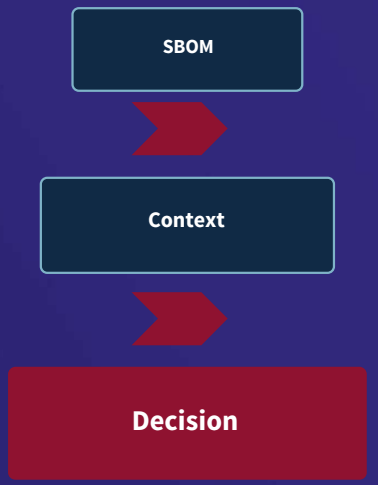
OpenSSF Community Day

From SBOMs to Decisions:

Prioritizing Supply Chain Risk

in Time-Bound M&A Reviews



How assessment-side evidence becomes decision-ready for time-bound software diligence.

Prashanth Chandrasekar

Bitsea US, Inc.

OpenSSF Community Day
NORTH AMERICA 2026

Prashanth Chandrasekar

Principal Open Source Consultant · Bitsea US, Inc.

- Open source risk, SCA, AppSec, and software supply chain security
- Supports M&A diligence across vulnerabilities, licenses, and provenance
- 11+ years turning software risk into decision-ready findings.

11+ years
OSS / SCA reviews

M&A technical diligence

Supply chain
licenses · vulns · provenance

The M&A Problem

Constraints in a transaction review

Limited visibility

You may not get all the artifacts

Time is fixed

The diligence window is short

Action-forcing decisions

Need to classify risk quickly

Evidence assembled

- SBOMs
- Vulns & EOL
- Code evidence & provenance
- Controls & Patch readiness
- Buyer business impact

Questions from the buyer team

C-Suite

“ Any outstanding issues?

“ Can we buy this company? Price?

Security team

“ How is our Day 1 readiness?

Integration / Engineering team

“ How do we integrate this to our product lines?

Timeline of an M&A review

Days 1-2

Scope

Products, artifacts, distribution model

Days 2-8

Inventory

Scans, discovery, dependencies, generate SBOMs

Target Review

Days 9-13

Prioritize

License, vulnerability, and provenance risk

Confirm

Confirm shipping, linking, exploitability, ownership

Days 14-15

Readout

Deal actions and remediation path

Likely review outcomes

Gate	Price	Accept	Pause
Must address before close (Covenant)	Affects economics	Manage post-close	Need more clarity

Generating Inventory

SBOMs are necessary, but not sufficient

SBOM

COMPONENT	**openssl 3.x**
DEPENDENCY	**log4j 2.x**
TRANSITIVE	**curl / protobuf**
INTERNAL	**proprietary-lib**

Visibility into what is inside the product.

Decision Context still matters

The SBOM becomes useful when surrounded by practical evidence.

1. **Where is the component used?**
   Customer-facing? Revenue-critical? Internal-only?
2. **Is the component shipped?**
   In product? In artifact? In deployment path?
3. **Is the affected code reachable?**
   Executed? Internet-facing? Controlled environment?
4. **Where did the code come from?**
   Source, build system, or copied snippet?
5. **Can the issue be fixed?**
   Owner identified? Verified mitigation timeline?

Decision Support

1. **Context**
   Product relevance mapping
2. **Confidence**
   Evidence-based verification
3. **Actions**
   Buyer-ready next steps

An SBOM provides visibility; the evidence stack
makes it decision-ready.

Validating the Inventory

1. **Coverage gaps**
   Build-time, runtime, container, or vendored code is missing.

2. **Noisy vulnerability mapping**
   Control large volumes of CVEs in unaffected code paths.

3. **License ambiguity**
   Check source notices and LICENSE files, package metadata and other disclosed components

4. **Missing usage context**
   Map component to products, artifacts and owner.

Prioritize Deal Risk

TRIAGE MODEL

CVE volume and license labels are inputs — deal impact determines priority.

Inputs

SBOM + evidence

SBOM

- license — transitive
- vulns — shipped
- snippets — exposure

License · CVE · Reachability · Distribution

Score the signal

Five lenses, one priority view

1. License impact
2. Exploitability
3. Business criticality
4. Fix friction
5. Execution readiness

Map to action

- Accept · Low impact
- Covenant · Reasonable cleanup
- Gate · Fix before close
- Price · Economic impact
- Pause · No-go uncertainty

Examples: Turning Findings into Actions

Issue #1: Code Snippets from unknown licensed blog articles

Closing Covenant

Snippet license uncertainty

Copied or unknown-license code appears in a shipped path.

What to verify

Shipped to customers?
Source / license known?
Rewrite path available?

Deal lens

Limited scope + credible cleanup path.

Likely action

COVENANT

Seller cleans up and provides proof before close.

Examples: Turning Findings into Actions

Issue #2: Apache Tika -> Apache Commons Compress transitive vulnerability

Vulnerability Signal

Distributed transitive Vulnerability issue

A direct dependency brings vulnerable transitives into the artifact.

What to verify

Shipped and reachable?
Uploads reach parser path?
Controls / owner?

Deal lens

Level of Exposure changes the transaction response.

Likely action

TRACK LATER / GATE

Depends on exposure and fixability.

Examples: Turning Findings into Actions

Issue #3: Redis -> Valkey Replacement Path

Gate Before Close

Vuln–license conflict with alternatives

Objective: to stay away from the restrictive SSPL.

A safer path improves license obligations but changes security posture.

What to verify

Legacy vs supported path
Commercial / license impact
Viable replacement path?

Deal lens

The fix path can change future product constraints.

Likely action

GATE BEFORE CLOSE

Resolve the replacement path before close.

Examples: Turning Findings into Actions

Issue #4: XZ Utils / liblzma artifact trust

Pause / No-Go

Untrusted upstream release artifact

Compromised upstream package may affect shipped product or base image.

What to verify

Affected version present?
Shipped image or appliance?
Clean rebuild / provenance?

Deal lens

If artifact trust is uncertain, product trust is uncertain.

Likely action

PAUSE

Pause until distribution is verified or the artifact is replaced.

READOUT

What a Decision-Ready Output Looks Like

IMPORTANT: Legal-safe Remediation actions: evidence + implications + options, not conclusions

1 Portfolio view
What product or workflow is affected?

2 Risk rationale
Why does it matter to the deal?

3 Required action
What must happen before or after close?

Issue	Risk	Description	Deal relevance	Action
Code snippet license uncertainty	**Yellow**	Unknown-license snippets in shipped code	Needs provenance + distribution review	Covenant + proof of cleanup
Shipped transitive vulnerability	**Yellow**	Vulnerable transitive in shipped artifact	Exposed risk if shipped and reachable	Remediate, price, or gate
Vuln–license conflict with alternatives	**Yellow**	Safer path changes license / commercial posture	Security fix can alter product direction	Gate close on replacement path
Untrusted upstream release artifact	**Red**	No credible owner or near-term fix	Close timing + viability concern	Pause until disproven or cured

Where OpenSSF Helps the Analysis

OpenSSF projects enrich the evidence stack: from SBOM coverage to posture and provenance signals.

Review level	Question addressed	OpenSSF project	How it helps
1 **Software inventory**	What is in the product?	**BOMCTL**	Bridges the gap between SBOM generation and analysis tools.
2 **Dependency context**	How do components connect and propagate?	**GUAC**	Connects SBOMs, dependencies, and attestations into a queryable graph.
3 **Security posture**	Health checks?	**Scorecard + Allstar**	Adds project-hygiene and control signals to posture review.
4 **Provenance & trust**	Can the artifact be trusted?	**SLSA + Sigstore**	Brings provenance, signing, and build-integrity evidence.
5 **Remediation guidance**	What does good look like going forward?	**OSPS Baseline**	Provides a practical baseline for gaps and remediation direction.

Thank you

Q&A

Connect on LinkedIn

Continue the conversation on SBOMs and software supply chain risk.

Bitsea US, Inc.