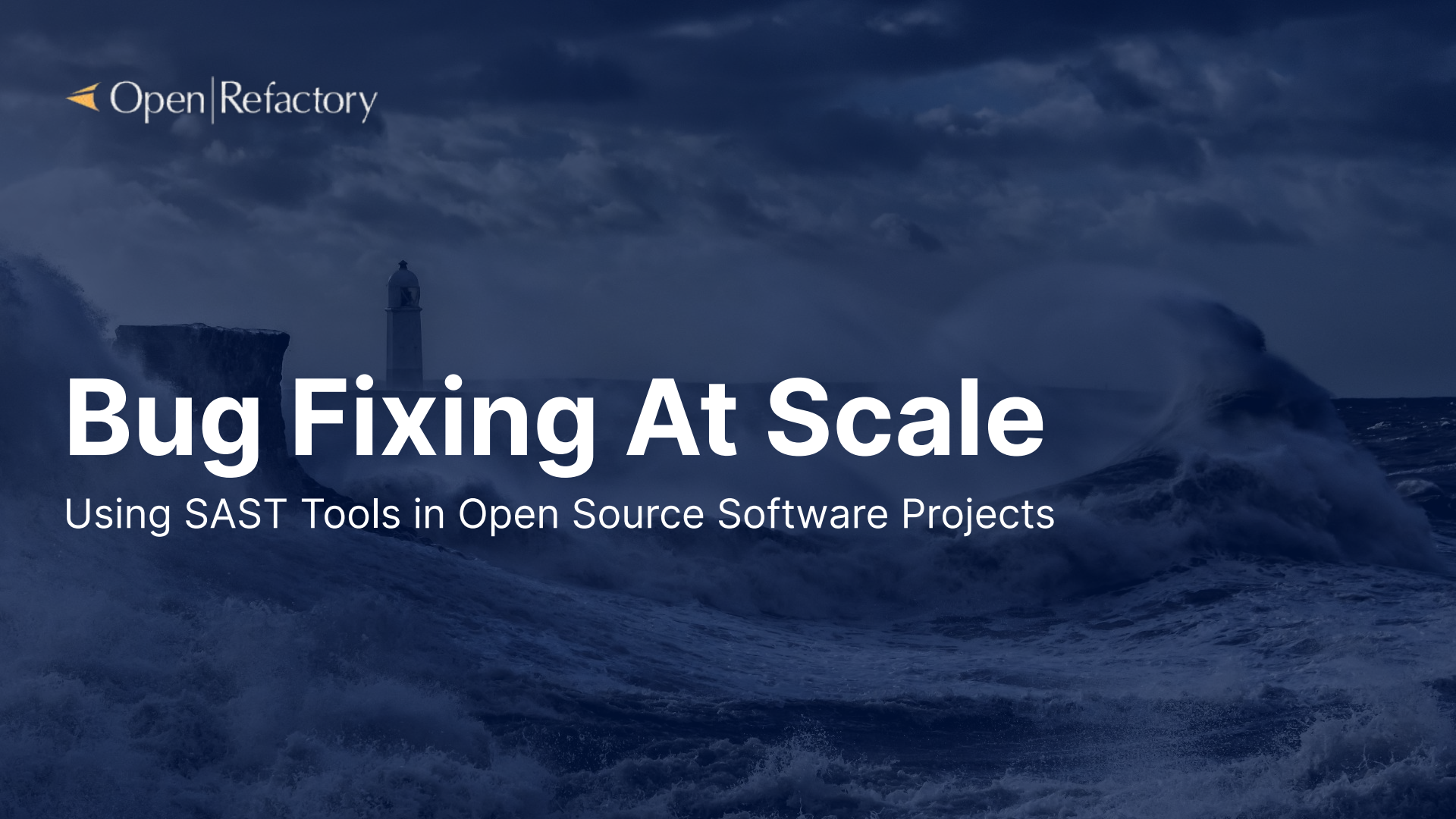

Open|Refactory
Bug Fixing At Scale
Using SAST Tools in Open Source Software Projects

Ataf Fazledin Ahamed

Secure Software Developer
@ OpenRefactory

Contents

- Overview of the Alpha Omega Project
- Collaboration with Open Source Security Foundation
- How We Are Fixing Bugs At Scale
- Don’t Eat The Pickle: Example of A Real Life Vulnerability
- Lessons Learned From Fixing Bugs

What is SAST ?

What is SAST ?

- Static Application Security Testing (SAST)
  - Runs on source code of a program
  - Example will be shown later

- Dynamic Application Security Testing (DAST)
  - Runs on live program
  - Example: BurpSuite, Nikto, ZAP, etc.

Why SAST ?

- Open Source Softwares are huge in numbers
  - NPM: 2M
  - Maven: 260K+
  - GitHub: 52M (2022)
- SAST tools can be easily automated
- Can be used in the early stages of development
- White box testing
  - Dependency problem or program bug

What is Alpha Omega?

Alpha Omega Project

Initiative taken by the **Open Source Security Foundation** (OpenSSF/OSSF)

December 10, 2021

- Log4Shell Exploit

January 13, 2022

- White House Meeting called by **President Joe Biden**
- Google, Microsoft, LFX, OpenSSF, etc.

- https://github.com/ossf/alpha-omega/

Alpha Omega Project

Initiative taken by the **Open Source Security Foundation** (OpenSSF/OSSF)

Alpha

- Node.js
- Eclipse Foundation
- Rust Foundation
- jQuery
- Python Software Foundation

Omega

- Identify security issues in the top 10,000 (or more) most-critical open source projects

- https://github.com/ossf/alpha-omega/

Omega Analyzer

- Collection of 30 tools (Open & Closed Source)

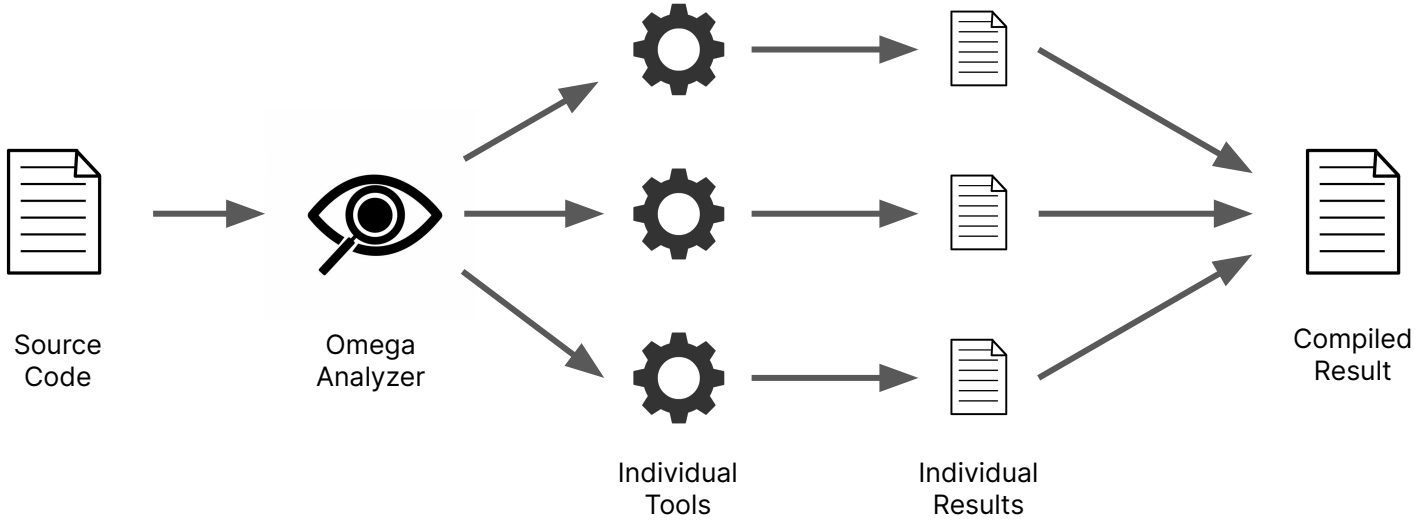


Fig: Working process of the Omega Analyzer

Tools Used in Omega Analyzer

- DevSkim
- NodeJsScan
- CppCheck
- Radare2
- CodeQL
- Lizard
- ShhGit
- Secret Scanner
- Detect-Secrets
- SCC
- Brakeman
- Graudit
- Application Inspector
- Manalyze
- binwalk
- ClamAV
- Bandit
- Semgrep
- Yara
- tbv
- ILSpy
- strace
- OSS Gadget
- npm audit
- Snyk Code

Tools Used in Omega Analyzer

- DevSkim
- NodeJsScan
- CppCheck
- Radare2
- CodeQL
- Lizard
- ShhGit
- Secret Scanner
- Detect-Secrets
- SCC
- Brakeman
- Graudit
- Application Inspector
- Manalyze
- binwalk
- ClamAV
- Bandit
- Semgrep
- Yara
- tbv
- ILSpy
- strace
- OSS Gadget
- npm audit
- **Snyk Code**

*** Closed Source/Proprietary**

Other SAST Tools

OpenRefactory

SonarSource

JetBrains (New)

Collaboration with OpenSSF

Collaboration with OpenSSF

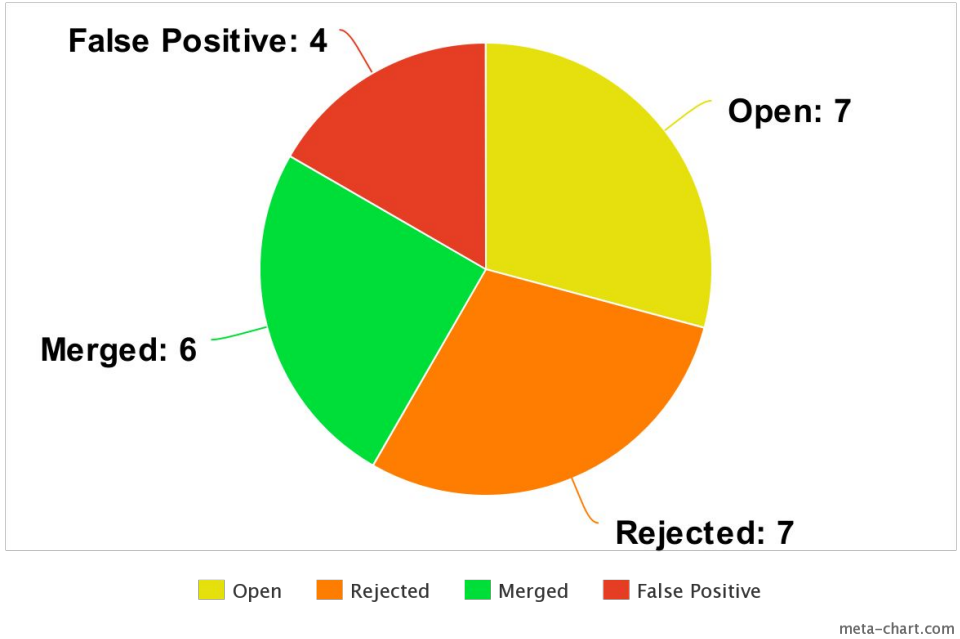
Phase 1 | October 2022

Total Reported Bugs: 24

Only 25% of the bug fixes got merged.

Important Projects

nltk, pandas



Collaboration with OpenSSF

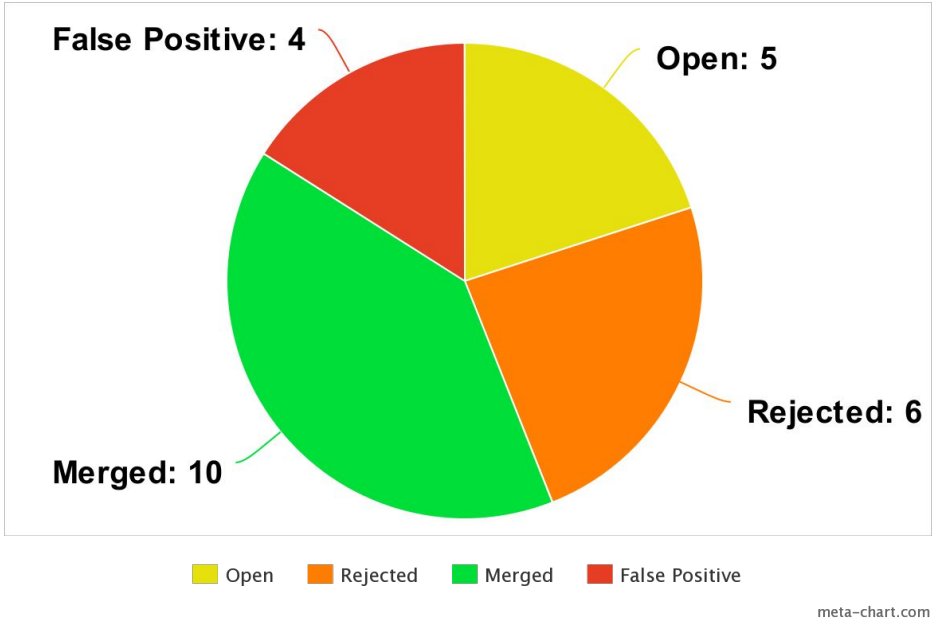
Phase 2 | December 2022

Total Reported Bugs: 25

Only 40% of the bug fixes got merged.

Important Projects

PyTorch, edX, cinder



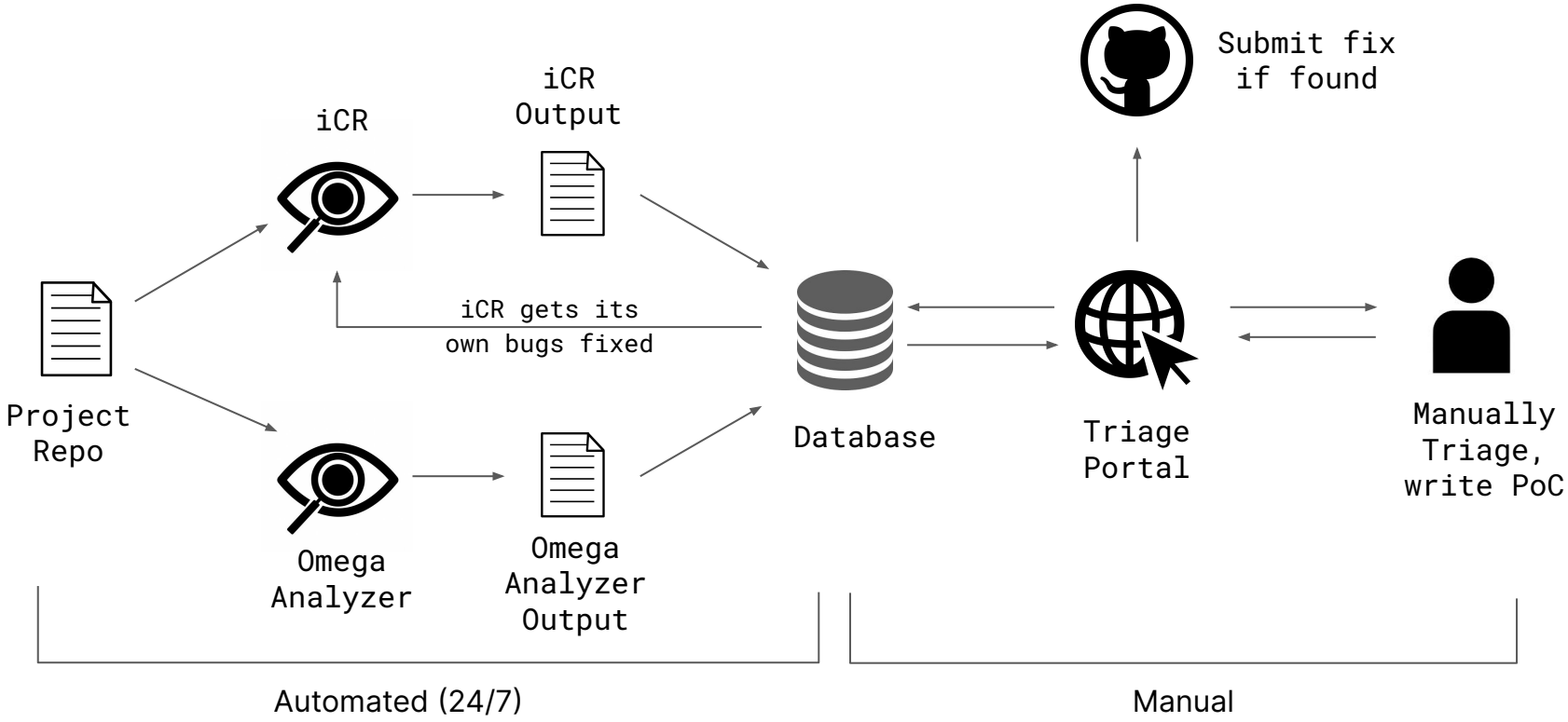
Collaboration with OpenSSF

- Scope & Coverage
  - Open Source **Java** & **Python** Projects
  - Bugs identified by **iCR** & **Omega Analyzer**

- KPIs
  - # of projects scanned
  - # of bugs reported
  - # of bugs fixed
  - # of security issues fixed
  - List of projects with no bugs found

How We Are Fixing Bugs At Scale

How We Are Fixing Bugs At Scale

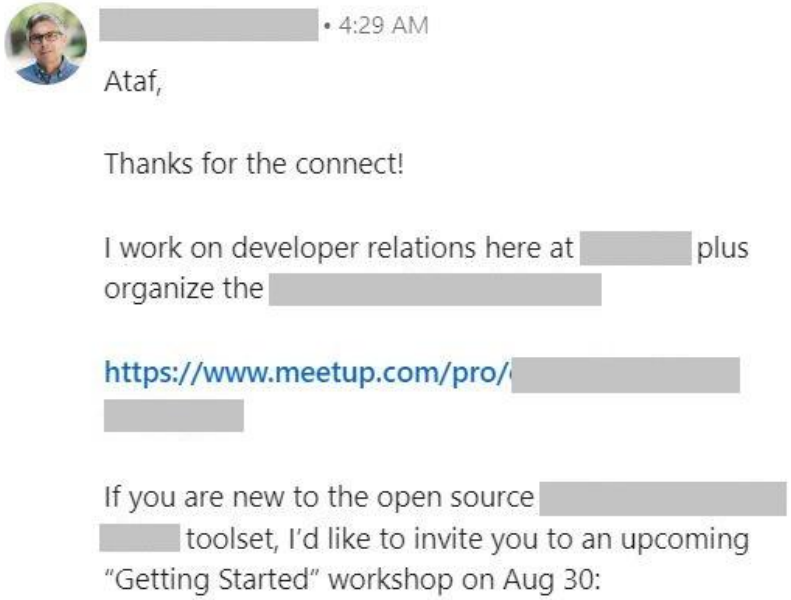


How We Are Fixing Bugs At Scale

July 25, 2023:	Project was scanned using iCR
Aug 3, 2023:	Triaged, Found a vulnerability Wrote PoC, Recorded a video
Aug 7, 2023:	Submitted a GitHub Issue with PoC
	No feedback for around 10 days
Aug 16, 2023:	...

How We Are Fixing Bugs At Scale

Aug 16, 2023: I get a LinkedIn Message



• 4:29 AM

Ataf,

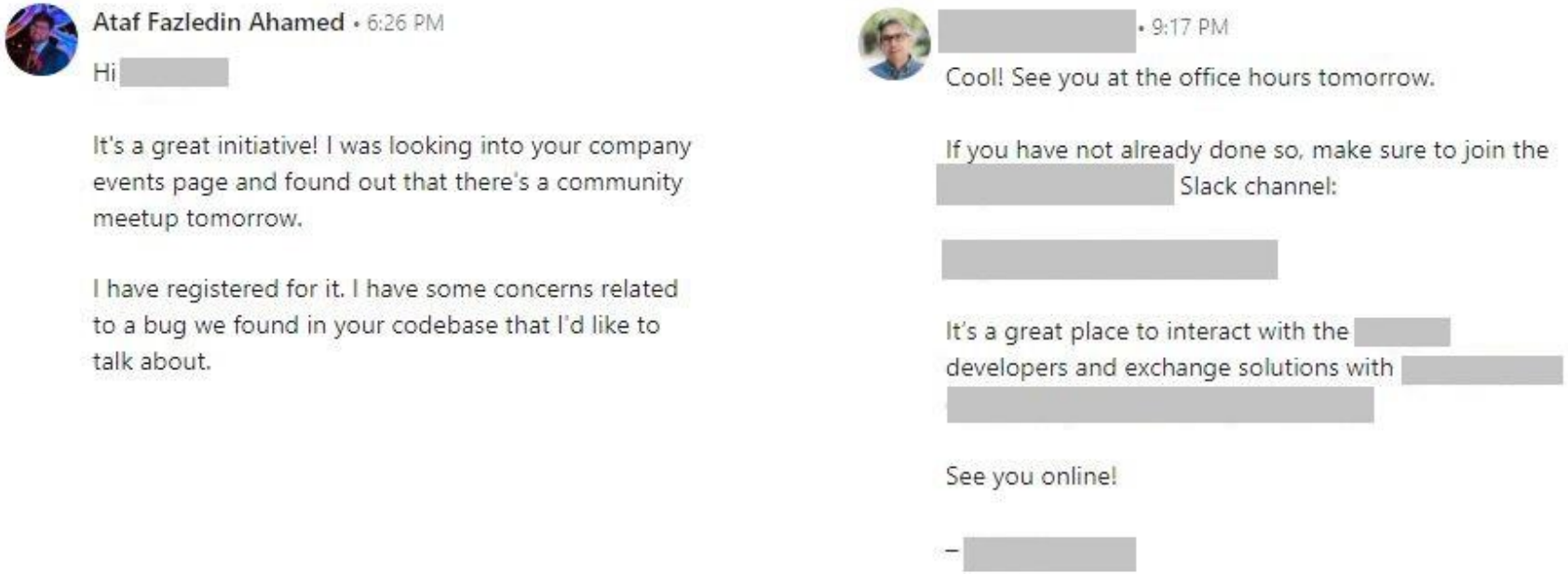
Thanks for the connect!

I work on developer relations here at [redacted] plus organize the [redacted]

https://www.meetup.com/pro/[redacted]

If you are new to the open source [redacted] toolset, I'd like to invite you to an upcoming "Getting Started" workshop on Aug 30:

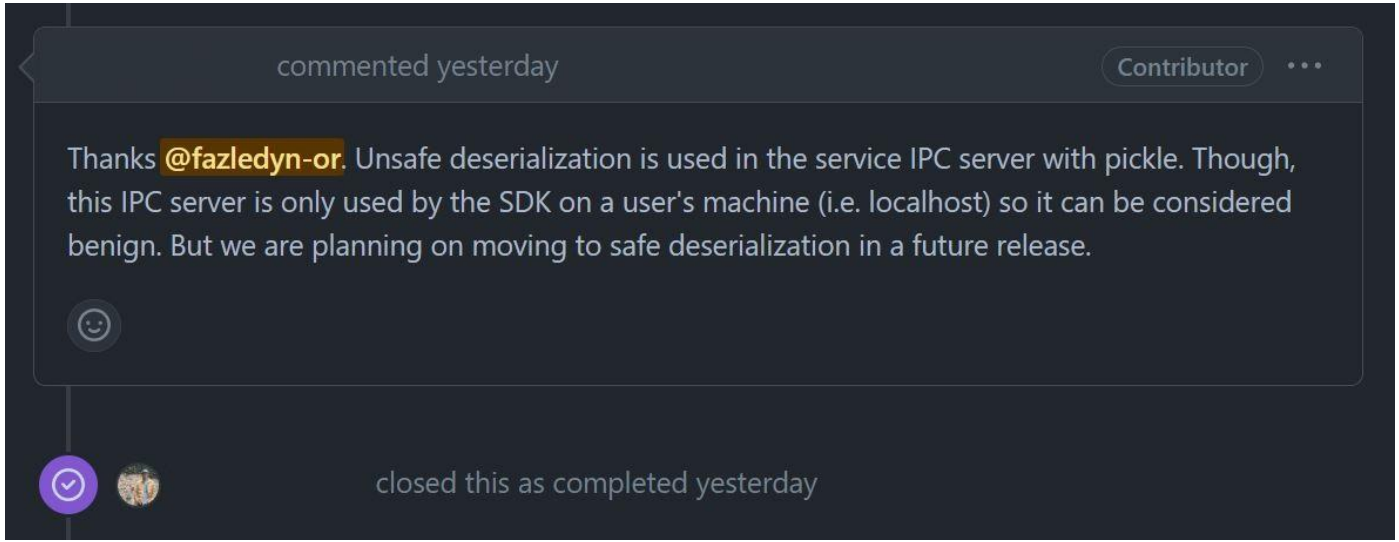
How We Are Fixing Bugs At Scale



How We Are Fixing Bugs At Scale

Aug 17, 2023: I joined the community meetup and informed them about the issue.

Aug 18, 2023:

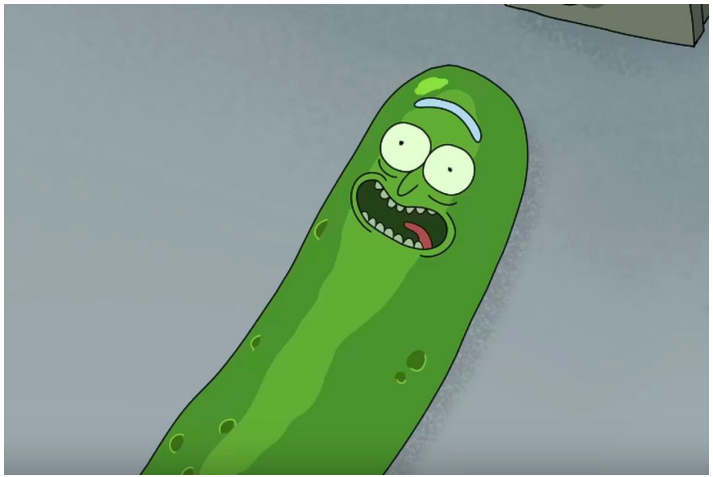


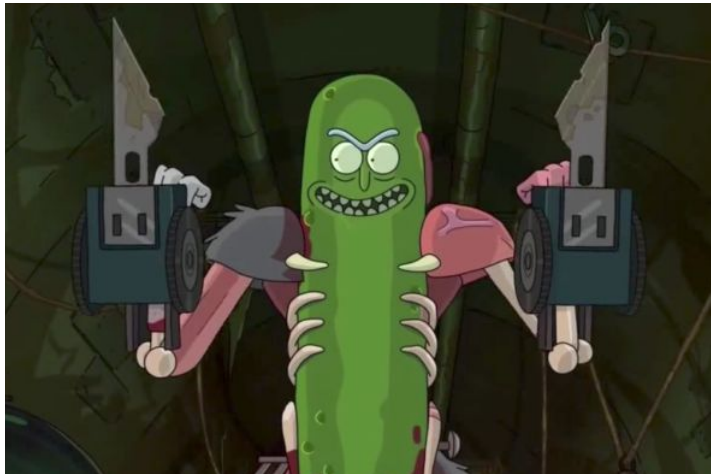
Don’t Eat The `Pickle`

Follow on-screen

Don’t Eat The `pickle`

- If care is not taken





Don’t Eat The `pickle`

- How to prevent?
  - Only unpickle the **data you trust**
  - If not a burden, use JSON/YAML

- Use Cases Seen So Far:
  - Sending & Receiving complex data using `serversocket` library
  - Storing & Retrieving data to/from **Redis** cache

Don’t Eat The `pickle`

- This vulnerability is not something new
- More Info:
  - https://snyk.io/blog/guide-to-python-pickle/
  - https://medium.com/ochrona/python-pickle-is-notoriously-insecure-d6651f1974c9

Lessons Learned

Lessons Learned

- Some of the PRs were false warnings because it is hard for a person triaging the bug report to **understand the attack surface.**

- For Example (Remember *Don't Eat The Pickle* ?)
  - Sending & Receiving complex data using `serversocket` library
  - Storing & Retrieving data to/from **Redis** cache

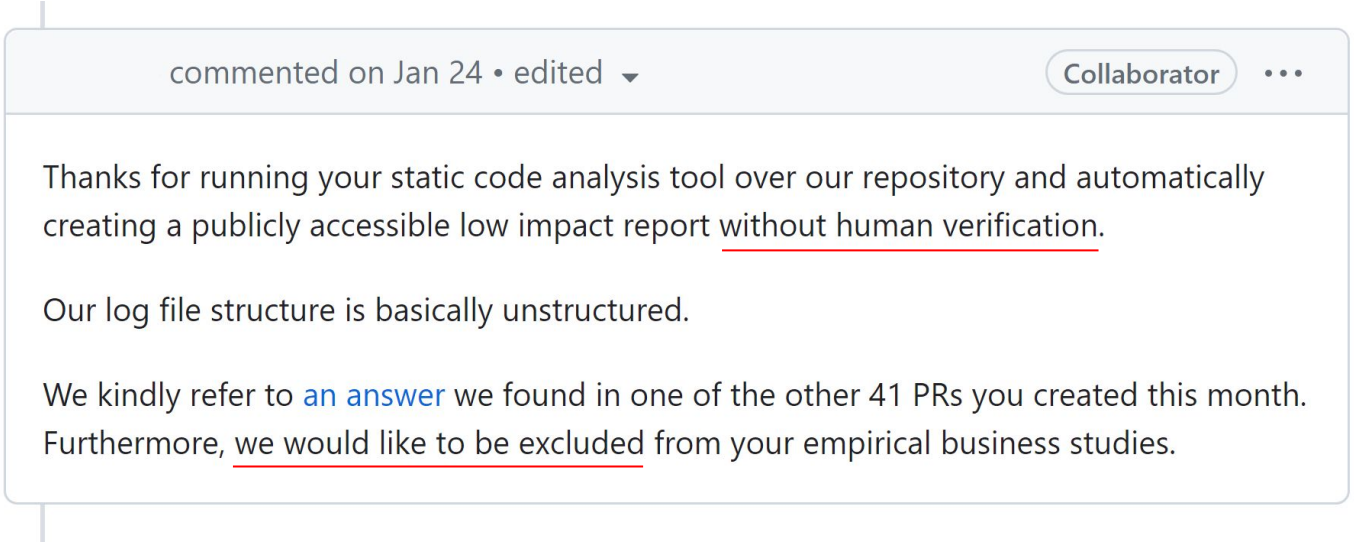
Lessons Learned

- Some of the PRs were false warnings because it is hard for a person triaging the bug report to **understand the attack surface.**

- For Example (Remember *Don't Eat The Pickle* ?)
  - ❌ Sending & Receiving complex data using `serversocket` library
  - ✅ Storing & Retrieving data to/from **Redis** cache

Lessons Learned

- PRs get rejected for different reasons, e.g- if the maintainer “thinks” that the **PR is coming from a bot**. This happens even for valid PRs.



Lessons Learned

- Different projects require **different bug reporting process**.

❖ Easy to fix, no prior knowledge needed *Example: Arithmetic, Logical Bug*	→ Pull Request (PR)
❖ Discussion needed to understand the usage	→ GitHub Issue
❖ Triaged, Possible Vulnerability *Example: Uses outdated version of a library*	→ PVR (Private Vuln. Reporting)

Lessons Learned

- PRs have their own lifetime. Many bugs **take a long time to fix** during which explicit hand-holding is needed.

❖ Phase 1 - October 2022
One PR got merged **after 2 months**

❖ Phase 2 - December 2022
3 months+ long collaboration was needed on a fix

Lessons Learned

- Increase Precision
  - We don’t want our tool to generate unnecessary fixes
  - Too much False Positives → Maintainers get irritated

- Increase Recall
  - We don’t want our tool to ignore real vulnerabilities
  - Compare results of multiple SASTs

Interested to Work With Us?

Work With Us

- We are interested to hire 2-3 apprentices
- Expected to work for around 4 weeks
- Based on the work quality, paid internships will be offered
- We work in a hybrid manner
- The apprentice role is **not paid** and completely **voluntary**

Work With Us

- If interested, apply here: https://forms.gle/3r67P79kgSUViksQ6

Any Questions ?

- If interested, apply here: https://forms.gle/3r67P79kgSUViksQ6