Trusting the Free Stuff

HOW MSPS ASSESS RISK IN OPEN SOURCE TOOLS

CHANNELCON
POWERED BY GTIA

San Diego
August 3-5, 2026

Jared Casner

BLACKSMITH INFOSEC

Agenda

- TPRM: Commercial Software vs Open Source
- Your current risk landscape
- The 5-question OSS Risk Review Framework
- Takeaways

Commercial vs Open Source

#channelcon

CHANNELCON
POWERED BY GTIA

San Diego
August 3-5, 2026

Third Party Risk Management

TRADITIONAL

- SOC 2 / ISO 27001
- Pentest
- Security Questionnaire
- Contract terms
- Support contact

VS.

OPEN SOURCE

- no salesperson
- no security questionnaire
- no account manager
- no warranty
- maybe no company at all

IS OPEN SOURCE LESS SECURE? NO, IT'S JUST DIFFERENT

Commercial Software Runs on OSS

A CONTRACT TELLS YOU WHO TO CALL, NOT WHAT IS INSIDE.

Ask your vendors:

1. Do you maintain an SBOM?
2. Is it product/version/build specific?
3. How often is it refreshed?
4. Do you monitor it for new vulnerabilities?
5. How do you notify me when an OSS component is affected?

Common Myths

COMMERCIAL IS BETTER

- Commercial = Safe
- Contracts protect the consumer

OPEN SOURCE IS BETTER

- More eyes = safer
- Open source is cheaper

A Little Dose of Reality...

OPEN SOURCE...

- ...is already part of your stack, both directly and indirectly
- ...can be reviewed
- ...needs to be reviewed differently

Your current risk landscape

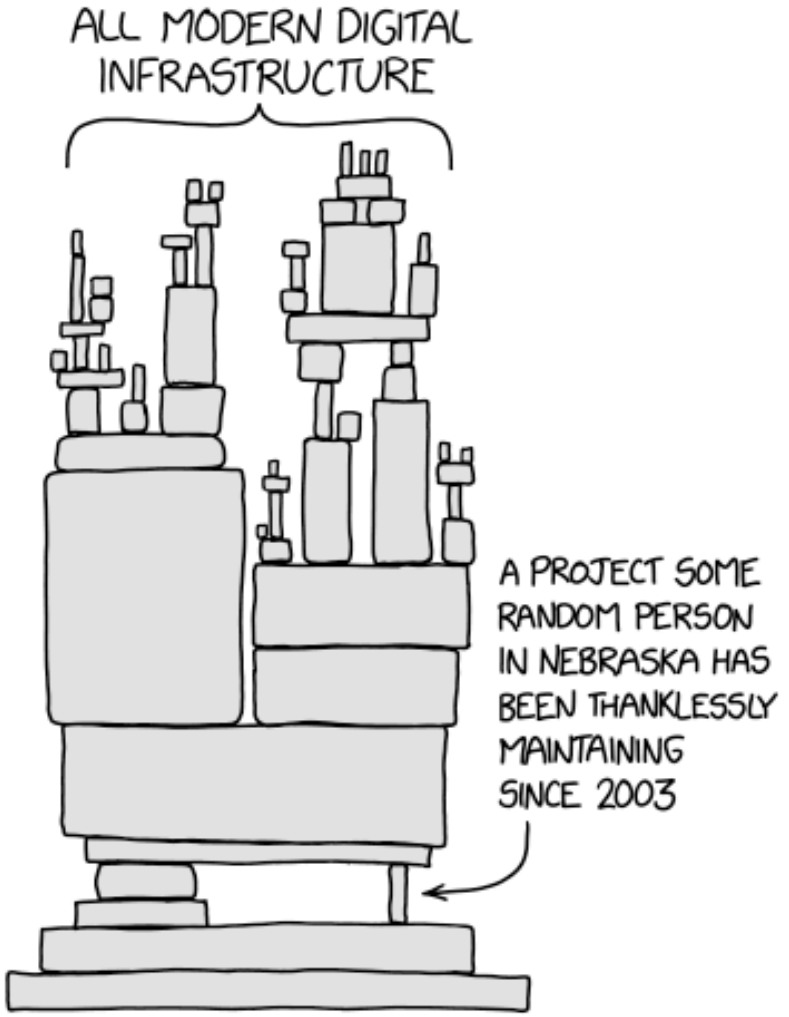
#channelcon

CHANNELCON
POWERED BY GTIA

San Diego
August 3-5, 2026

Open Source Gone Wild...

- Heartbleed (2014)
- Shellshock (2014)
- left-pad (2016)
- Log4Shell (2021)
- XZ Utils (2024)



So, what's in scope?

- WordPress plugins and themes
- Browser extensions
- Docker containers
- Linux packages
- firewall/router packages
- monitoring agents
- backup scripts
- PowerShell modules
- Python utilities
- NPM packages embedded inside commercial products
- SIEM rules, detection content, dashboards
- self-hosted tools

IF IT RUNS IN THE CLIENT ENVIRONMENT, TOUCHES CLIENT DATA, AFFECTS AVAILABILITY, OR BECOMES PART OF YOUR SERVICE DELIVERY

What would you review more carefully?

- A free open source password manager
- A free Chrome extension
- A Docker image from a random GitHub repo
- A WordPress plugin with 500,000 installs
- A commercial SaaS tool with no published security docs
- Code generated by Claude Code (or any other LLM)

OSS Risk Review Framework

5 QUESTIONS YOU NEED TO ANSWER

#channelcon

CHANNELCON
POWERED BY GTIA

San Diego
August 3-5, 2026

1) What does it touch, and how bad is it if it breaks?

How much trust are we giving this thing, and how painful is failure?

1. DATA ACCESS

Can it read your secrets, rewrite your spreadsheets, or yeet a database?

2. ADMIN PRIVILEGES

Does it need god-mode, or is it just being dramatic?

3. NETWORK EXPOSURE

Can it phone home, hit random ports, or wander the network unsupervised?

4. PROD VS. LAB USE

Cute in the lab. Still cute in prod?

5. CLIENT IMPACT IF IT FAILS

If this thing faceplants, who gets yelled at first?

Trust nothing.
Verify everything.
Document the pain.

PARANOIA IS A FEATURE, NOT A BUG.

PLEASE DON'T GET HIT BY A BUS

2) Who maintains it, and is it actually alive?

Is there a healthy ecosystem around this project, or are you betting production on wishful thinking?

STILL MAINTAINED? PROVE IT.

1. SINGLE MAINTAINER VS. TEAM

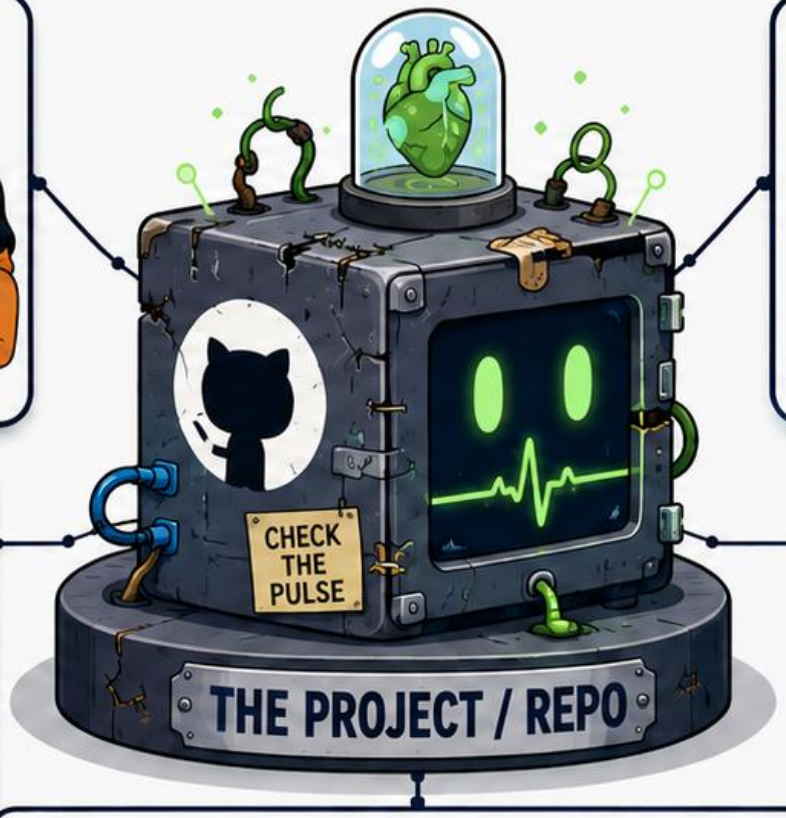
One brave goblin, or an actual crew?

Busy being the whole department.

2. FOUNDATION-BACKED VS. INDEPENDENT PROJECT

Backed by a real org, or independently maintained by people who consistently show up?

Independent can be excellent.



3. MAINTAINER RESPONSIVENESS

When stuff breaks, do they answer, ghost, or vanish into the void?

Seen 2 years ago.

4. IS IT ACTIVE?

Recent commits, releases, and issue activity... or still haunting 2021?

Pushed 2 days ago

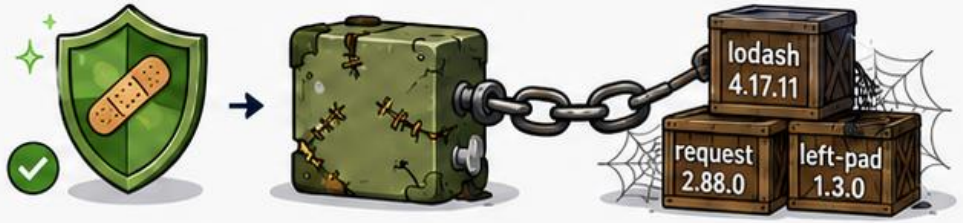
Released 1 week ago

Last update 2 years ago

It's alive... maybe.

5. SECURITY FIXES & STALE DEPENDENCIES

Do they patch scary stuff, or is ancient code holding this thing together?



Patches > prayers.

SOME ASSEMBLY REQUIRED

ABANDONWARE DETECTED? PROCEED WITH CAUTION

TRUST BUT VERIFY

3) HOW DOES IT HANDLE SECURITY?

SECURITY ISN'T OPTIONAL

A project can be active and still be terrible at security.

THE PROJECT / REPO

1. SECURITY POLICY

Do they tell you where to report bugs, or are you expected to telepathically DM the maintainer?

Please don't disclose into the void.

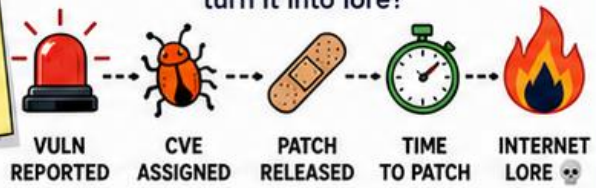
2. DISCLOSURE PROCESS

Is there a real process, or is the plan basically 'good luck'?

Step 1: Panic. Step 2: ???

3. CVE HISTORY & PATCH SPEED

When issues show up, do they patch fast, or let the internet turn it into lore?



Fast patches beat wishful thinking.

4. SIGNED RELEASES

Can you verify this release, or are you just taking it on faith?

Verify the package, not the promise.

5. CHANGELOG QUALITY

Do release notes explain what changed, or just say 'misc fixes" and disappear?

Fixed stuff. Probably.

DISCLOSURE IS A FEATURE, NOT A SIDE QUEST.

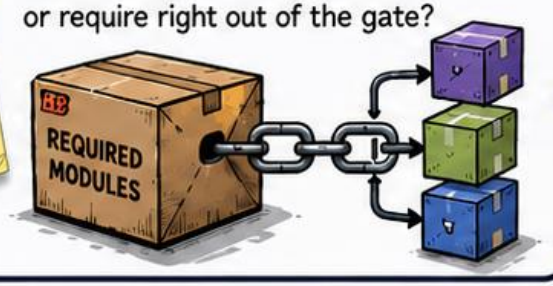
4) WHAT ARE ITS DEPENDENCIES AND SUPPLY CHAIN RISKS?

The tool is not just the tool. It's the code it pulls in.

1. DIRECT DEPENDENCIES

What does it knowingly install or require right out of the gate?

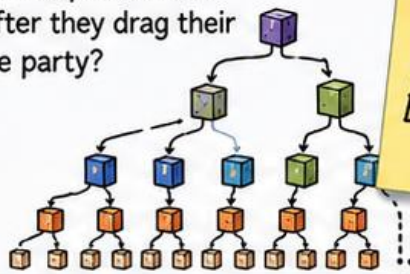
Know the starter pack.



2. TRANSITIVE DEPENDENCIES

What do those dependencies depend on after they drag their friends to the party?

Small package. Big family tree.



3. PACKAGE SOURCE & PROVENANCE

Are you installing the real thing from a source you trust, or a lookalike with a sketchy origin?

Typosquats love busy people.

4. BUILD & RELEASE TRUST

Can you trust how this thing gets built and shipped, or are we just downloading mystery meat?

UNSIGNED? RISKY.

Trust the pipeline less than you think.

5. DEPENDENCY SPRAWL

How big is the blast radius when one tiny package goes weird?

Tiny package, giant blast radius.

⚠ SUPPLY CHAIN RISK = YOUR PROBLEM.

APPROVAL ISN'T THE FINISH LINE.

5) How will we monitor and own it after approval?

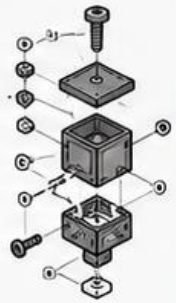
Who is responsible for not getting surprised later?

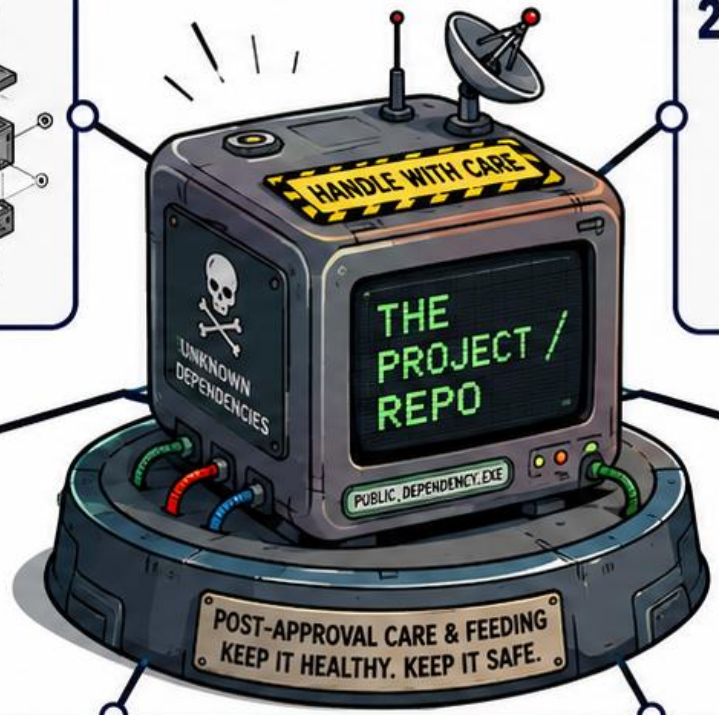
ADOPT IT. DON'T ABANDON IT.

1. SBOM

Do we know what's in this thing, or are we just hoping?

Nutrition facts for code.





2. Vulnerability monitoring

How will we know when one of its parts catches fire?

Pager go brrr.

3. Release monitoring

Who notices when the project ships updates, fixes, or weirdness?

v1.2.1 → v1.2.2 → v1.2.3

Update all the things. Carefully.

4. Sources to watch

OSV, GitHub Advisories, NVD... where are we actually checking?

OSV

GitHub Advisories

NVD

One feed to rule them all. Sadly, no.

5. Ownership & re-evaluation

Who owns patching and review, and when do we revisit the decision?

If everyone owns it, nobody owns it.

WHO OWNS IT?

WHAT WE OWN

REVIEW REGULARLY

RE-EVALUATE

PASS IT ON (ON PURPOSE)

APPROVED ONCE ≠ SAFE FOREVER.

RECEIPT

TRUST, BUT KEEP RECEIPTS.

SURPRISES ARE FOR BIRTHDAYS, NOT REPOS.

SOMEONE OWNS THIS. ON PURPOSE.

The Decision Model

FOUR POSSIBLE OUTCOMES

1. Approve
2. Approve with Guardrails
3. Pilot / Sandbox Only
4. Reject / Replace

Some Scenarios to Talk Through

HINT: THIS IS INTERACTIVE...

1. Self-hosted documentation tool with limited access, regular releases, security policy
2. Self-hosted configuration tool with write access to your MS admin console; bi-weekly releases, security policy, active community
3. Random PowerShell script requiring global admin from a personal GitHub repo
4. Popular WordPress plugin, 500k installs, stale releases, unresolved security issues

Sample Decision Language

APPROVE WITH GUARDRAILS:

- “We reviewed this tool using the same risk lens we use for commercial vendors, but with open-source-specific signals. The project is actively maintained, has a security disclosure process, publishes frequent releases, and will be monitored for vulnerabilities. We recommend using it with these guardrails.”

REJECT / REPLACE:

- “We do not recommend this tool in production because it requires sensitive access, has limited maintainer activity, and does not have a clear security response process. The functionality is useful, but the operational risk is too high for this environment.”

Takeaways

#channelcon

CHANNELCON
POWERED BY GTIA

San Diego
August 3-5, 2026

The 5 Questions Before You Approve OSS

Assess the project, not the logo.

1. What does it touch, and how bad is it if it breaks?
2. Who maintains it and is it actually alive?
3. How does it handle security?
4. What are its dependencies and supply chain risks?
5. How will we monitor it and own it after approval?

DECISION FRAMEWORK

- Approve
- Guardrail
- Sandbox
- Reject

Open source is not the problem. **Unexamined trust is.**

Take-Home Resources

Everything from this talk in one place.

The landing page includes:

- Slides from this session
- Open source risk checklist
- Tools to monitor risk
- Further reading and updates
- How to contact me with questions / follow up

Scan here

For the landing page

Open source is not a problem. **Unexamined risk is**