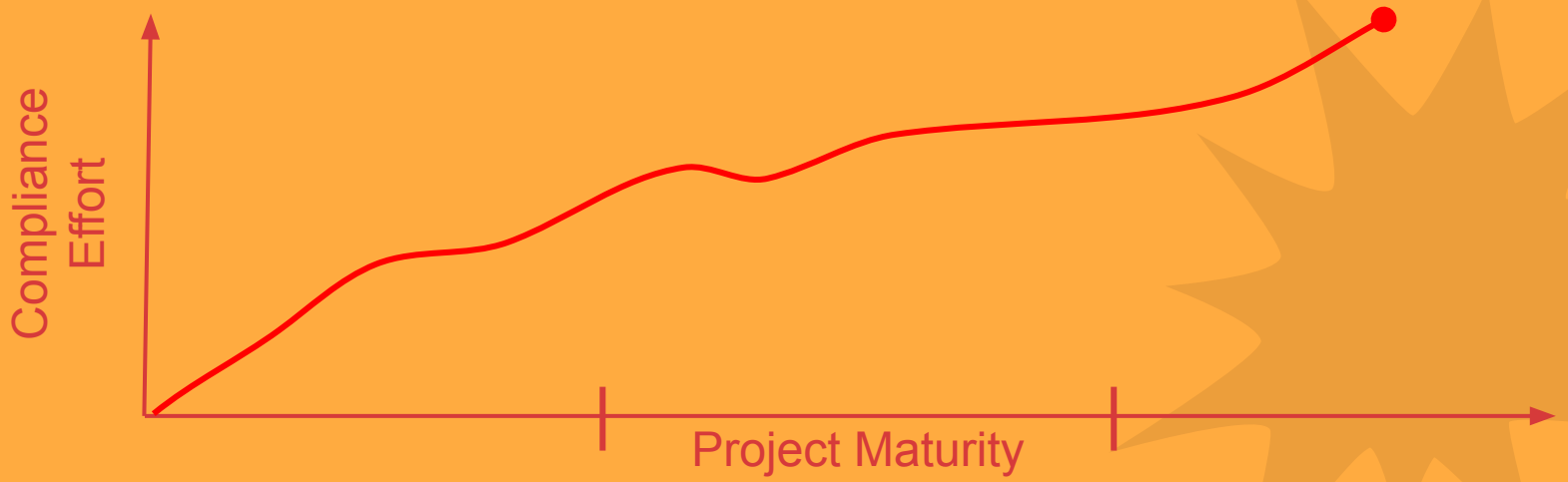
NIST's Cyber Security Framework

...AND MANY OTHERS!

COMPLIANCE CROSSWALK (FUTURE DELIVERABLE)

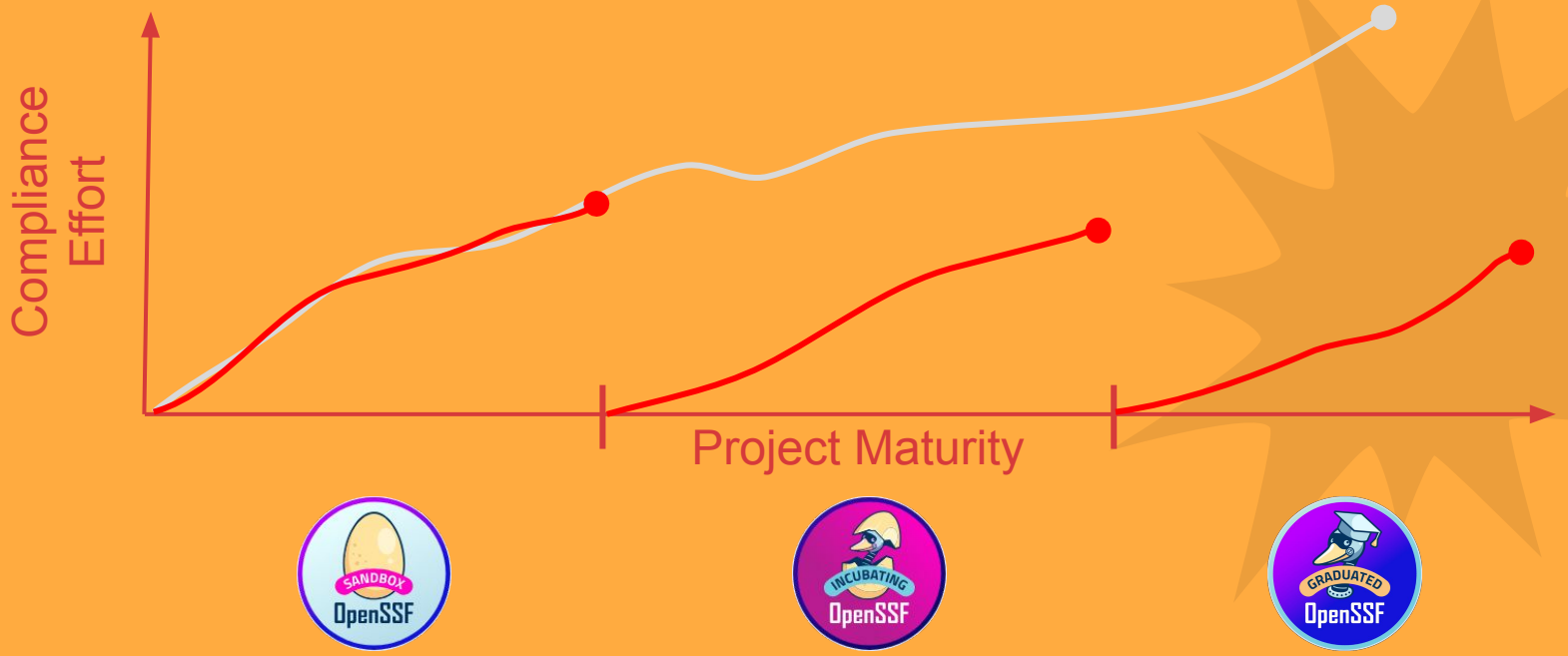
	WIP - Best Practices: Future						OpenSSF Mappings	CRA+ PLD	SSDF 1.1
	Use present tense: State what is actively enforced or applied now. Allows for T/F result. **Start with the item that should be evaluated**: Begin each statement with the subject (e.g., "Permissions" or "All software releases"). **Actionable verbs**: Use clear, action-oriented verbs like "require," "restrict," "ensure." **Name a specific control activity**: Include relevant details to clarify the action being taken. **Consistent terminology**: Use terms uniformly to avoid ambiguity. Example Format: Subject + Action + Control Activity (e.g., "Permissions are scoped to reduce access to only those necessary for performing operations.")		**Explain the purpose**: Describe why the check is valuable or necessary. **Highlight risk mitigation**: Identify the risks addressed by the check. **Reference standards**: Mention relevant regulations or industry best practices. **Discuss consequences**: Explain what could happen if the check isn't followed. **Be concise**: Keep the explanation clear and to the point. **Emphasize prevention**: Focus on how the check prevents issues.		**Level 1** should contain criteria that would benefit any code or non-code project with any number of maintainers or users. (this definition is WIP) **Level 2** should contain criteria that would benefit any code project that has at least 2 maintainers and a small number of consistent users. (this definition is WIP) **Level 3** should contain criteria for any code project that has a large number of consistent users.		this column has all of the "passing" BP criteria	This column has all of the CRA, CRA Annex, and PLD requirements	This column has the NIST SSDF (800-218) requirements
ID	**Category**	**Revised Criteria Statement**	**Maturity Level**	**Notes**	**Security Insights Value**	**Automated Check Description**	**BP Badges Text**		
OSPS-01	Access Control	The project's **version control system** MUST require multi-factor authentication (MFA) for users when modifying the project repositories or accessing sensitive data.	1 (sandbox)	Multi-factor authentication (MFA) significantly reduces the risk of unauthorized access by adding an extra layer of security beyond passwords. By requiring MFA for modifications to project repositories or access to sensitive data, the organization mitigates the threat of credential theft and ensures compliance with security best practices and industry standards. This proactive approach helps prevent security breaches and protects critical assets.			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. The project MUST require two-factor authentication (2FA) for developers for changing a central repository or accessing sensitive data (such as private vulnerability reports).	1.2d, 1.2e, 1.2f	PO3.2, PS-1
OSPS-02	Access Control	The project's **version control system** MUST require multi-factor authentication (MFA) that does not include SMS for users when modifying the project repositories or accessing sensitive data.	3 (graduated)					1.2f	PO-2, PO-3.2, PS-1
OSPS-42	Build & Release	All **released software assets** MUST be created with consistent, automated build and release processes.	2 (incubating)	Consistent and automated build and release processes ensure that each software release is produced in a reliable, repeatable manner. Automation reduces human error, enhances efficiency, and ensures that security checks are consistently applied. This practice helps prevent vulnerabilities introduced through manual processes and aligns with industry best practices for secure and efficient software delivery.	project-lifecycle: release-process (needs improvement?)	no	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.)	1.2b, 1.2d, 1.2f, 1.2h, 1.2j	PO3.2, PS-1
OSPS-05	Build & Release	The project's **build and release pipelines** MUST NOT execute arbitrary code that is input from outside of the build script.	1 (sandbox)	Preventing the execution of arbitrary code in software release automation protects the integrity of the build process and prevents malicious code from being injected into the release pipeline. This safeguard mitigates the risk of supply chain attacks and ensures that only trusted, vetted code is executed. It aligns with security best practices and reduces the likelihood of compromised releases due to unauthorized code execution.		maybe: Dangerous-Workflow		1.2f	PO-3.2, PS-1

OSPS BASELINE - MATURING WITH YOUR PROJECT



Thanks to Puerco

OSPS Baseline - Maturing with your project



Thanks to Puerco

Our newest project, Minder, allows you to define your desired security posture

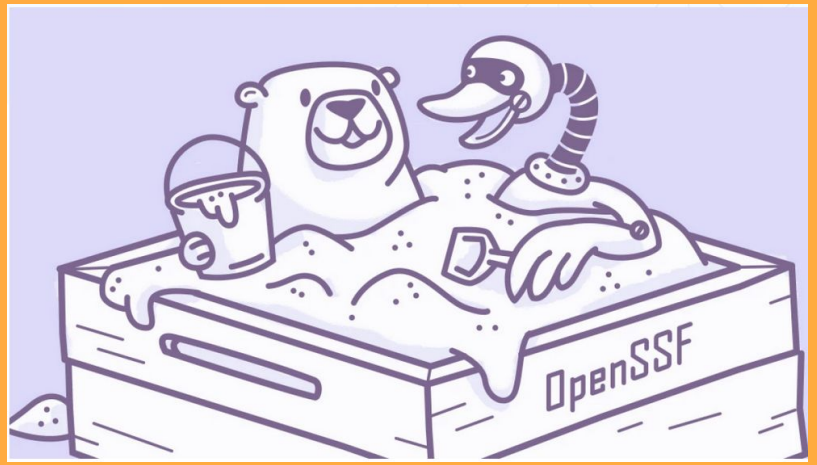


Image Source

https://openssf.org/blog/2024/10/28/openssf-adds-minder-as-a-sandbox-project-to-simplify-the-integration-and-use-of-open-source-security-tools/

DEFINING YOUR SECURITY POSTURE: RULES

Rules define conditions in your SDLC entities

Types are defined in YAML

Minder learns new rule types through its CLI:

```
minder ruletype \
  apply -f myrule.yaml
```

```yaml
---
version: v1
release_phase: beta
type: rule-type
name: actions_check_pinned_tags
display_name: Ensure immutable version of GitHub action
short_failure_message: GitHub Actions are not pinned to specific SHA-1 hashes
severity:
  value: medium
context:
  provider: github
description: Verifies that any actions use pinned tags
guidance: |
  Please change the action to use a pinned SHA-1 hash instead of a
  tag.

  Pinning an action to a full length commit SHA is currently the only
  way to use an action as an immutable release. Pinning to a
  particular SHA helps mitigate the risk of a bad actor adding a
  backdoor to the action's repository, as they would need to generate
  a SHA-1 collision for a valid Git object payload.  When selecting a
  SHA, you should verify it is from the action's repository and not a
  repository fork.

  For more information, see [GitHub's
  documentation](https://docs.github.com/en/actions/security-guides/security-hardening-for-g
def:
  # Defines the section of the pipeline the rule will appear in.
  # This will affect the template used to render multiple parts
  # of the rule
```

Curious to learn more about Minder? Watch Puerco's "Ace of Base" presentation from SOSS Fusion

BRINGING TOGETHER YOUR FAVOURITE OSSF TOOLS!

BEST PRACTICES BADGES

Interview-based verification of security practices

SCORECARD

Automated checking of 19 security aspects of a project

SECURITY INSIGHTS

Machine-readable output of assertions

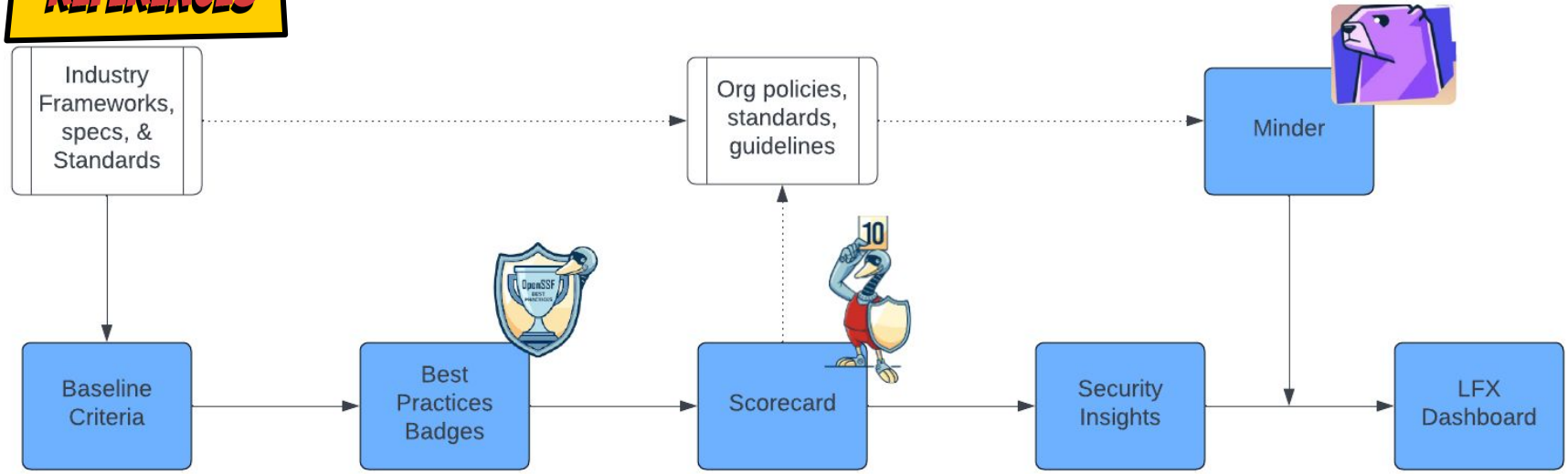
MINDER

Policy-driven decisions on desired security criteria

FUTURE WORKFLOW

REFERENCES

POLICY-BASED DECISIONS



CATALOGUE

ONBOARDING

AUTOMATED EVALUATION

MACHINE-READABLE ATTESTATIONS

COMPLIANCE DASHBOARD

BASELINE ROADMAP

- Work with 6 Pilot OSSF projects to implement the Baseline
- Work with 3 CNCF or FINOS projects to implement the Baseline
- Complete Compliance Crosswalk & publish
- Complete process documentation for updating & rolling out new/revised Baseline criteria
-
- 2025 - Complete integration with Best Practices Badge, Scorecard, Insights, Minder, & LFX
- 2025 - Work with broader LF on Baseline adoption
- 2025 - All OpenSSF projects aligned to Baseline
- 2025 - Develop automation for high-priority tasks to achieve Baseline compliance

04

CALL TO ACTION

What can YOU do?

WHAT CAN YOU DO?

- Review the Baseline criteria
- Just us feedback on criteria you feel should be included, suggestions for criteria phrasing or supporting text, proposals for Maturity Levels of criteria
- Particpate in the SIG (Every other Wednesday at 1030a ET)
- Join our Slack channel to join in the conversation
- Suggest additional compliance frameworks to account for in Compliance Crosswalk
- Explore other OpenSSF Technical Initiatives that can help you more securely manage your ingestion of open source software

THANKS

Do you have any questions?

@SecurityCRob

@SecurityCRob@infosec.exchange

https://github.com/SecurityCRob

The Security Unhappy Hour,
Chips & Salsa
What's in the SOSS?

https://www.linkedin.com/in/darthcrob/