Open Source Doesn't Care about YOU

....but YOU should care about IT!

Who is THIS guy?

CRob, n, adj, and v

Pronunciation: U.S. (K-robe)

- CRob is a 41st level Dungeon Master
- 24th level Securityologist
- Ambassador For Intel Product Assurance and Security – I help manage brand reputation around security
- Working Group lead for the OpenSSF Dev Best Practices & Vuln Coordination WGs, OpenSSF TAC member, FIRST PSIRT TPC WG, and others
- Co-Author FIRST PSIRT Services Framework & others
- Pirate-enthusiast & hat-owner

The thoughts and feels expressed here are personally held or experientially earned, and not necessarily those of my employer

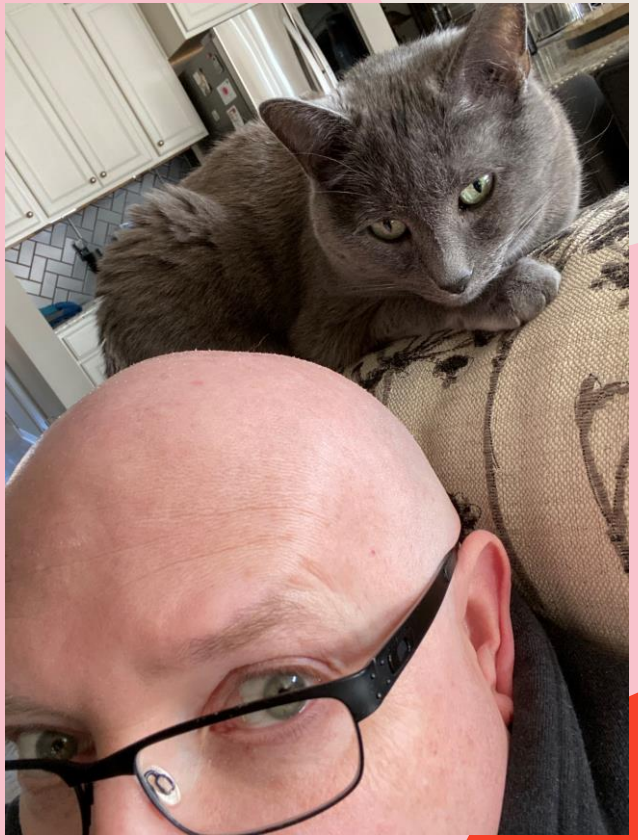


TABLE OF CONTENTS

Pop Quiz, hotshots!

How many people here use open source at work?

How many people think they know where ALL the OSS is they use is?

Who here has an SBOM from their suppliers?

Of those, who can actually USE that SBOM and tie it to REAL problems that need solved?

[Image Source]


01
Framing YOUR
OSS “Problem”

[Image Source]

The Many Goals of OSS projects

Solving a problem

Academic project

Building a community/helping others

Having fun/learning something new

Seeking recognition from peers

"Since I use this piece of FOSS, I feel I should contribute back to it"

"I believe in the mission of FOSS or the particular area I contribute to"

Feel that contributions will help their career

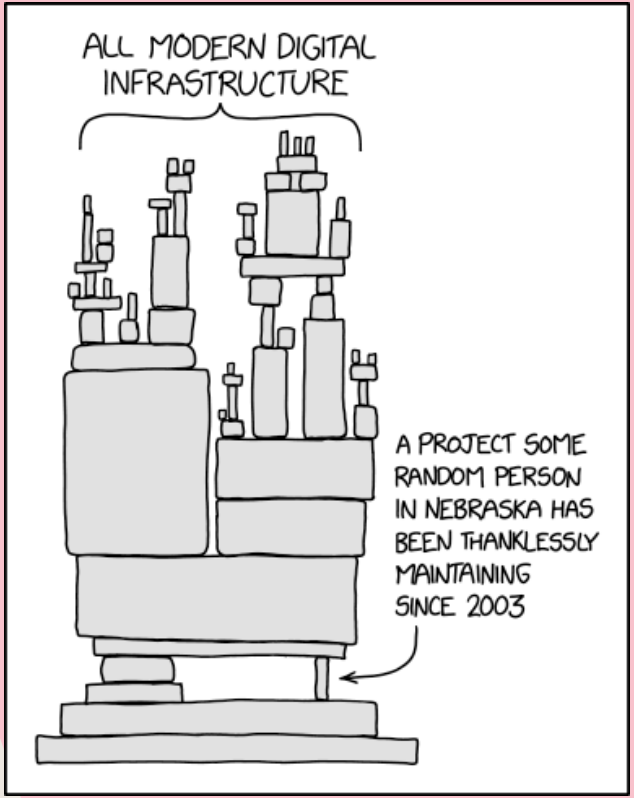
Paid by some company for some reason to work on FOSS

Linux Foundation – Report on the 2020 FOSS Contributor Survey

[Image Source]

FUN Fact!

91% of all commercial software uses OSS components today*



[Image Source]

* Synopsys Study Shows 91% of Commercial Applications Contain Outdated or Abandoned Open Source Components - 12May2020 SecurityWeekly article

What are you doing when you use open source software?

You are knowingly (or unknowingly) using SOMEONE's work to solve YOUR problem without offering compensation to the originator so that YOU can offer a profitable product or service to your customers and reduce YOUR costs

…and that's OK-ish. OSS is created for the Public Good with no expectation of payment as long as you understand a few things…

FIRST - Your business needs may not align with the developer/communities needs or desires

[Image Source]

What does OSS "do" for YOU?

Aka Let CRob dissuade you from some illusions you may have….

[Image Source]

OSS never gave you a support contract

OSS doesn't care if you have dependency conflicts

OSS doesn't know and doesn't care how YOU cobbled together YOUR solution out of random software off of the Internet

OSS does NOT care that YOU gave YOUR customers a 10 year support contract

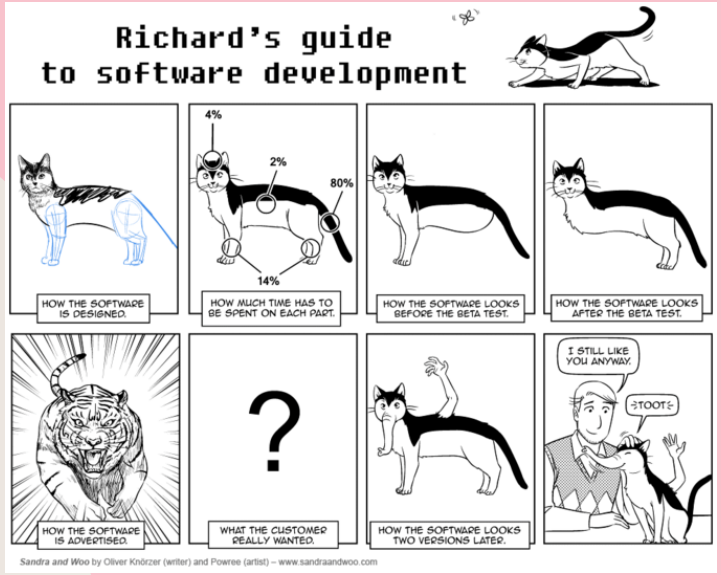
OSS doesn't care that you haven't or can't upgrade to the latest version

02

How OSS Works / How Closed Source Works

How OSS projects are supposed to work

Developer lovingly creates something awesome → Developer gives their pride and joy to the UNIVERSE → Others discover this wonderous gift → Some give back and add to the great thing



[Image Source]

How many Suppliers treat OSS

[Image Source]

What makes OSS interesting to attackers?

[Image Source]

Deobfuscated and **public-facing source code** lowers attacker barrier to entry

Distributed **community-driven development** with contributions from unknown third-parties

Different economic incentives & feedback loops than enterprise devs & threat actors

Lack of resources for monitoring & typical **underpreparedness for incident response**

Lack of consistently-deployed security standards, reviews and tooling

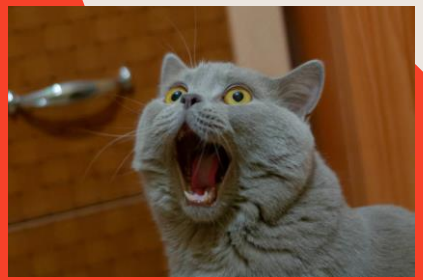
From Fernick & Robinson BHUSA2021 – “Securing Open Source Software: End-to-End, At Massive Scale, Together”

How security vulnerabilities are addressed in assorted OSS communities

It depends.....

“Individual Contributor”

- 1-2 people
- Typically passion-projects
- No-to-little process or tooling
- Likely no formal or secure means of sharing security-related issues privately
- No SLAs on review or remediation
- Minimal, if any, documentation or DevSecOps capability



[Image Source]

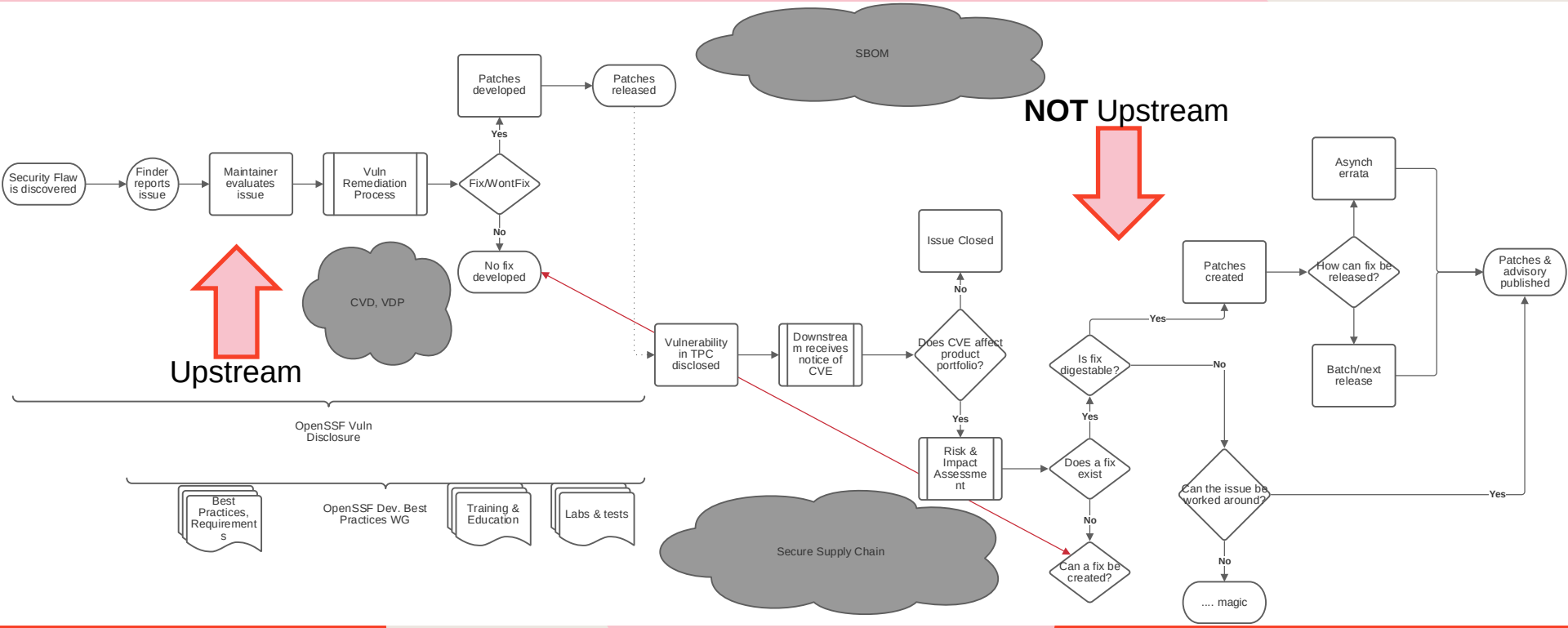
Intermediate: Project / Community focused

- Small-Large group contributing
- Will have basic processes & tooling in place
- Probably have listed contact information/communication process
- Has bug review process, may or may not be able to handle security reports privately
- Most likely will “Have a guy/gal that does that” for DevSecOps
- Stronger documentation/avenues to engage community
- “Best effort” coordination
- Possible CVD practices with key downstream suppliers

Mature: Foundation-level

- Many related projects
- Typically has budget & specialists
- Well developed and documented processes and issue intake
- Well-known/well-documented communication channels includes means to privately disclosure security issues
- Dedicated people performing compose, build, and deployment and infrastructure management
- Documented release timelines and defect handling expectations
- CVD practices for key downstream suppliers

Generic CVD Process



CVD for Closed vs. Open Source

1. Supported by some (corporate) entity, updates come ONLY from Supplier
2. Governed by stakeholders, regulators, and customer demands
3. Security reports and communications typically not done in public
    - Most certain to have restrictive embargos
4. Release of security updates typically tied to some predetermined release schedule (generally synchronized to new product releases/updates
5. Updates typically well-orchestrated of communicated to Consumers

1. Supported by a community (probably not from one corporate entity) of possible global contributors, many contribute anonymously or using psuedonyms
2. Generally, little formal governance, no regulatory oversite, and responds to customer feature requests of personal interests
3. Near-universal open defect tracking processes, private methods to disclose available in more mature projects
    - Some projects work ALL issues in the open, with NO embargos
4. Updates released when they are ready (minutes-to-days typically)
5. Updates typically only notified through public source-code repo commit, project blog, or notice to user mailing list

03

The OSS Threat Model

OSS Threat Model

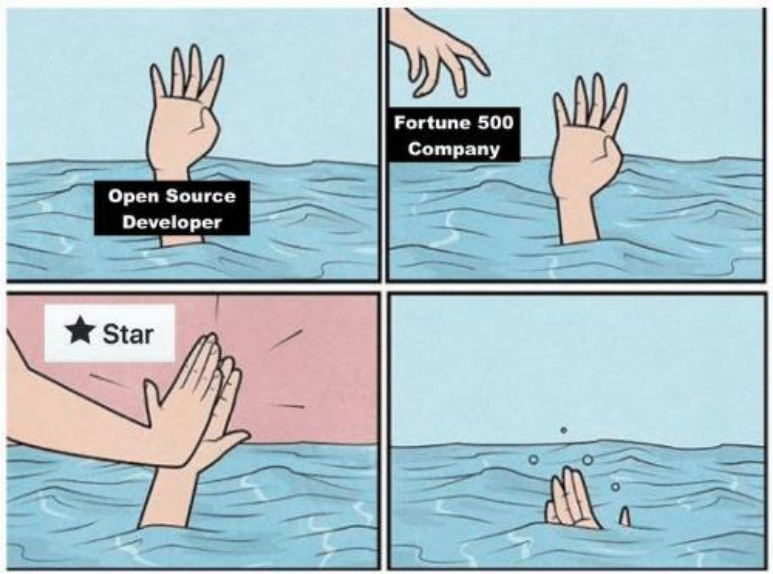
With the global nature of open source software, anyone from anywhere can, and often does, contribute code

[Image Source]

list of Threats

ibutors
em resolution times
em notifications
l" attacks
-style name confusion
cture attacks

How the many of Suppliers & Consumers "support" OSS



[Image Source]

How Suppliers can work better with OSS

- Understand how the upstream software they are ingesting and supplying downstream works
- Understanding how are issues worked upstream
- Understanding how are security issues handled upstream
- Understanding how are patches released by project
- Understanding how are security patches identified (are advisories published?)
- Understanding how can the proper patch be acquired, tested, and staged on a safe network (*aka* pulling straight from the internet can be a suboptimal idea)

Predicatable real-world example

CVE-2021-44228
10Dec2021
CVSS 10.0
CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H (ZOMG!) Apache Log4j2 2.0-beta9 through 2.15.0 (excluding security releases 2.12.2, 2.12.3, and 2.3.1) JNDI features used in configuration, log messages, and parameters do not protect against **attacker controlled LDAP and other JNDI related endpoints**. An attacker who can control log messages or log message parameters can execute arbitrary code loaded from LDAP servers when message lookup substitution is enabled.

CVE-2021-45046
14Dec2021
CVSS 9.0
CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:C/C:H/I:H/A:H (ZOMG!)
It was found that the fix to address CVE-2021-44228 in Apache Log4j 2.15.0 was incomplete **in certain non-default configurations.** This could allows attackers with control over Thread Context Map (MDC) input data when the logging configuration uses a non-default Pattern Layout with either a Context Lookup (for example, $${ctx:loginId}) or a Thread Context Map pattern (%X, %mdc, or %MDC) to craft malicious input data using a JNDI Lookup pattern resulting in an information leak and remote code execution in some environments and local code execution in all environments. Log4j 2.16.0 (Java 8) and 2.12.2 (Java 7) fix this issue by removing support for message lookup patterns and disabling JNDI functionality by default.

CVE-2021-45015
18Dec2021
CVSS 5.9
CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:U/C:N/I:N/A:H (Meh.)
Apache Log4j2 versions 2.0-alpha1 through 2.16.0 (excluding 2.12.3 and 2.3.1) did not protect from uncontrolled recursion from self-referential lookups. This allows **an attacker with control over Thread Context Map data to cause a denial of service when a crafted string is interpreted**. This issue was fixed in Log4j 2.17.0, 2.12.3, and 2.3.1.

CVE-2021-44832
28Dec2021
CVSS 6.6
CVSS:3.1/AV:N/AC:H/PR:H/UI:N/S:U/C:H/I:H/A:H (Meh.)
Apache Log4j2 versions 2.0-beta7 through 2.17.0 (excluding security fix releases 2.3.2 and 2.12.4) are vulnerable to a remote code execution (RCE) attack when a configuration uses a JDBC Appender with a JNDI LDAP data source URI **when an attacker has control of the target LDAP server**. This issue is fixed by limiting JNDI data source names to the java protocol in Log4j2 versions 2.17.1, 2.12.4, and 2.3.2.

To Post-mortem a bit

1. Active exploitation reports lead to earlier-than-desired public disclosure - #SAD!
2. Apache Security team has CVE published ***within hours*** of PD and patches available to downstream – #WOOT!!
3. First patch found not sufficient to close out vuln in **NON-DEFAULT configs** (i.e. someone would have to intentionally change the developers' defaults) - #OOPS, SORRY!
4. Orgs working on testing/rolling out initial patch must restart integration efforts & testing - #MEH.
5. During incident, additional review finds ***other bugs*** that subsequently **are fixed** - #COOL!
6. Downstream realizes that **Log4Jv1 was publicly deprecated ~5years previous**, yet they still are supplying/supporting it with no patches forthcoming from upstream - #OUCH!
7. Downstream Suppliers of Log4J start rolling their advisories and patches out 1-30+ days after initial PD
8. End-consumers need to evaluate and deploy the upstream patches through out their fleets

[Image Source]

04

How to fix YOUR OSS “Problem”

Ask not what the open source can do for you, ask what YOU can do for the open source

[Image Source]

[Image Source]

"The BEST way to make Open Source better.....

....is to participate and contribute back."

–SOMEONE FAMOUS

Suggestions to “fix” open source

- Contribute to and participate in communities of software that you are providing through your products
- Contribution can take many forms, but the BEST contribution is active developer participation in their project (help the community fix THEIR problems while getting attention to your needs)
- Build trust and respect through transparent engagement
- Support industry groups like OpenSSF, OWASP, FIRST, and others that are actively working to address concerns like yours
- KNOW what oss packages, libraries, dependencies YOU are baking into your products and have a plan how to handle functional and security defects in them.
- Be prepared to pivot from using a component if that community “dies”, moves on, or can not meet YOUR needs
- Aggressively monitor upstream sources for changes, issues, PRs, mailing lists, etc.

[Image Source]

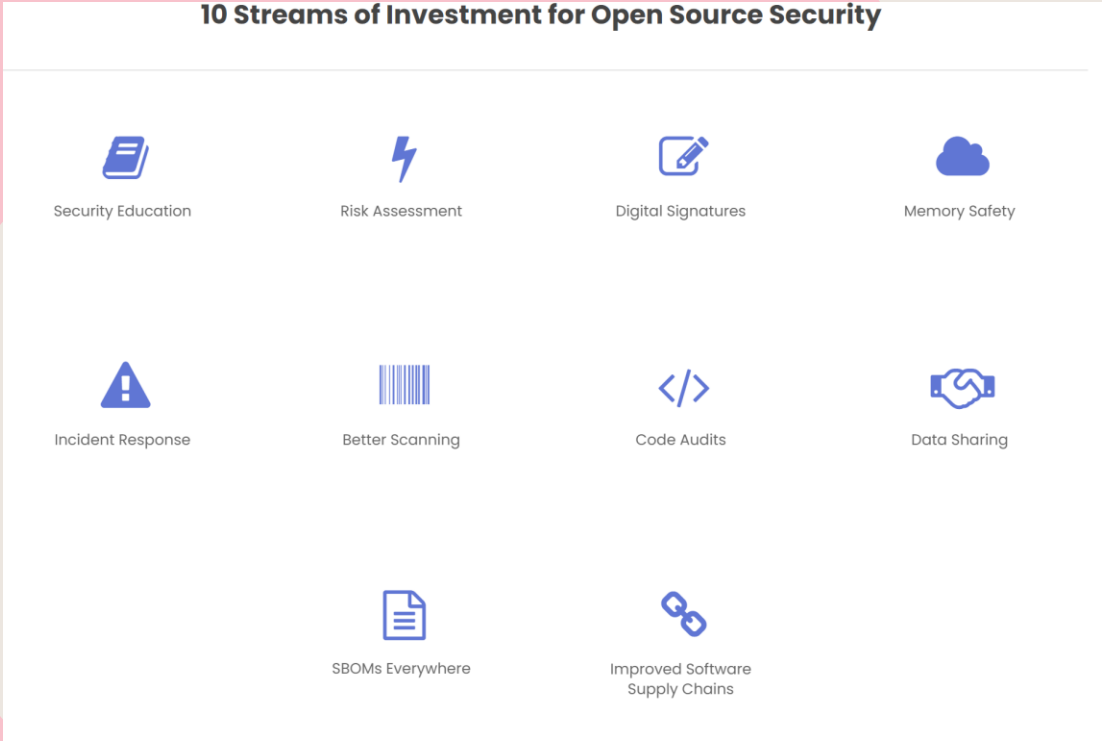
MOAR Suggestions

- Unless you are actively participating in it, stop complaining about “THE open source”
- There is no monolithic THE open source. Each project has unique goals, objectives, culture, and behaviours.
- There is no CEO of **the** OSS. No one group OWNS the open source. OSS is made up of individuals, groups, companies, all collaborating towards shared goals, but one project will not have the same people, goals, or processes as another. Each group needs to be approached uniquely

[Image Source]

OSS does not need more people standing around pointing out problems and not contributing to solutions

Someone IS doing something! (and you can help out too!)



https://openssf.org/oss-security-mobilization-plan/

By engaging OSS the RIGHT ways....

[Image Source]

MEOW!

DO YOU HAVE ANY QUESTIONS?

CRob_at_Intel_dot_com

@SecurityCRob

SecurityCRob

The Security Unhappy Hour

CREDITS: This presentation template was created by Slidesgo, including icons by Flaticon and infographics & images by Freepik