Integrating Static Application Security Testing (SAST) Tools in DevSecOps

Jonathan Velasco,
FAE, PhD.

Outline

- Introduction
- DevOps and Security
- Workflow Integration
- Integrating SAST in DevSecOps
- The Added Benefit of Binary Analysis
- Summary

Introduction

DevOps

- DevOps is the combination of software development and systems operations (QA)
- **Delivery** of complex Software Systems (e.g., cyber physical systems) **in shorter time** with **less resources**
- **Cost of failure** increasing as software is often **integrated** into more complex business chains (e.g., IoT edge-to-cloud, smart-grid deployment or automotive use cases)
- Teams trying to **improve** their **tools**, **methodologies** and **processes** enabled the birth of **DevOps**

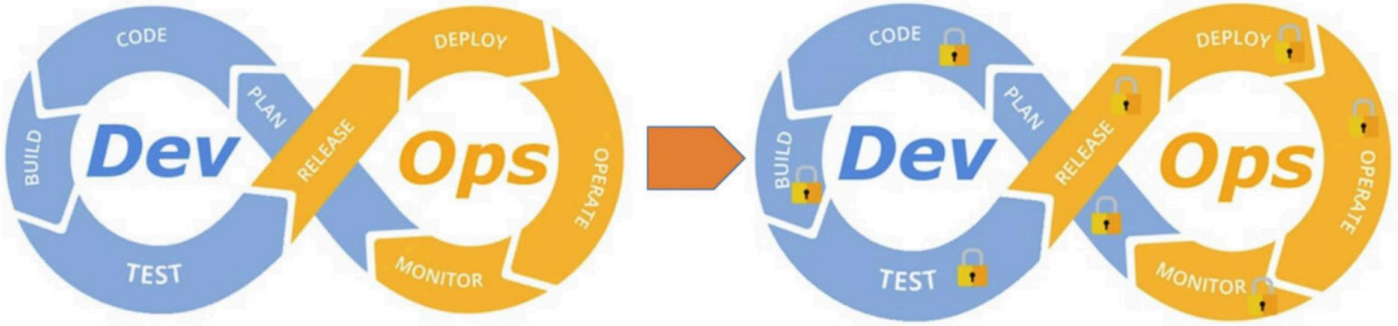
DevSecOps

- DevSecOps is the natural progression in software **development improvement** methods
- **Security** is **included** as a critical part of the **development** process
- **Security failure** is the **same or worse** than a **quality** failure
- DevSecOps is the **integration** at **team level** of the teams **building** software, **operating** the software and **securing** the software

DevOps and Security

DevOps and Security

- **Security** is usually **included** only if it's **part of** the **requirements** and **development** culture
- What **tends to happen** is teams either **ignoring** security or **delegating** it to the end of the development effort
- The **key reason to build security** into Agile and continuous processes is to **build upon knowledge** that accumulates over the project



What are DevOps' and Security's main considerations?

- Security in Code
- Security as Code

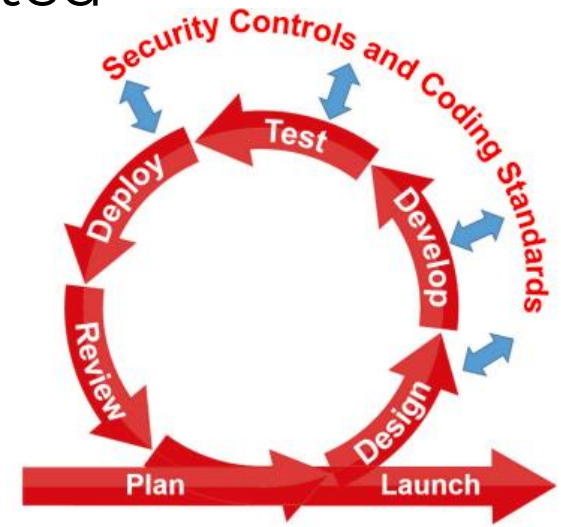
Security **IN** Code

- It is **common** that security-oriented teams **start** with coding **guidelines** or **standards** such as MISRA or the CERT guidelines
- Coding **standards** however **don’t avoid memory problems**
- A **recent study by Microsoft** shows that 70% of today’s security vulnerabilities are **caused by memory vulnerabilities**

Security **AS** Code

- DevSecOps is the concept of **treating security as code**
  - **Guided** by requirements
  - **Leading** to Implementation of security controls
  - **Validation** through testing
  - Documentation

- Security **implementations** ending up **as actual code** with associated tests can be **automated**.



Workflow Integration

Workflow Integration

- **Automation** is one of the main tenants of **DevOps**
- **SAST** can be used **as soon as code has been typed** by a developer
- **When using legacy or third-party code** SAST can be used on those products **before even starting** the current project
- **SAST** comes **in all flavors**, some focusing on **coding standards** and **some more advance** tools go beyond coding standards and explore:
  - Data Flow
  - Control Flow
  - Abstract execution techniques
- **CI/CD processes rely on automation** in order to realize the benefits thereof. **SAST improve code security and quality early in the process** and help the developer to do this as early as possible

Integrating SAST in DevSecOps

Integrating SAST in DevSecOps

- **DevSecOps teams set up** some form of **workflow** where a developer can code in their favorite **environment**.

- The developer then
  - Authors code
  - Builds it
  - Performs unit testing it and debugging

- Once the work item is completed
  - It is submitted
  - Merged back into the mainline codebase

- **Tool integration is a must**. From **requirements management** and **defect tracking** (e.g., JIRA) to **build** automation (e.g., Jenkins), automated **testing**, etc.

Integrating SAST in DevSecOps cont.

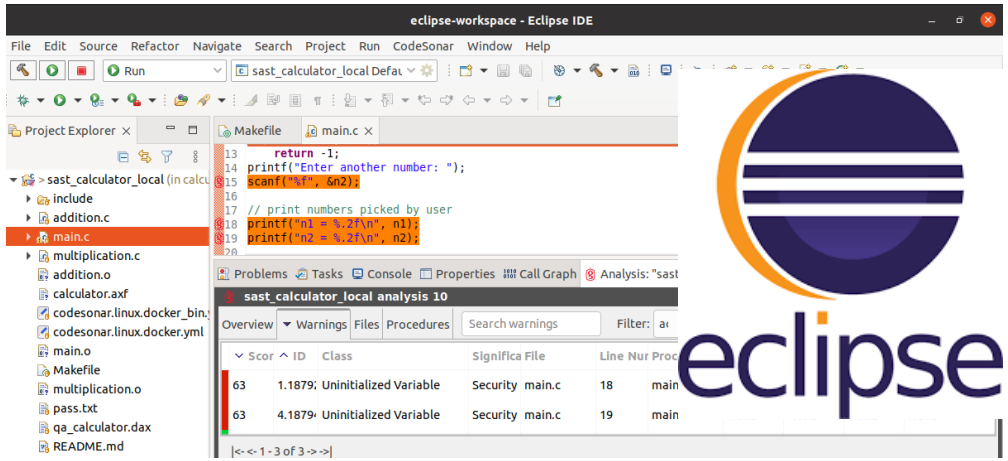
- Development
  - The **integration into the IDE** is important here as it lessens the cognitive load on the software developer
  - SAST is also essential in this phase for ensuring secure **coding standards are met**
- Test
  - **All changes** of the developers are **brought together** for more comprehensive testing
  - **Turnaround time** in this phase is **less critical**, hence SAST provides **more thorough analysis** (e.g., concurrency, tainted dataflows, pointer analysis)
- Deploy
  - SAST can be used to analyze **deployable binaries** and **libraries**
  - This is a good practice **to detect bugs introduced** during the **building** deployment of deliverables
- Review
  - **Evaluate individual** and **higher-level assessments** of application quality and security

The Added Benefit of Binary Analysis

Benefits of Binary Analysis

- **Binary analysis** can be used by security teams to perform **“black box” analysis** of product deliverables
- Not all SAST tools work the same nor all teams work the same way. The **ultimate goal** as a team is to have a **low false negative rate** -- even more important than a low false positive rate.
- The **cost of a false negative** is that a problem **could make it into** the **code base**, maybe it is found during testing, but often it makes it to the **deployed systems**

Demo

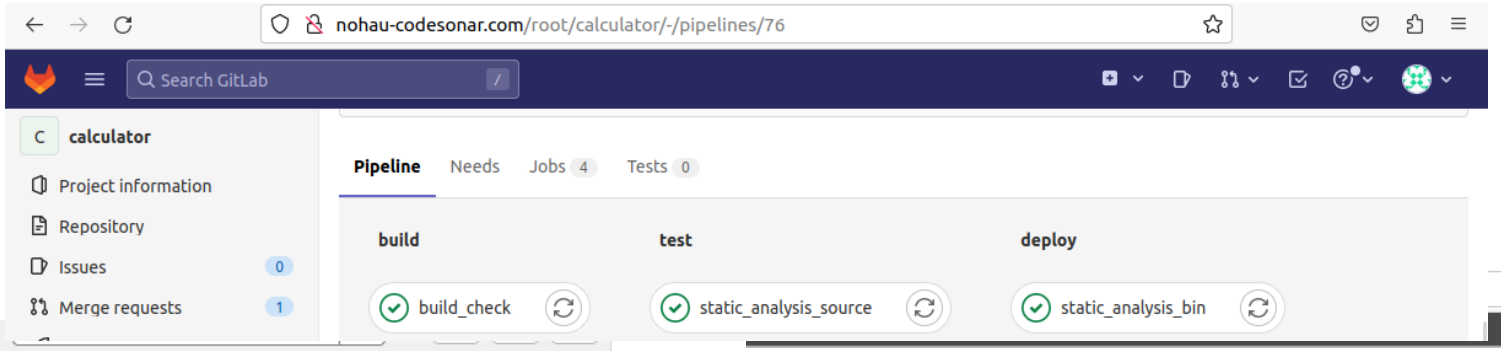


eclipse

nohau

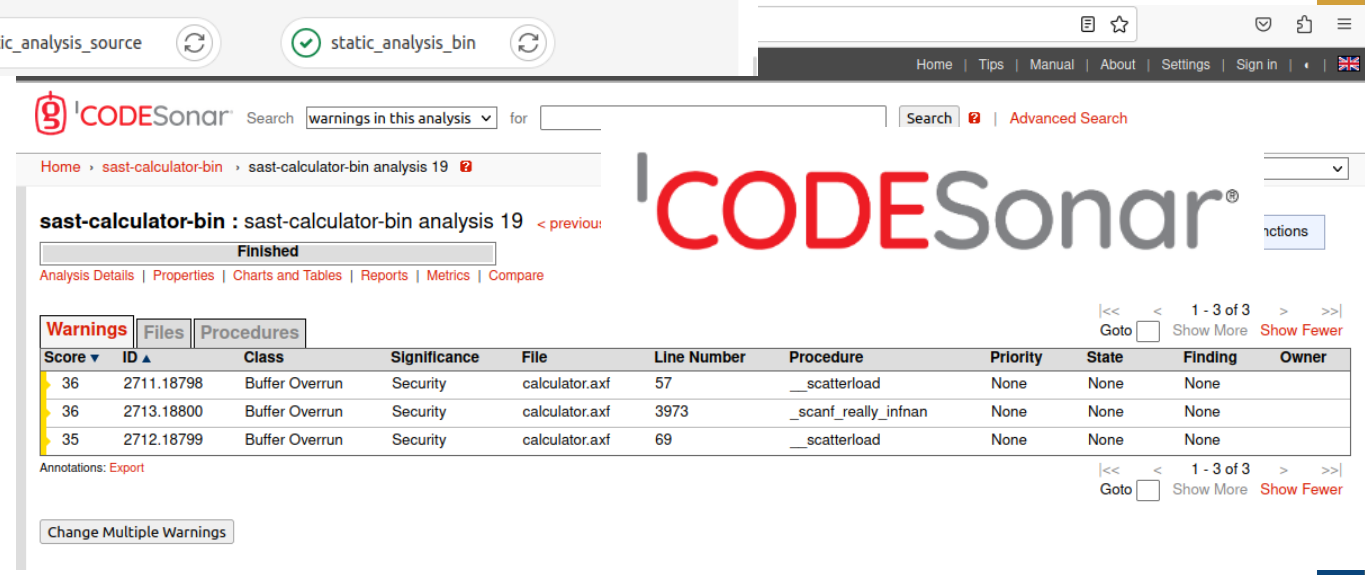
GitLab

docker




```
27 Using docker image sha256:f29eed36a19a74f4cd9db3c402d2034e63a7a1e2d1
   92c0375918870720411 99f for registry.nohau-codesonar.com/root/calculat
   or/codesonar:7.4 with digest registry.nohau-codesonar.com/root/calcul
   ator/codesonar@sha256:5b511153702d8a7bcffebfadfcc8b52d3d824978022400d
   fee0d9658b41aad18 ...
28 $ make
29 armclang --target=arm-arm-none-eabi -mcpu=co
   td=c99 -mfpu=fpv4-sp-d16 -mfloat-abi=hard -fr
   fshort-enums -fshort-wchar -gdwarf-4 -O1 -ffu
   ked -Wno-missing-variable-declarations -Wno-m
   issing-noreturn -Wno-sign-conversion -Wno-nor
   no-reserved-id-macro -Wno-unused-macros -Wno-
   mmand -Wno-documentation -Wno-license-managem
   ality -Wno-reserved-identifier -MD -c -o addi
```

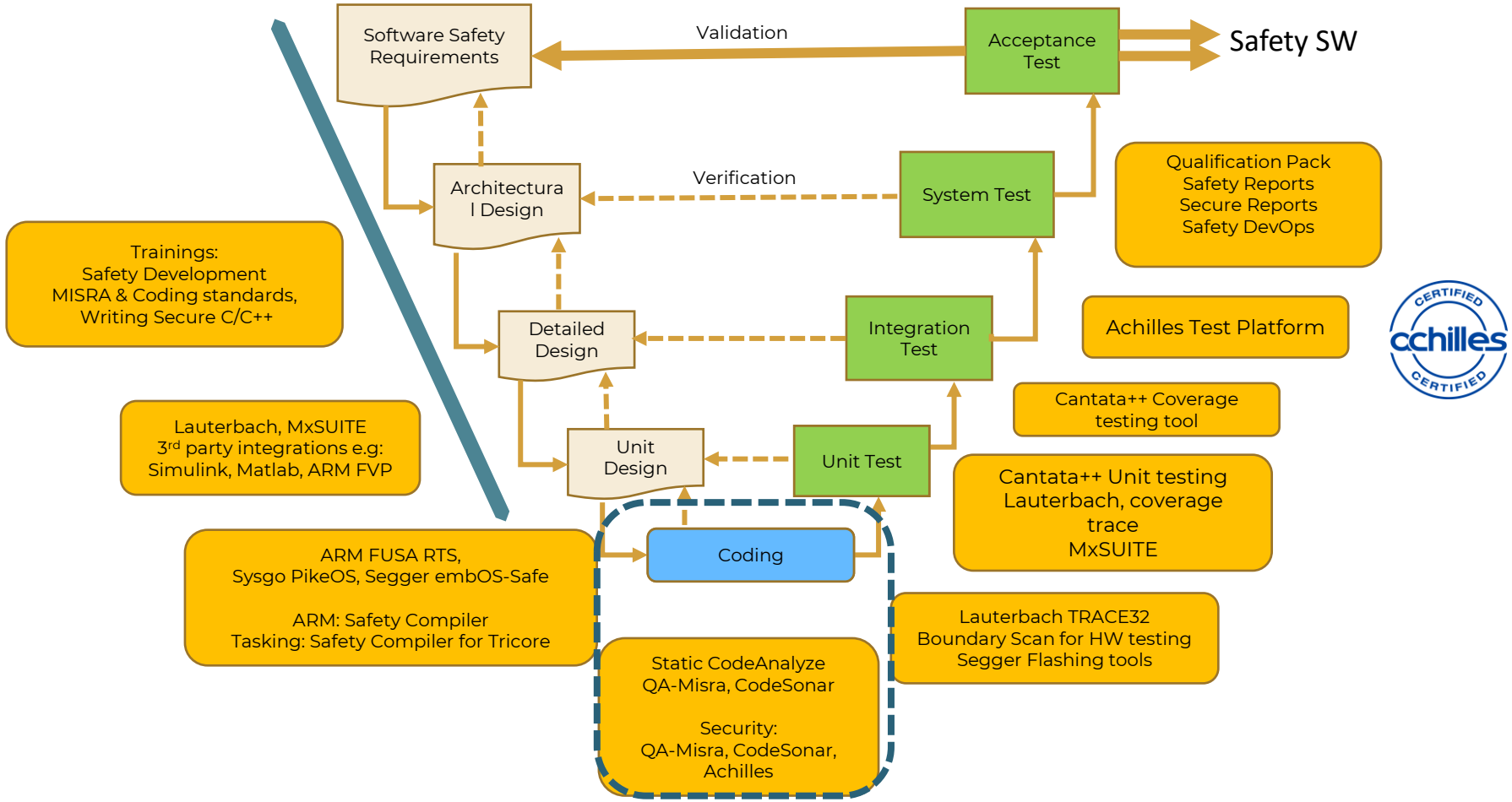
arm



CODESonar®

Nohau Solutions: FuSa products

Nohau's products are TÜV certified IEC 61508, ISO 26262, EN 50128, IEC 62304, DO-178C, etc.



Summary

Summary

- There's an **important role for SAST in the DevSecOps** and are important part of adapting **security** practices **in a continuous integration and deployment pipeline**

- SAST is an important part of the software development automation tool chain that provides **added** security **vulnerability detection**, coding **standard enforcement** and **complementary** security **assurance** to testing and dynamic analysis.

- Reference: https://www.grammatech.com/our-white-papers/integrating-static-application-security/

Nohau Solutions AB ©