FINOS

Fintech
Open Source
Foundation

Balancing risk mitigation among compliance practices

Aaron Williamson
Open Source Readiness Lead

October 2019

Risks managed via OSS compliance program

- Security vulnerabilities
- Open source IP infringement
- Disclosure of proprietary code
- Disclosure of sensitive data

Risks managed: open source dependencies

Relevant to both security vulnerabilities & open source IP:

- Explicit OSS dependencies (build system)
- Local OSS dependencies - source
- Local OSS dependencies - partial/snippets
- Local OSS dependencies - containers
- Local OSS dependencies - other binaries

Risks managed: disclosing proprietary code

- Proprietary modifications to OSS code
- Proprietary components of corporate open source project
- Copyleft (e.g. GPL) code in proprietary product

Risks managed: disclosing sensitive data

- Security-sensitive data (e.g. keys, passwords)
- Privacy-sensitive data (i.e. PII)
- Business-confidential data (e.g. network architecture)
- Customer-confidential data
- Embarrassing data (e.g. code comments)

Key OSS Compliance Program Processes

- **Training** of developers, managers, and legal & compliance staff
- **Documentation** of OSS policies, guidelines, and systems
- **Information management** re: use of OSS components
- **Approval workflows** for new OSS licenses and components
- **Automation** to detect OSS components and vulnerabilities
- **Code review** of OSS contributions and product releases
- **Audit** of existing products

Training

- Purpose: inform about open source issues, corporate policies, employee role in risk mitigation, specific guidelines and practices
- Strengths: mandatory & tracked, broad-based
- Weaknesses: general, infrequent, limited retention, knowledge expires
- Best for: OSS - source & partial
- Worst for: OSS - build

Documentation

- Purpose: ready reference for information about policies, guidelines, processes, and systems
- Strengths: current, comprehensive, available as-needed
- Weaknesses: depends on user initiative
- Best for: all
- Worst for: all
- Depends on: training

Information management

- Purpose: track information about OSS components used, modifications, security alerts, BOLO info re: disclosure
- Strengths: captures nuance, integrates with automation
- Weaknesses: large manual component, garbage in/garbage out
- Best for: OSS - partial, container/other, proprietary code, sensitive data
- Worst for: OSS - build & source
- Depends on: training, documentation, automation

Approval workflows

- Purpose: enforce clearance of new OSS licenses, components, contributions through required channels
- Strengths: uses existing systems, integrates with automation and information management
- Weaknesses: bottleneck on people's availability, opportunities for circumvention (esp. w/o automation), large backlog before common licenses and components are cleared
- Best for: OSS - all
- Depends on: training, documentation, info mgmt, automation

Automation

- Purpose: build compliance and security checks into SDLC
- Strengths: identify issues as they arise, raise issues/tickets automatically, potentially comprehensive
- Weaknesses: major engineering effort, complexity multiplies with technologies, costly, false negatives & positives
- Best for: OSS - build & source, proprietary mods, some sensitive data
- Worst for: OSS - partial & binary, proprietary code, other sensitive data
- Depends on: training, documentation, information management

Code review

- Purpose: catch issues automation can’t, failsafe for automation
- Strengths: expands existing process, better than automation for IP leakage and some sensitive data
- Weaknesses: highly manual, training-intensive
- Best for: sensitive data, proprietary code
- Worst for: everything else
- Depends on: training, documentation, information management, approval workflows

Audit

- Purpose: identify issues in existing products
- Strengths: multiple approaches available, parallelizable
- Weaknesses: slow, expensive (in time or vendor tools), training-intensive, results age w/development
- Best for: OSS - all, sensitive data
- Worst for: all
- Depends on: training, documentation, information management, automation (to keep results current)

Process-to-risk mapping

	Training	Docs	Info Mgmt	Approvals	Automation	Code review	Audit
OSS (build)							
OSS (source)							
OSS (partial)							
OSS (container)							
OSS (other)							
Proprietary mods							
Proprietary code							
Sensitive data							

Which processes are you using to control for which risks?

FINOS

Fintech
Open Source
Foundation

info@finos.org

finos.org