Source Code Security Like a Pro - SAST, DAST, IAST, RASP, SCA, Reachability Analysis, EPSS, AST, CST, CFG, DFG, CPG & Hela Tool

By - Rohit Kumar (@rohitcoder)

Who am I

- Top 20 Security Researcher at Meta Bug Bounty for last 5 years
- Maintaining some open-source projects like Hawk-Eye and Hela
- Participated in Some Live Hacking Events by Meta
- I Code in Rust, Python, Javascript, Scala, Java, Whatever you can think off
- Source Code Security, Supply Chain Security & Web Security.
- Product Security Engineer @ Groww

 @rohitcoder  @rohitcoder  @rohitcoder

Shift Left vs Shift Right

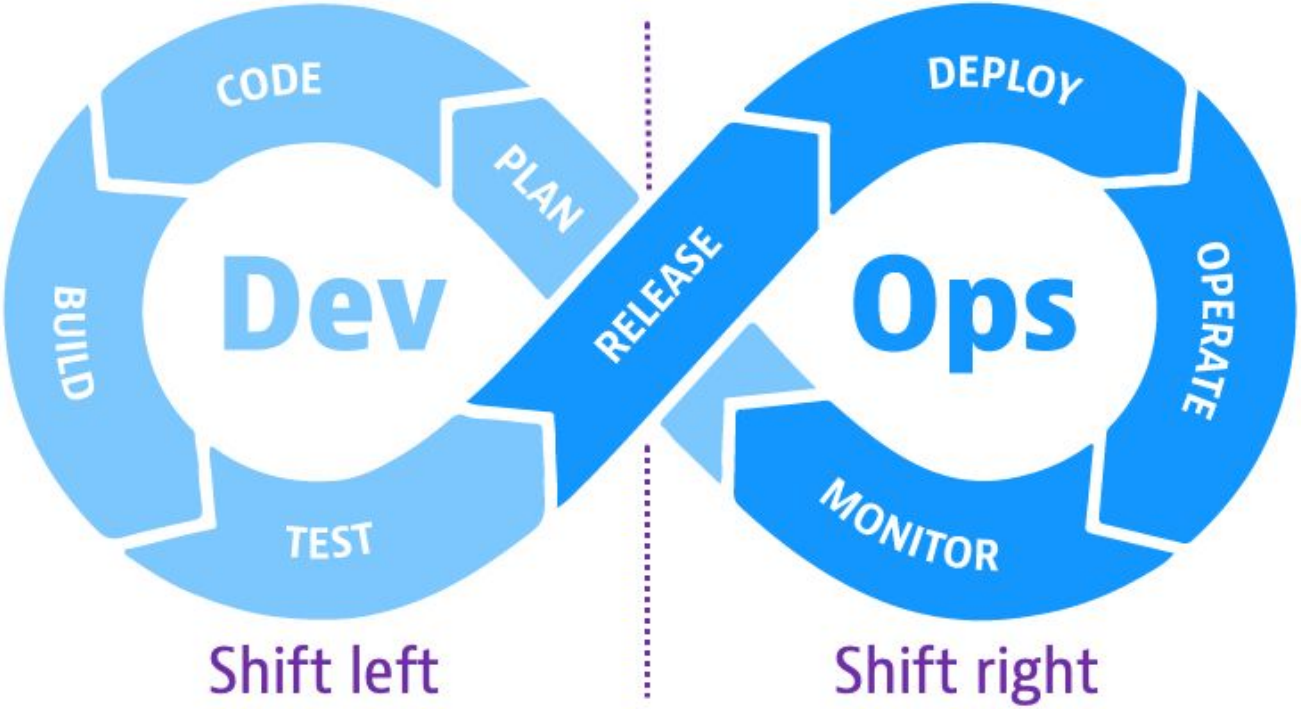


Image Source: Dynatrace

What's SAST and How it works?

Static Application Security Testing (SAST) looks for weaknesses in custom code (Written by your team/developers).

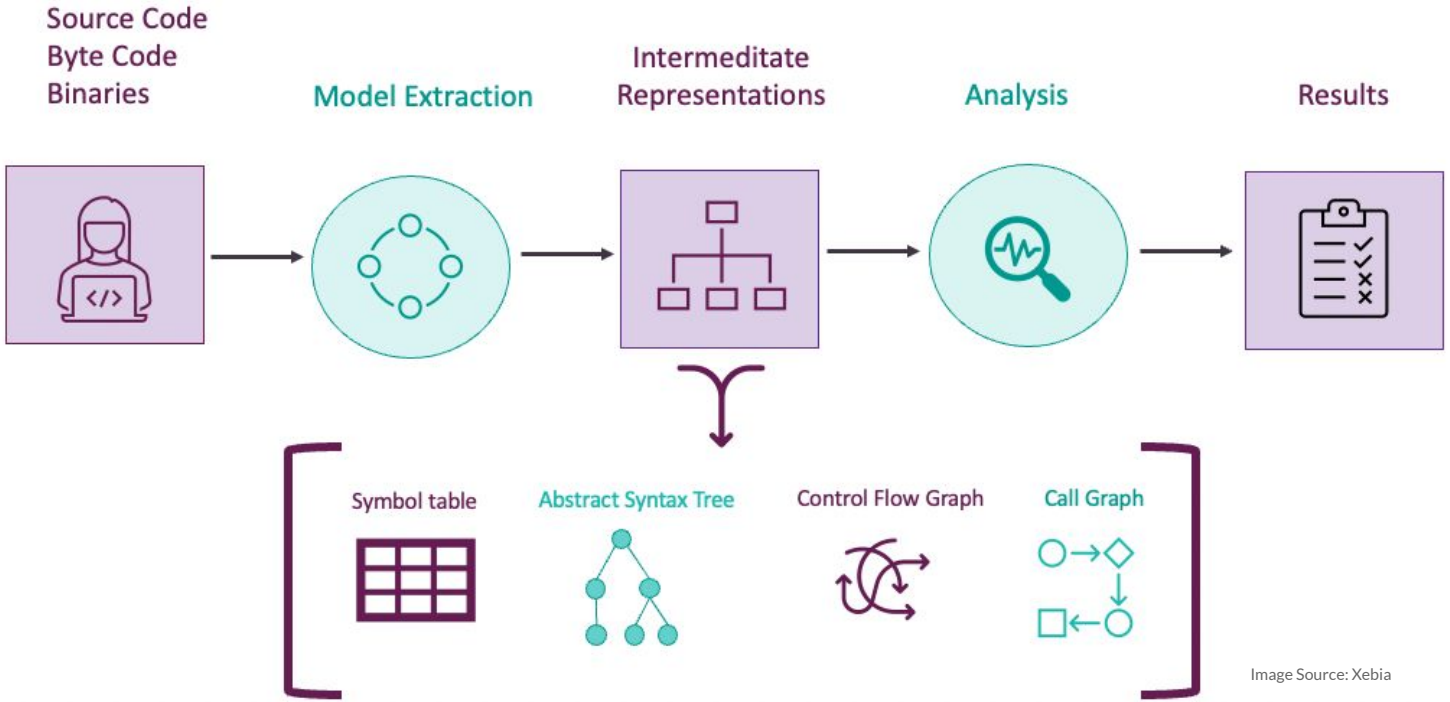


Image Source: Xebia

What's DAST and How it works?

Dynamic Application Security Testing (DAST) performs automated attacks on applications to test them for weaknesses when they are running

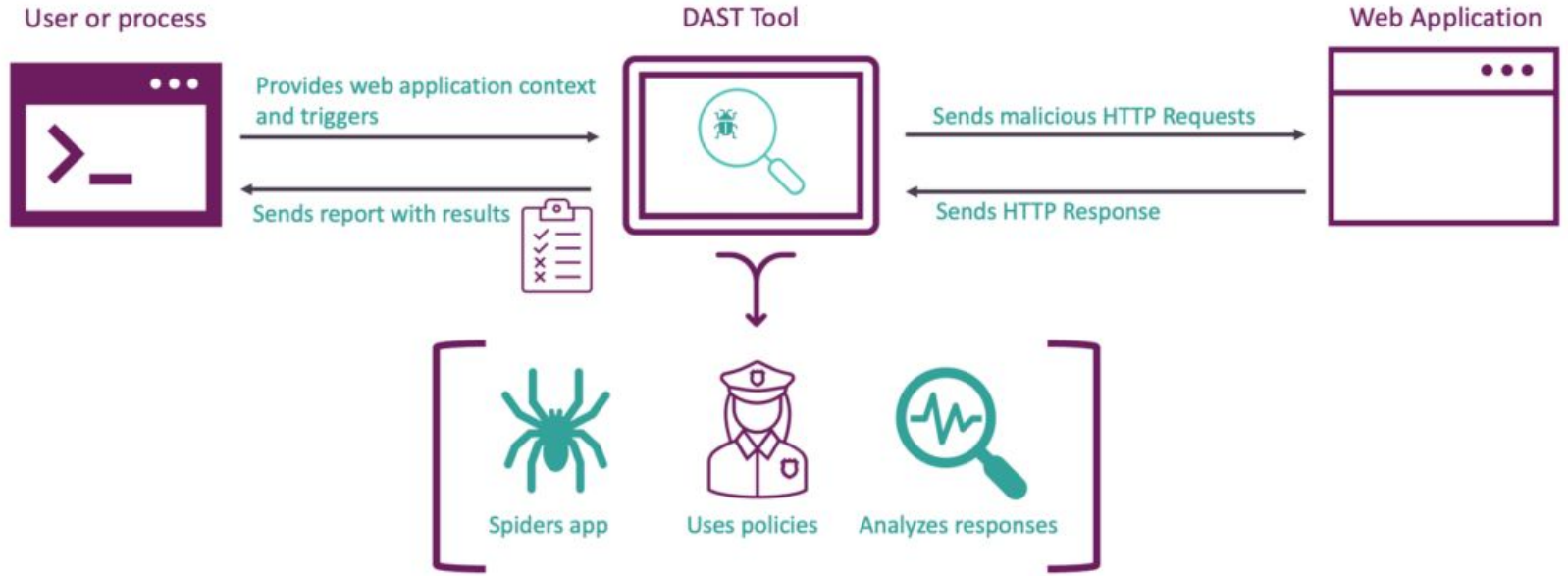


Image Source: Xebia

What's IAST and How it works?

Interactive Application Security Testing (IAST) combines DAST capabilities with SAST insights

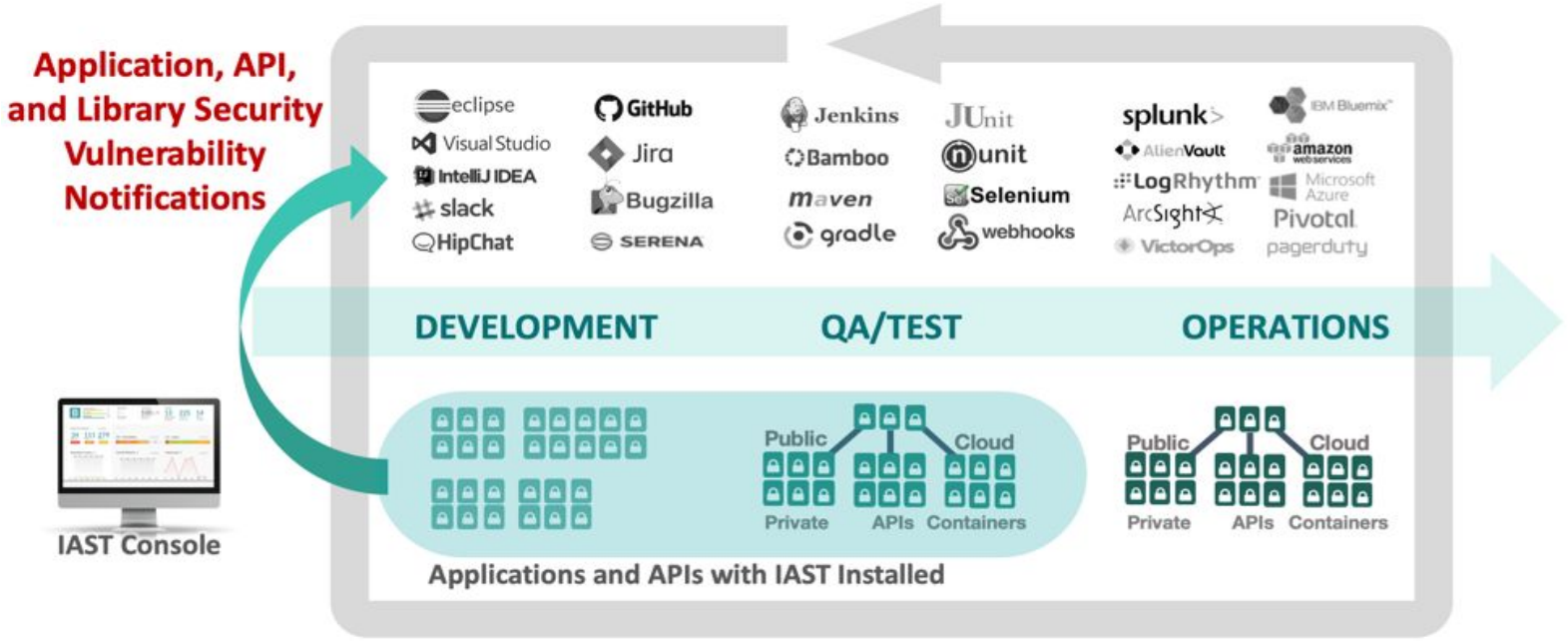
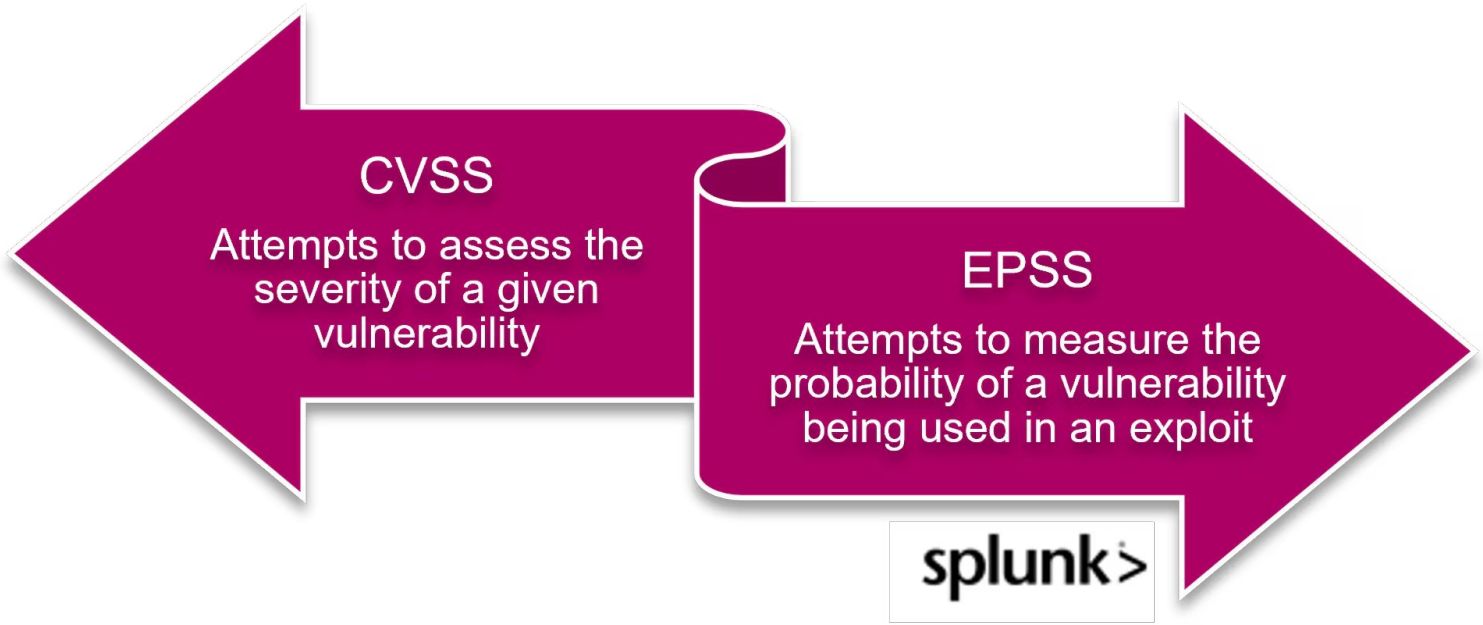


Image Source: DZone

CVSS vs EPSS, Which one to use?

Let's Use both?



What's RASP and How it works?

Runtime Application Self-Protection (RASP) is built into a program to protect it after deployment. It is capable of detecting and preventing external threats in real time.

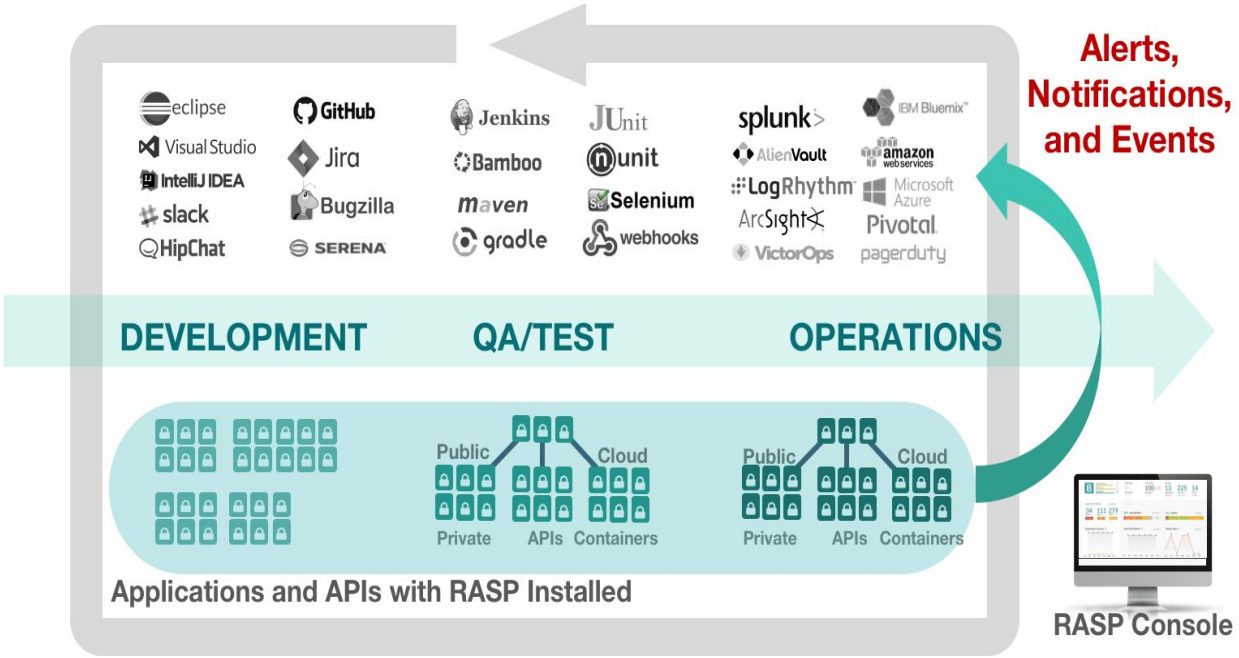


Image Source: DZone


SAST
DAST
IAST
Static
Dynamic
Interactive
Application Security Testing
Gartner

Application Security Testing

	Coverage	Low False Positives	Exploitability	Code Visibility	SDLC Integration	Broad Platform Support
SAST	✓	○	○	✓	✓	✓
DAST	○	✓	✓	○	○	✓
IAST	○	✓	✓	✓	✓	○

What's SCA and How it works?

SCA tools identify all open source packages in an application and all the known vulnerabilities of those packages. This knowledge can be used to notify developers of the issues in their code to fix them before they are exploited.

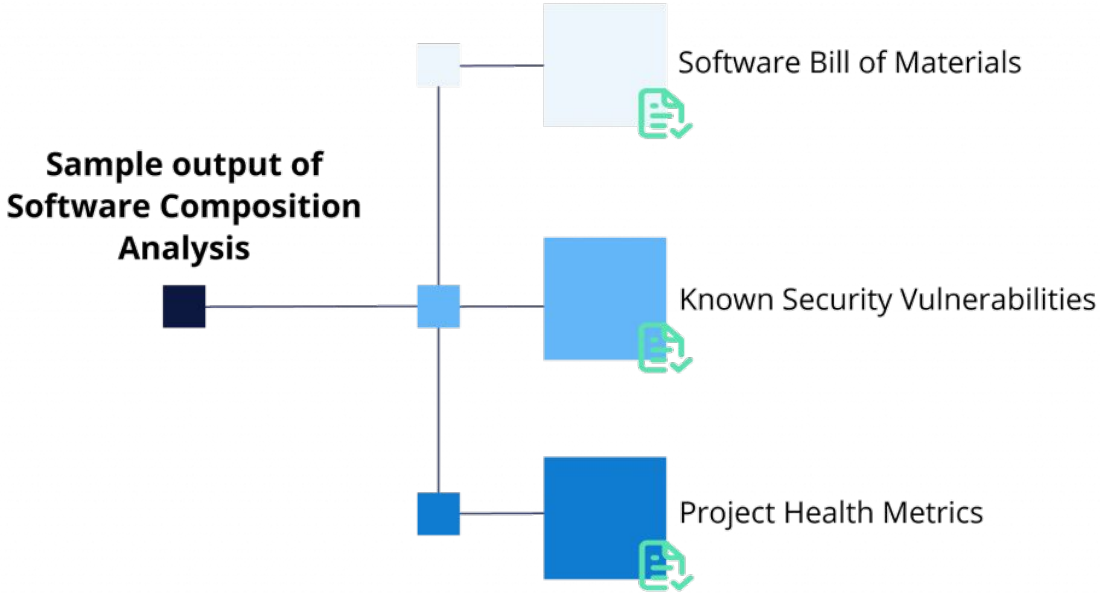
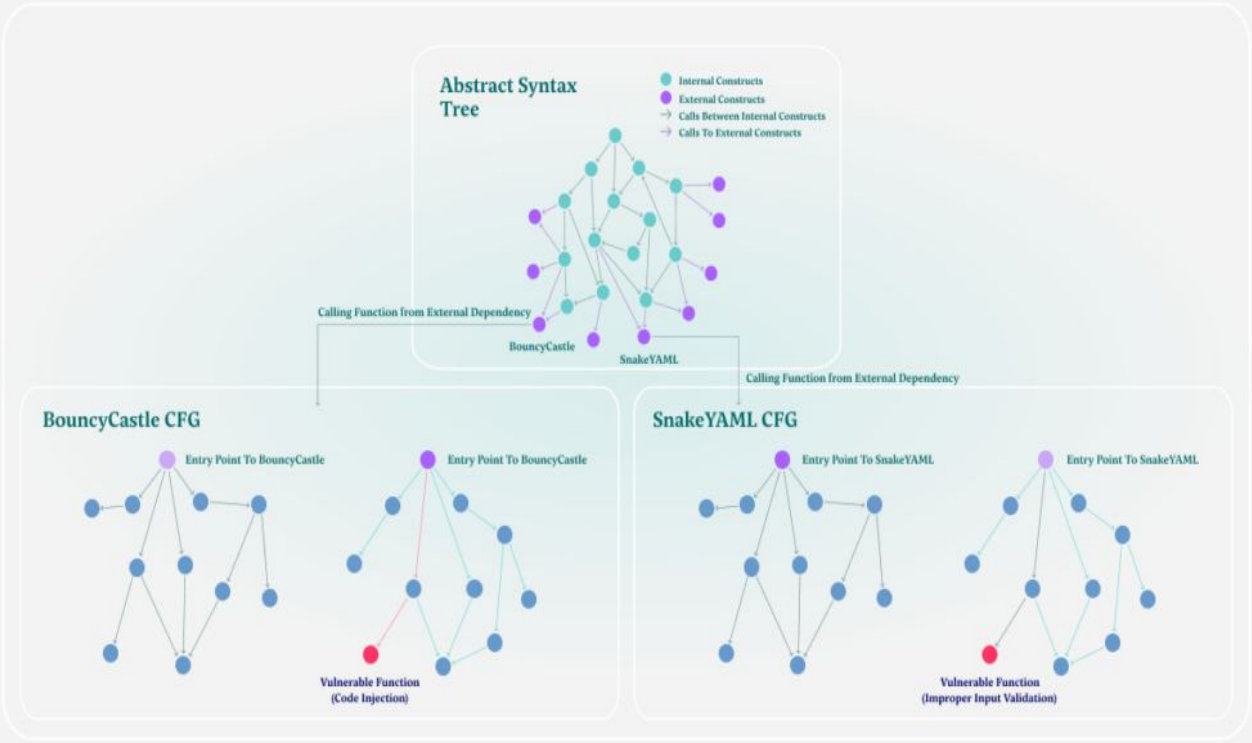


Image Source: DZone

What's Reachability analysis?



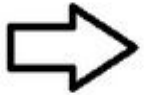
Reachability analysis in SCA checks if vulnerable functions in third-party libraries are invoked by your application, helping prioritize real security risks.

Image Source: Myrror Security

Source Code Representation - AST

Abstract syntax tree Represents the syntactic structure of code.

```
int min(int firstNumber, int secondNumber)
{
    if (firstNumber > secondNumber) {
        return secondNumber;
    }
    else {
        return firstNumber;
    }
}
```



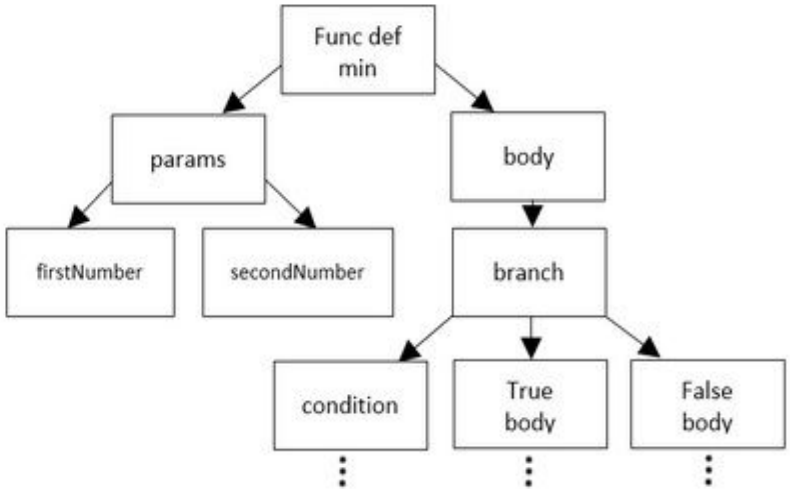
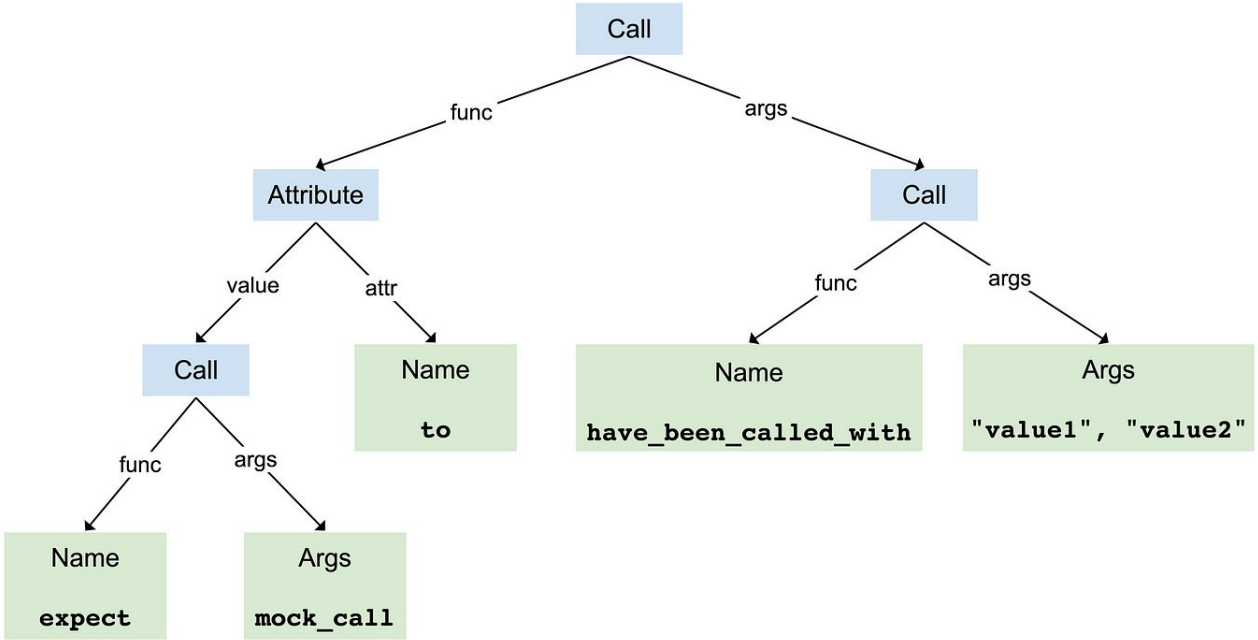


Image Source: ResearchGate

Source Code Representation - CST

CST (Concrete Syntax Tree): Represents the full syntactic structure of the code, including all tokens, closely reflecting the actual source code.



Source Code Representation - CFG

Control Flow Graph Represents the flow of control between statements or instructions.

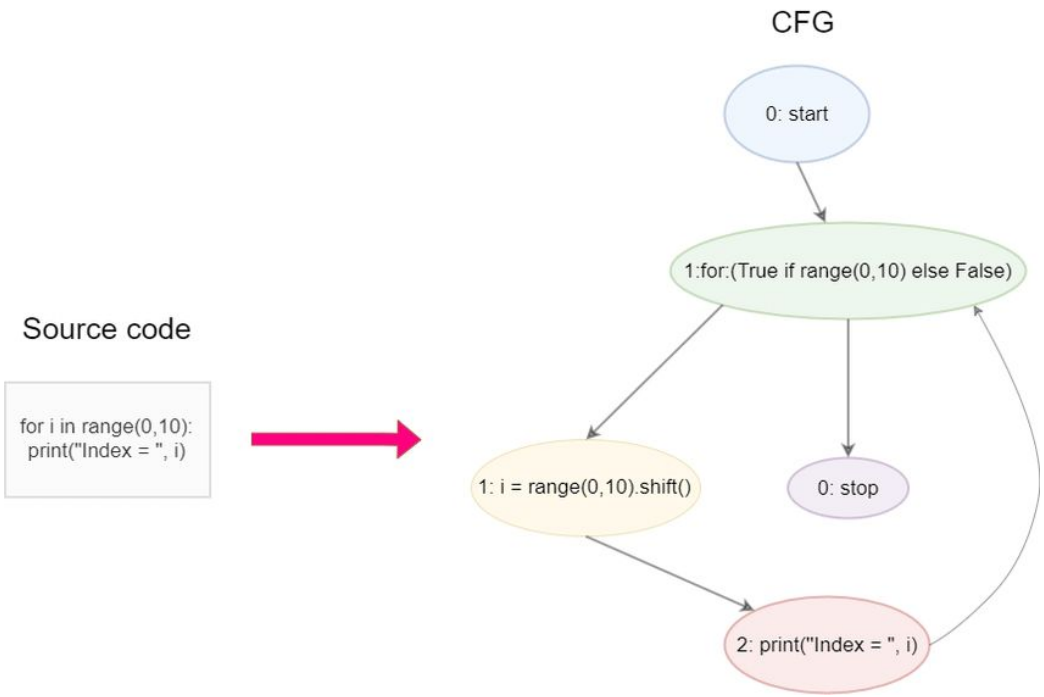
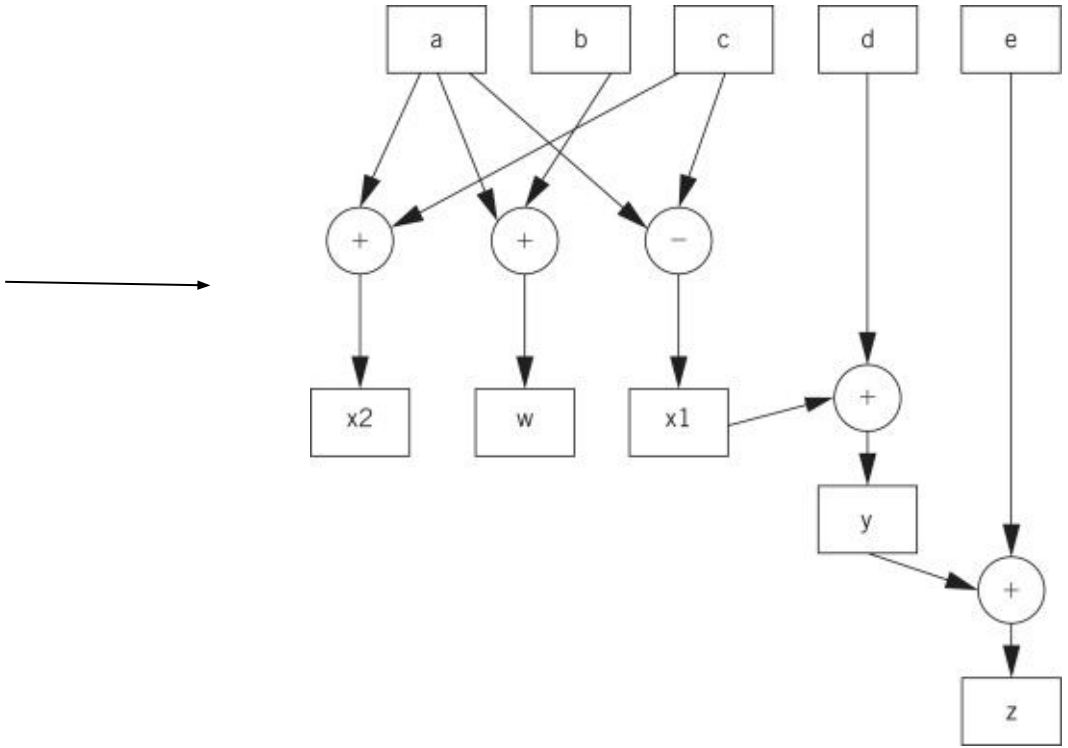


Image Source: ResearchGate

Source Code Representation - DFG

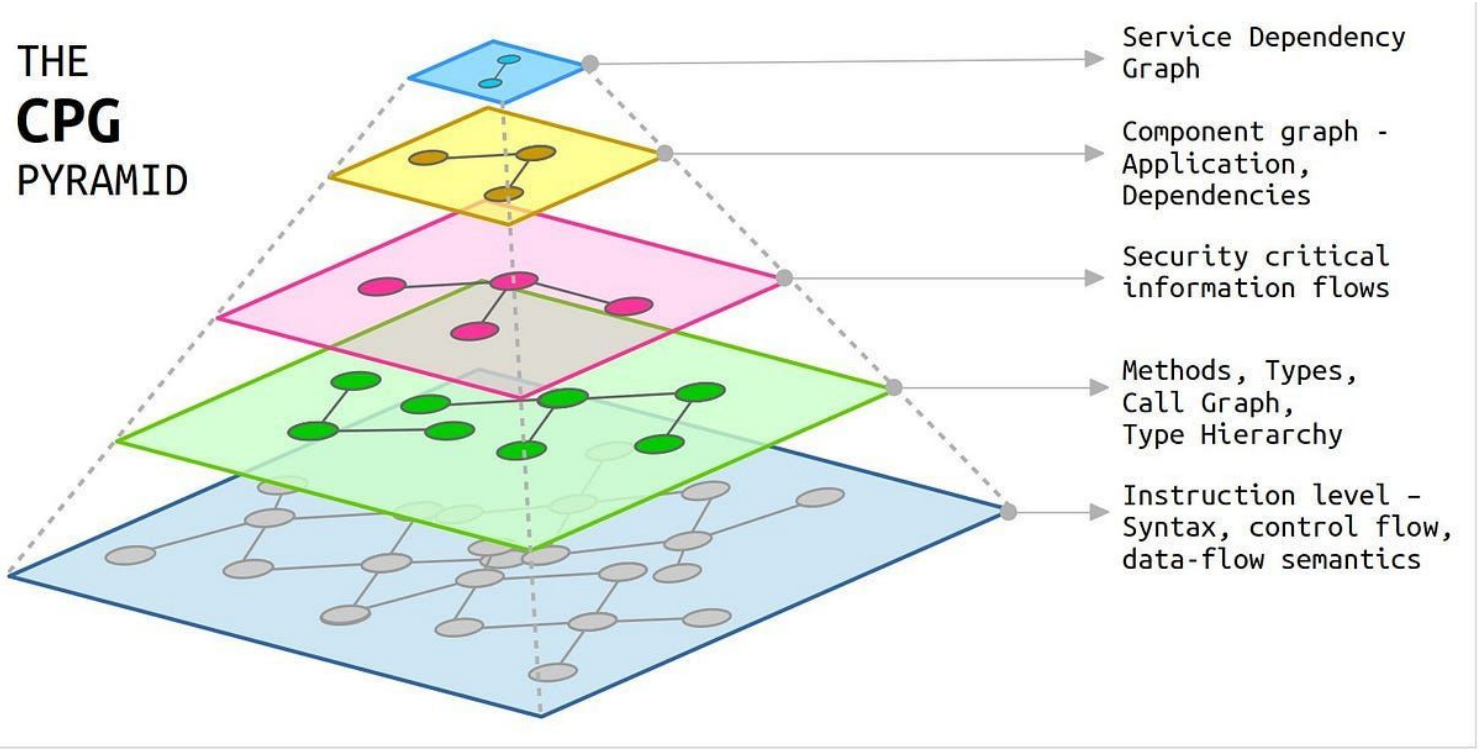
Data flow Graph, Represents how data values flow through the code.

```
w = a + b;
x = a − c;
y = x + d;
x = a + c;
z = y + e;
```



Source Code Representation - CPG

Code Property Graph, Combines AST, CST, CFG, and DFG for a comprehensive view of code properties.



—

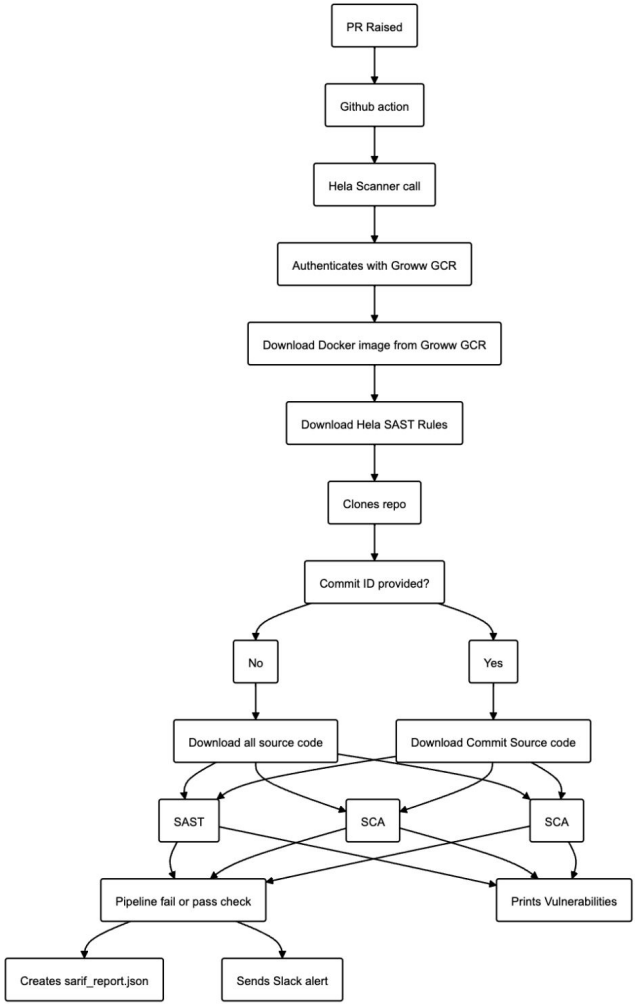
But,

What's Hela?

Hela: God of Death, but here a source code security tool :)

Hela is a fully open-source tool built in Rust that integrates Semgrep, OSV-Scanner, and TruffleHog to perform SAST, SCA, and secret scanning simultaneously. It adds features like scanning PRs instead of the entire codebase, YAML-based declarative rules to fail pipeline builds, a Dashboard UI using defectdojo, and a server mode that improves scan times by 77%.

https://github.com/rohitcoder/hela



How Hela Works?

How to run a Scan?

1. Pull Docker image => docker pull rohitcoder/hela
2. Run Scan => docker run rohitcoder/hela:latest --code-path https://<PAT>@github.com/<ORG>/<REPO> --sast --sca --secret

```yaml
sast:
  critical_count:
    operator: greater_than
  high_count:
    operator: greater_than
    value: 2

sca:
  critical_count:
    operator: greater_than
    value: 2
  high_count:
    operator: greater_than
    value: 1

secret:
  contains:
  - JDBC
  - GITHUB
  - SLACKTOKEN

license:
  contains:
  - AGPL
  - GPL
  - LGPL
```

How to Declare YAML Hela Policy?

```
docker run --platform linux/amd64 rohitcoder/hela --help
Usage:
  hela [OPTIONS]

Scan CLI tool

Optional arguments:
  -h,--help             Show this help message and exit
  -v,--verbose          Enable verbose mode!
  -p,--code-path CODE_PATH
                        Pass the path of the project to scan (Local Path or
                        HTTP Git URL)
  -t,--rule-path RULE_PATH
                        Pass the path of the rules to use (Local Path or HTTP
                        Git URL)
  -i,--commit-id COMMIT_ID
                        Pass the commit ID to scan (Optional)
  -b,--branch BRANCH    Pass the branch name to scan (Optional)
  -s,--sast             Run SAST scan
  -u,--defectdojo-url DEFECTDOJO_URL
                        Pass the defectdojo url to post scan results
  -t,--defectdojo-token DEFECTDOJO_TOKEN
                        Pass the defectdojo API token to post scan results
  -x,--product-name PRODUCT_NAME
                        Pass the defectdojo product name to post scan results
  -g,--engagement-name ENGAGEMENT_NAME
                        Pass the defectdojo engagement name to post scan
                        results
  -c,--sca              Run SCA scan
  -e,--secret           Run Secret scan
  -l,--license-compliance
                        Run License Compliance scan
  -j,--json             Print JSON output, Note: This won't work with
pipeline                check implementation
  -y,--policy-url POLICY_URL
                        Pass the policy url to check if pipeline should fail
  -n,--no-install       Skip installing dependencies
  -r,--root-only        Scan manifests only in the root directory, don't look
                        for manifests in subdirectories
  -d,--build-args BUILD_ARG
                        Pass the build context args to scan
  -m,--manifests MANIFESTS
                        Pass the manifests pom.xml, requirements.txt etc to
                        scan and we will look for only that kind of manifests
  -k,--slack-url SLACK_URL
                        Pass the slack url to receive scan alerts
  -w,--job-id JOB_ID    Pass the job id to store scan results in mongo db
  -o,--mongo-uri MONGO_URI
                        Pass the mongo uri to store scan results
```

What features does Hela support?

1. Full Repo, Branch & PR Scan (Including all commits)
2. Built on Top of Git - Supports Github, BitBucket, CodeCommit, etc
3. SAST, SCA, SECRET, Licence Compliance
4. Pushes Results to DefectDojo and can be integrated with any Vulnerability Management tool, hela outputs report in sarif format.
5. Supports Slack Notifications and maintains DB for reducing Noise for same alerts

For more, Check Hela Repo

https://github.com/rohitcoder/hela

Q&A Time!

Thanks!