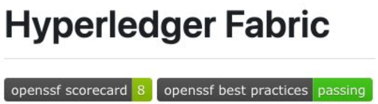
OpenSSF Scorecard

Viewer https://scorecard.dev/viewer/
https://scorecard.dev/viewer/?uri=github.com/hyperledger/fabric



Hyperledger Fabric

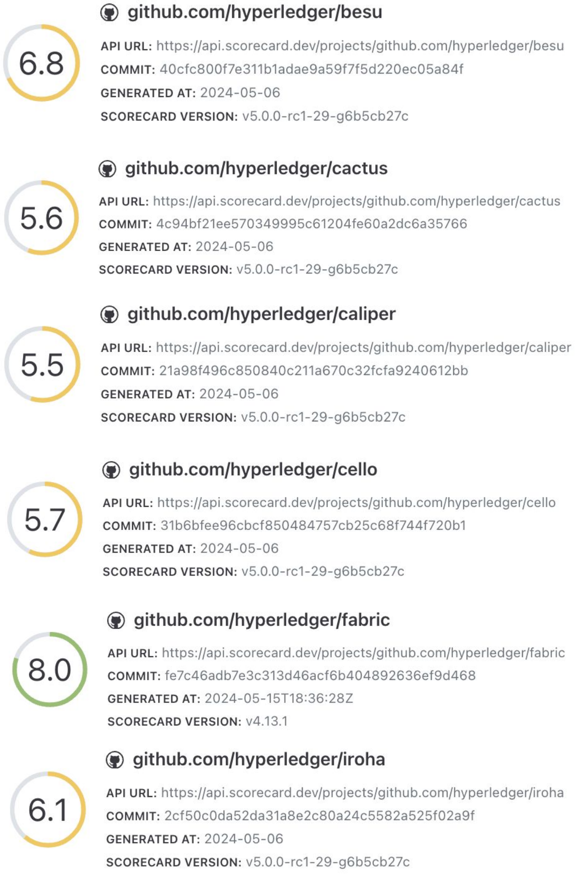
openssf scorecard 8 | openssf best practices passing

Overview
https://securityscorecards.dev/

More information
https://github.com/ossf/scorecard/tree/main

Reference
https://github.com/ossf/scorecard/blob/main/docs/checks.md

Github Action (Fabric sample)
https://github.com/hyperledger/fabric/blob/main/.github/workflows/scorecard.yml



github.com/hyperledger/besu

6.8

API URL: https://api.scorecard.dev/projects/github.com/hyperledger/besu
COMMIT: 40cfc800f7e311b1adae9a59f7f5d220ec05a84f
GENERATED AT: 2024-05-06
SCORECARD VERSION: v5.0.0-rc1-29-g6b5cb27c

github.com/hyperledger/cactus

5.6

API URL: https://api.scorecard.dev/projects/github.com/hyperledger/cactus
COMMIT: 4c94bf21ee570349995c61204fe60a2dc6a35766
GENERATED AT: 2024-05-06
SCORECARD VERSION: v5.0.0-rc1-29-g6b5cb27c

github.com/hyperledger/caliper

5.5

API URL: https://api.scorecard.dev/projects/github.com/hyperledger/caliper
COMMIT: 21a98f496c850840c211a670c32fcfa9240612bb
GENERATED AT: 2024-05-06
SCORECARD VERSION: v5.0.0-rc1-29-g6b5cb27c

github.com/hyperledger/cello

5.7

API URL: https://api.scorecard.dev/projects/github.com/hyperledger/cello
COMMIT: 31b6bfee96cbcf850484757cb25c68f744f720b1
GENERATED AT: 2024-05-06
SCORECARD VERSION: v5.0.0-rc1-29-g6b5cb27c

github.com/hyperledger/fabric

8.0

API URL: https://api.scorecard.dev/projects/github.com/hyperledger/fabric
COMMIT: fe7c46adb7e3c313d46acf6b404892636ef9d468
GENERATED AT: 2024-05-15T18:36:28Z
SCORECARD VERSION: v4.13.1

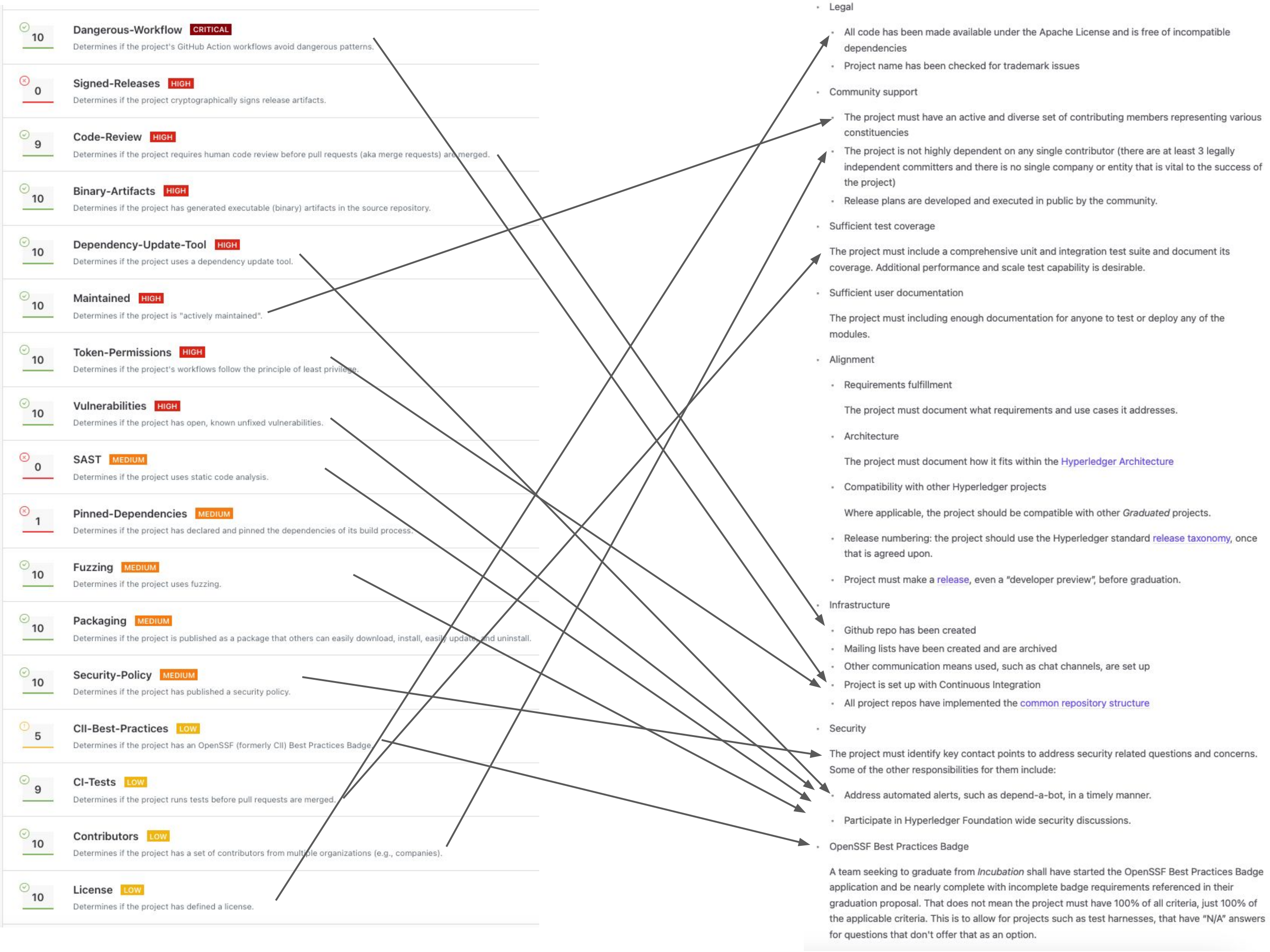
github.com/hyperledger/iroha

6.1

API URL: https://api.scorecard.dev/projects/github.com/hyperledger/iroha
COMMIT: 2cf50c0da52da31a8e2c80a24c5582a525f02a9f
GENERATED AT: 2024-05-06
SCORECARD VERSION: v5.0.0-rc1-29-g6b5cb27c

OpenSSF Scorecard

Mapping to Hyperledger graduation criteria



Score	Check	Risk	Description
10	Dangerous-Workflow	CRITICAL	Determines if the project's GitHub Action workflows avoid dangerous patterns.
0	Signed-Releases	HIGH	Determines if the project cryptographically signs release artifacts.
9	Code-Review	HIGH	Determines if the project requires human code review before pull requests (aka merge requests) are merged.
10	Binary-Artifacts	HIGH	Determines if the project has generated executable (binary) artifacts in the source repository.
10	Dependency-Update-Tool	HIGH	Determines if the project uses a dependency update tool.
10	Maintained	HIGH	Determines if the project is "actively maintained".
10	Token-Permissions	HIGH	Determines if the project's workflows follow the principle of least privilege.
10	Vulnerabilities	HIGH	Determines if the project has open, known unfixed vulnerabilities.
0	SAST	MEDIUM	Determines if the project uses static code analysis.
1	Pinned-Dependencies	MEDIUM	Determines if the project has declared and pinned the dependencies of its build process.
10	Fuzzing	MEDIUM	Determines if the project uses fuzzing.
10	Packaging	MEDIUM	Determines if the project is published as a package that others can easily download, install, easily update, and uninstall.
10	Security-Policy	MEDIUM	Determines if the project has published a security policy.
5	CII-Best-Practices	LOW	Determines if the project has an OpenSSF (formerly CII) Best Practices Badge.
9	CI-Tests	LOW	Determines if the project runs tests before pull requests are merged.
10	Contributors	LOW	Determines if the project has a set of contributors from multiple organizations (e.g., companies).
10	License	LOW	Determines if the project has defined a license.

- Legal
  - All code has been made available under the Apache License and is free of incompatible dependencies
  - Project name has been checked for trademark issues
- Community support
  - The project must have an active and diverse set of contributing members representing various constituencies
  - The project is not highly dependent on any single contributor (there are at least 3 legally independent committers and there is no single company or entity that is vital to the success of the project)
  - Release plans are developed and executed in public by the community.
- Sufficient test coverage

  The project must include a comprehensive unit and integration test suite and document its coverage. Additional performance and scale test capability is desirable.
- Sufficient user documentation

  The project must including enough documentation for anyone to test or deploy any of the modules.
- Alignment
  - Requirements fulfillment

    The project must document what requirements and use cases it addresses.
  - Architecture

    The project must document how it fits within the Hyperledger Architecture
  - Compatibility with other Hyperledger projects

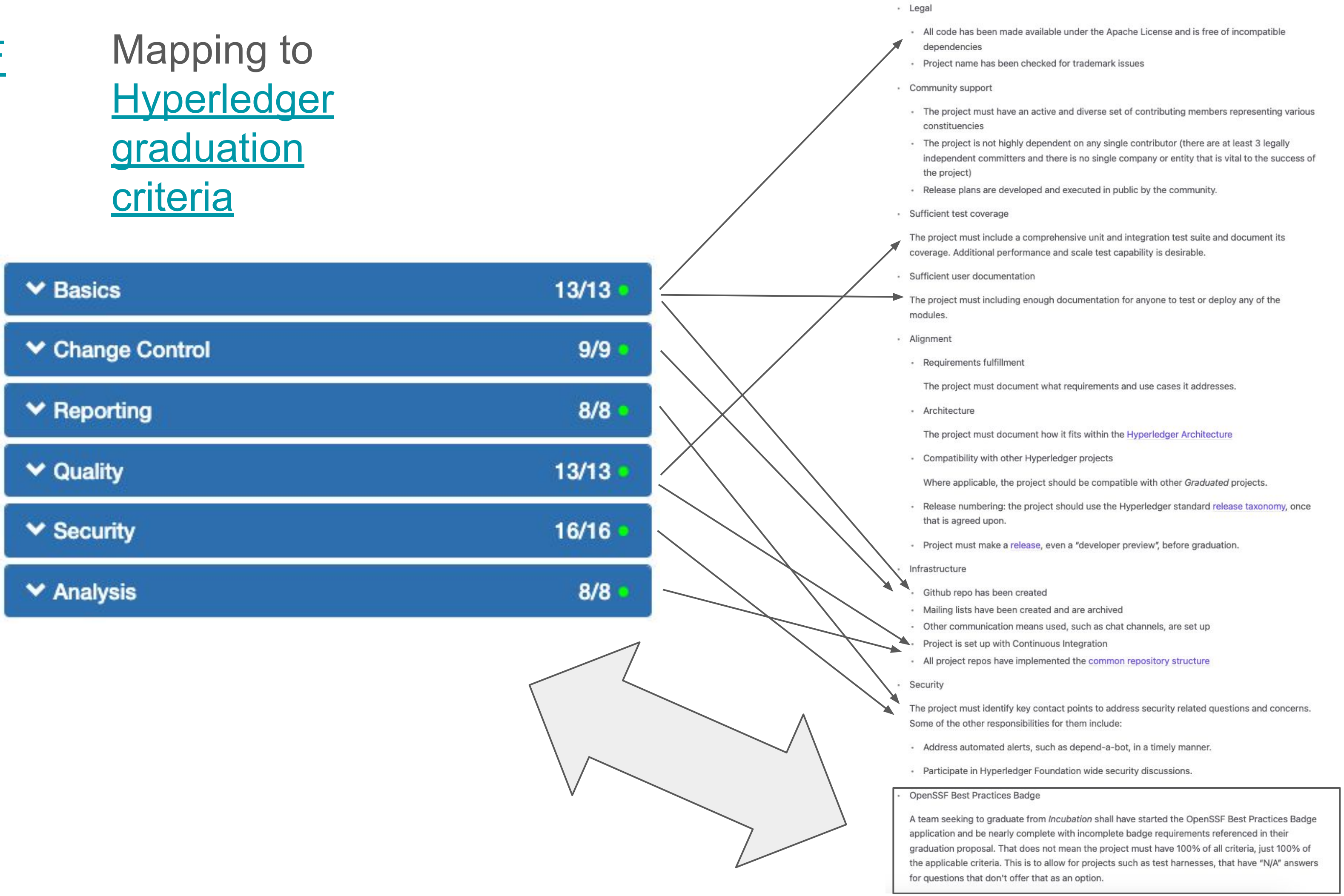
    Where applicable, the project should be compatible with other *Graduated* projects.
  - Release numbering: the project should use the Hyperledger standard release taxonomy, once that is agreed upon.
  - Project must make a release, even a "developer preview", before graduation.
- Infrastructure
  - Github repo has been created
  - Mailing lists have been created and are archived
  - Other communication means used, such as chat channels, are set up
  - Project is set up with Continuous Integration
  - All project repos have implemented the common repository structure
- Security

  The project must identify key contact points to address security related questions and concerns. Some of the other responsibilities for them include:
  - Address automated alerts, such as depend-a-bot, in a timely manner.
  - Participate in Hyperledger Foundation wide security discussions.
- OpenSSF Best Practices Badge

  A team seeking to graduate from *Incubation* shall have started the OpenSSF Best Practices Badge application and be nearly complete with incomplete badge requirements referenced in their graduation proposal. That does not mean the project must have 100% of all criteria, just 100% of the applicable criteria. This is to allow for projects such as test harnesses, that have "N/A" answers for questions that don't offer that as an option.

OpenSSF Best Practices

Mapping to Hyperledger graduation criteria



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

- Legal
  - All code has been made available under the Apache License and is free of incompatible dependencies
  - Project name has been checked for trademark issues
- Community support
  - The project must have an active and diverse set of contributing members representing various constituencies
  - The project is not highly dependent on any single contributor (there are at least 3 legally independent committers and there is no single company or entity that is vital to the success of the project)
  - Release plans are developed and executed in public by the community.
- Sufficient test coverage

  The project must include a comprehensive unit and integration test suite and document its coverage. Additional performance and scale test capability is desirable.
- Sufficient user documentation

  The project must including enough documentation for anyone to test or deploy any of the modules.
- Alignment
  - Requirements fulfillment

    The project must document what requirements and use cases it addresses.
  - Architecture

    The project must document how it fits within the Hyperledger Architecture
  - Compatibility with other Hyperledger projects

    Where applicable, the project should be compatible with other *Graduated* projects.
  - Release numbering: the project should use the Hyperledger standard release taxonomy, once that is agreed upon.
  - Project must make a release, even a "developer preview", before graduation.
- Infrastructure
  - Github repo has been created
  - Mailing lists have been created and are archived
  - Other communication means used, such as chat channels, are set up
  - Project is set up with Continuous Integration
  - All project repos have implemented the common repository structure
- Security

  The project must identify key contact points to address security related questions and concerns. Some of the other responsibilities for them include:
  - Address automated alerts, such as depend-a-bot, in a timely manner.
  - Participate in Hyperledger Foundation wide security discussions.
- OpenSSF Best Practices Badge

  A team seeking to graduate from *Incubation* shall have started the OpenSSF Best Practices Badge application and be nearly complete with incomplete badge requirements referenced in their graduation proposal. That does not mean the project must have 100% of all criteria, just 100% of the applicable criteria. This is to allow for projects such as test harnesses, that have "N/A" answers for questions that don't offer that as an option.

OpenSSF Best Practices	OpenSSF Scoreboard	Proposed Badge
Basics→License	License	Legal
Change Control→Repository	CII Best Practices	Structure
	Contributors (from multiple organizations)	Diversity/Decentralized
Change Control→Version numbering	Packaging	Release
Quality→Builds Quality→Automated test suite Quality→New functionality testing	CI Tests Dangerous Workflow Token Permissions	Testing and CI/CD
Basics→Documentation		Documentation
Reporting→Vulnerabilities Security→Vulnerabilities fixed Analysis→Static code analysis Analysis→Dynamic code analysis	Security Policy Dependency Update Tool Pinned Dependencies, Vulnerabilities Fuzzing, SAST	Security
		Conformity
Basics→Contributions Reporting→Bugs		Responsiveness/Engagement
		Production