ARVOS

AI- and Risk-based Vulnerability Management
for Trustworthy Open Source Adoption

debricked

Emil Wåreus

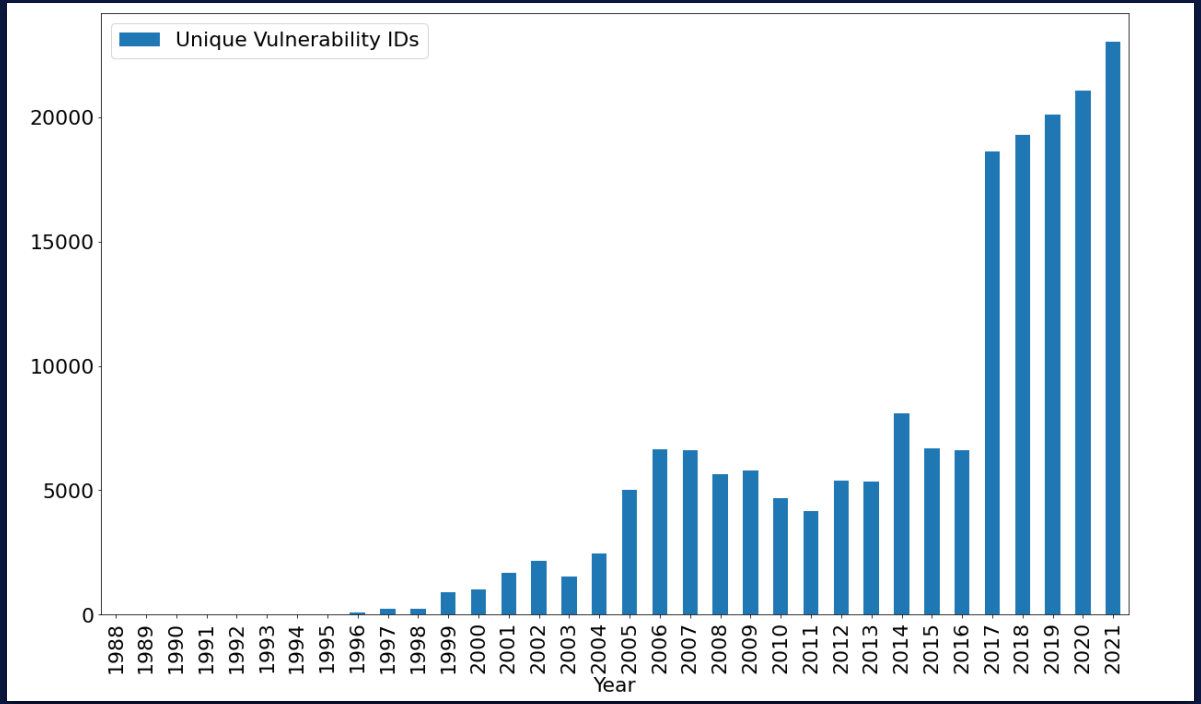
Co-Founder &
Head of Data Science

emil.wareus@debricked.com

Vulnerabilities in Open Source

More vulnerabilities discovered each year

More alerts and work required for developers

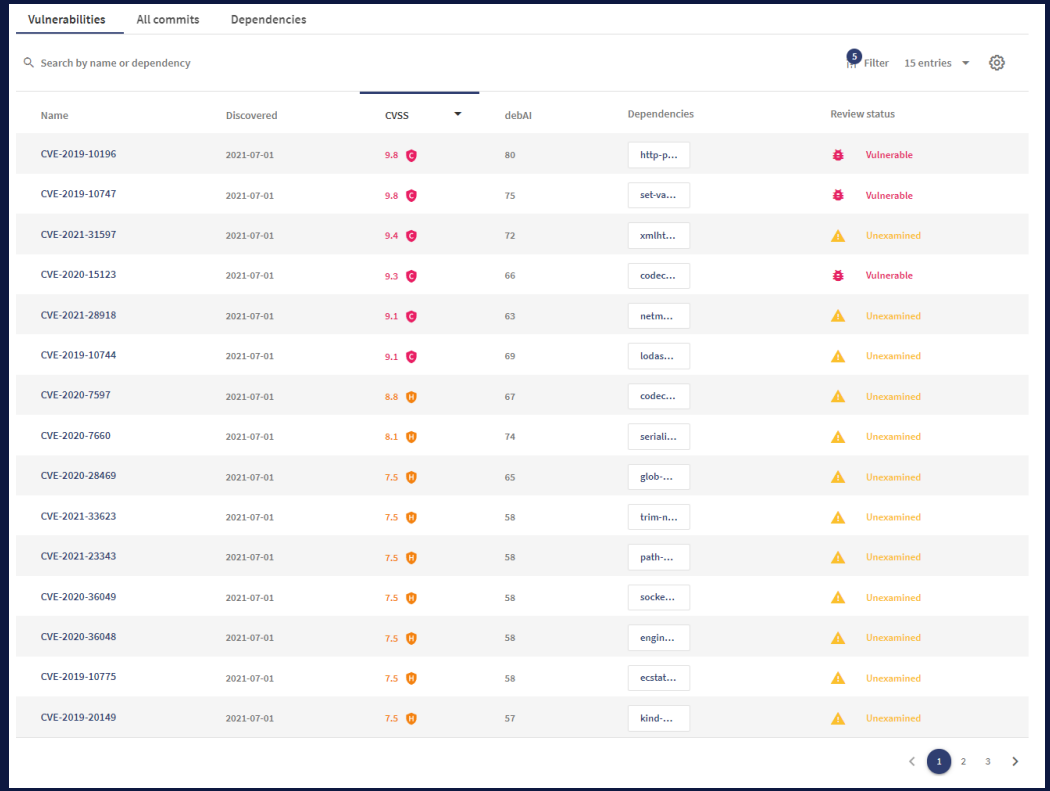


The “Cry Wolf” problem

Large lists of vulnerabilities to handle

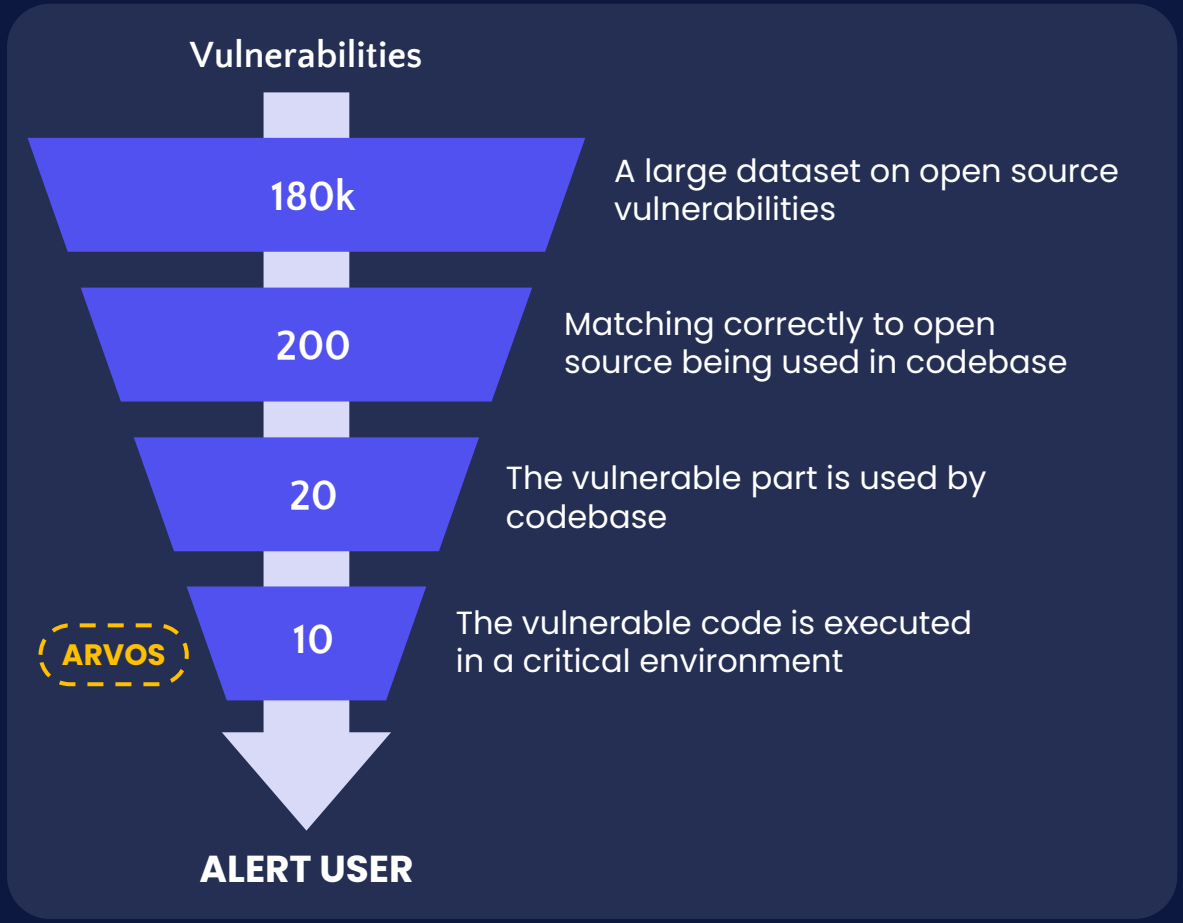
Rich information on the vulnerability itself

Poor contextualization to my code



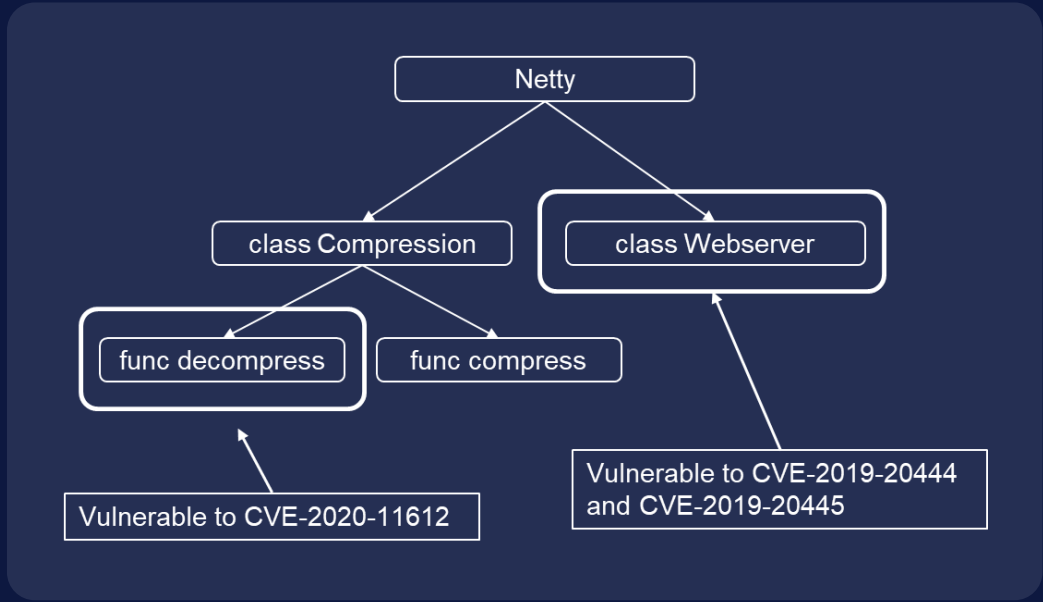
The 4 levels of precision

1. Are you using vulnerable OSS?
2. Are you calling the vulnerable part of the OSS?
3. Is the vulnerable part being called in a critical environment?
4. Is the vulnerability exploitable?

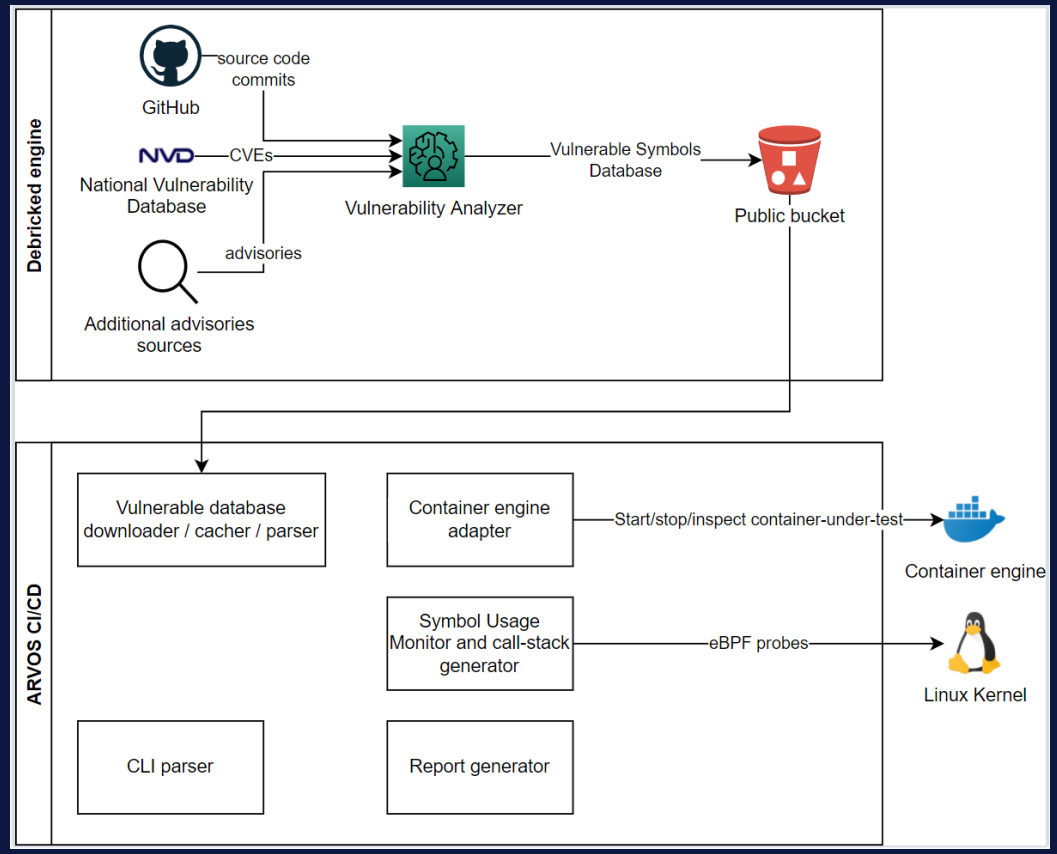


Finding the Vulnerable Functionality

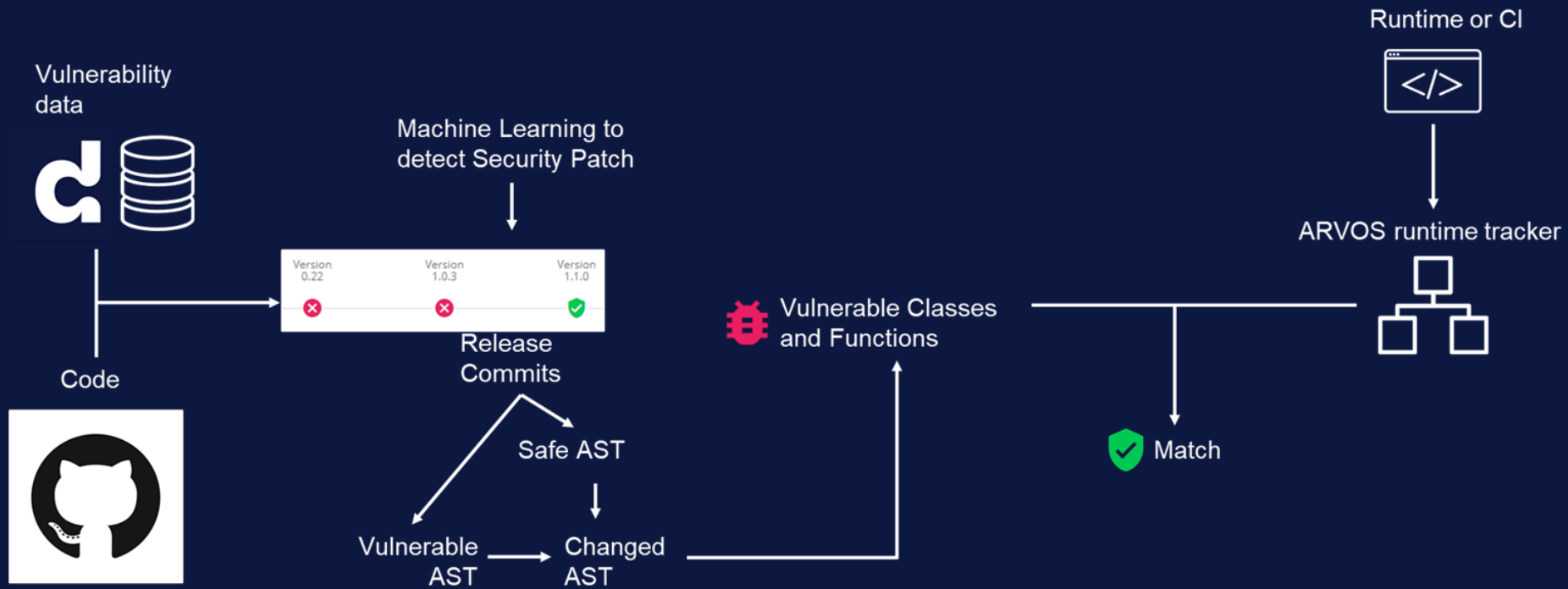
Only a part of the OSS project is affected by the affected vulnerability



ARVOS architecture



Finding the Vulnerable Functionality

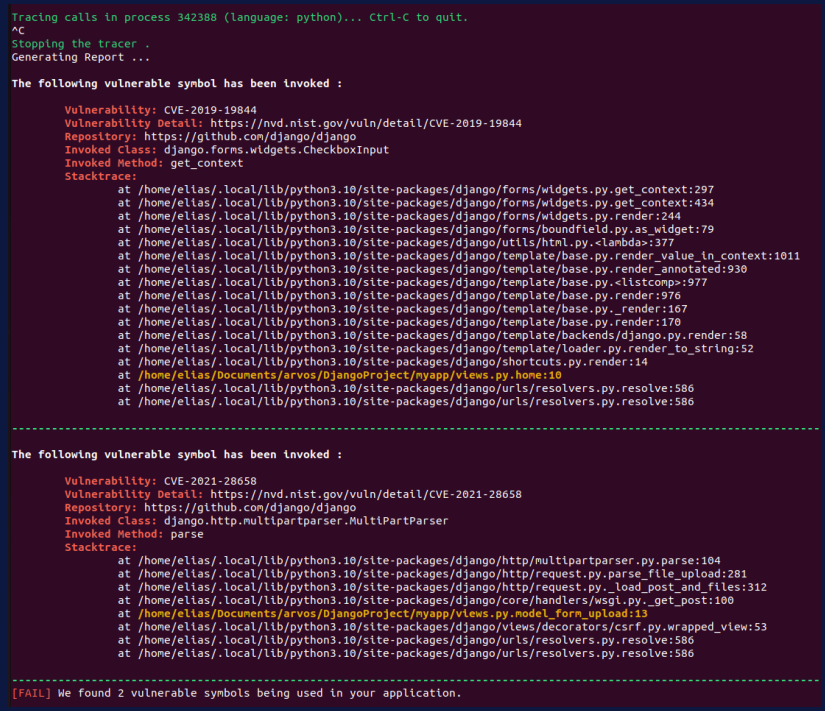


Tracing the Vulnerable Function Calls

Trace Vulnerable Code Execution in your CI

Get stack-trace reports, displaying how you reached the vulnerable code

Available as GitHub Action



Demo

https://github.com/arvos-dev/spring-vulnerable-app

Thank you!

emil.wareus@debricked.com

https://github.com/arvos-dev