The Security Testing Pyramid

Entwicklertag Karlsruhe - 10.06.2021

https://github.com/andifalk/bookmark-service

NOVATEC

About Me

Andreas Falk

Novatec Consulting (Germany)

andreas.falk@novatec-gmbh.de

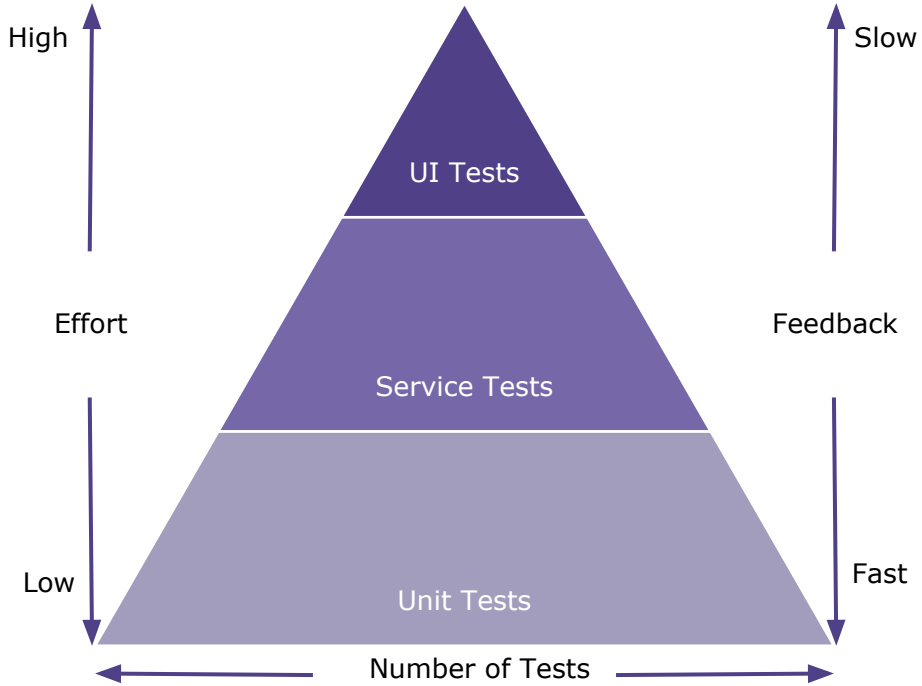
@andifalk

OWASP Open Web Application Security Project

OpenID

NOVATEC

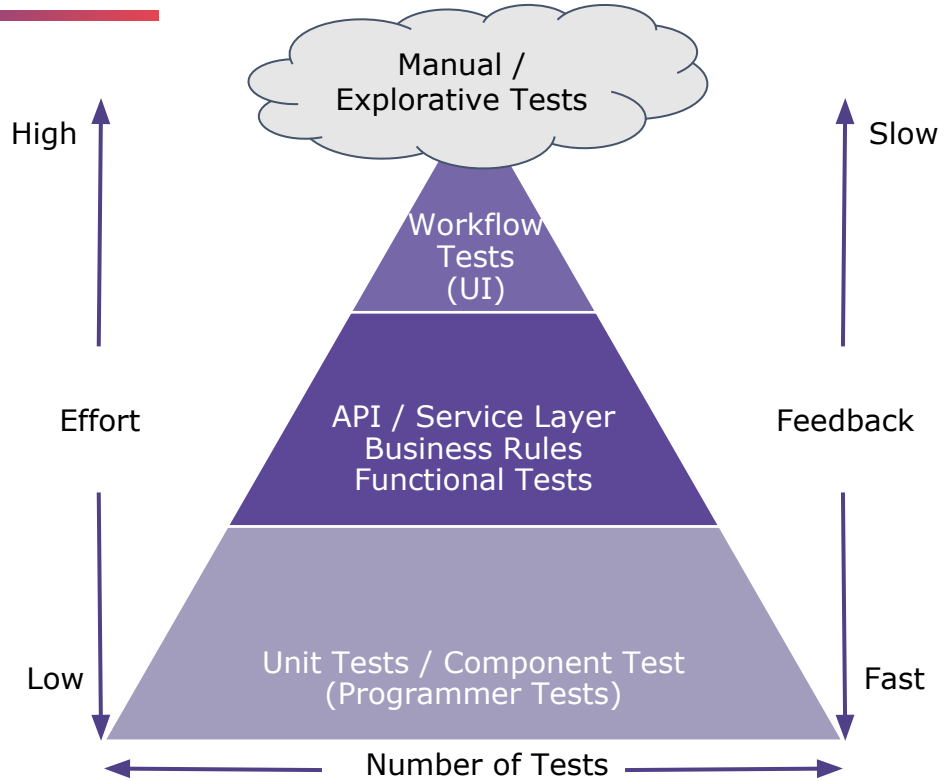
The Testing Pyramid (by Mike Cohn)



https://www.mountaingoatsoftware.com/blog/the-forgotten-layer-of-the-test-automation-pyramid
https://martinfowler.com/articles/practical-test-pyramid.html

NOVATEC

The Testing Pyramid (Gregory / Crispin)

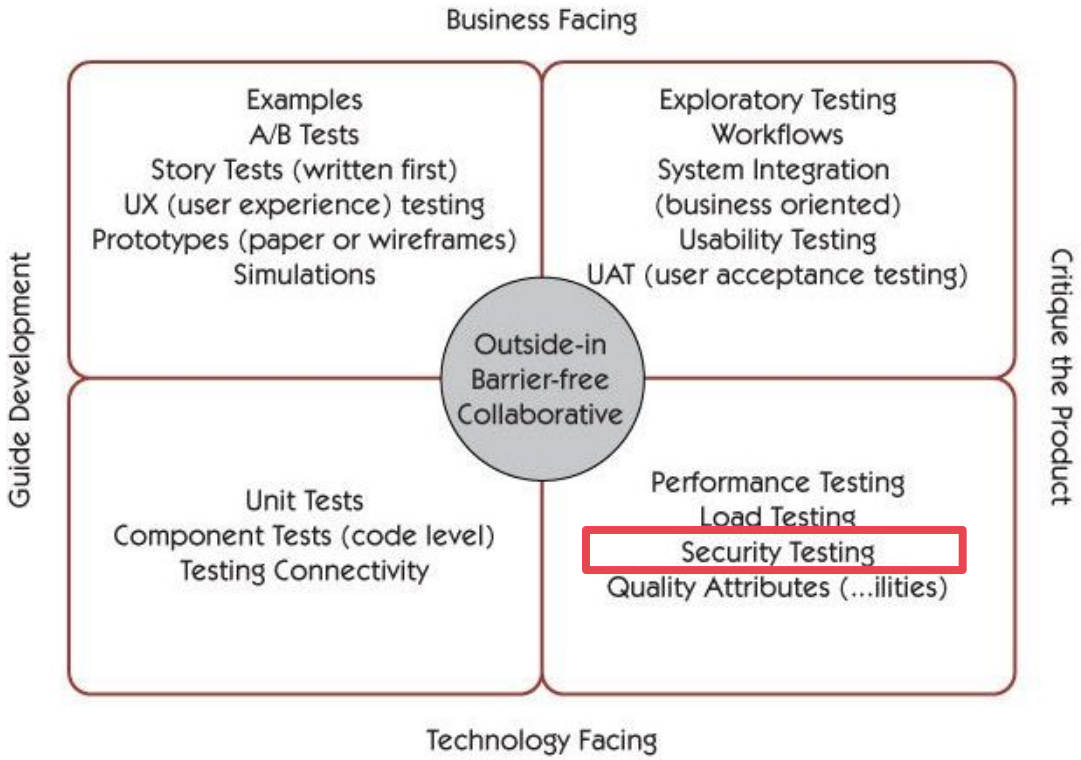


https://agiletester.ca/more-agile-testing-the-book

NOVATEC

What About Security?

Agile Testing Quadrants (Gregory / Crispin)



https://agiletester.ca/more-agile-testing-the-book

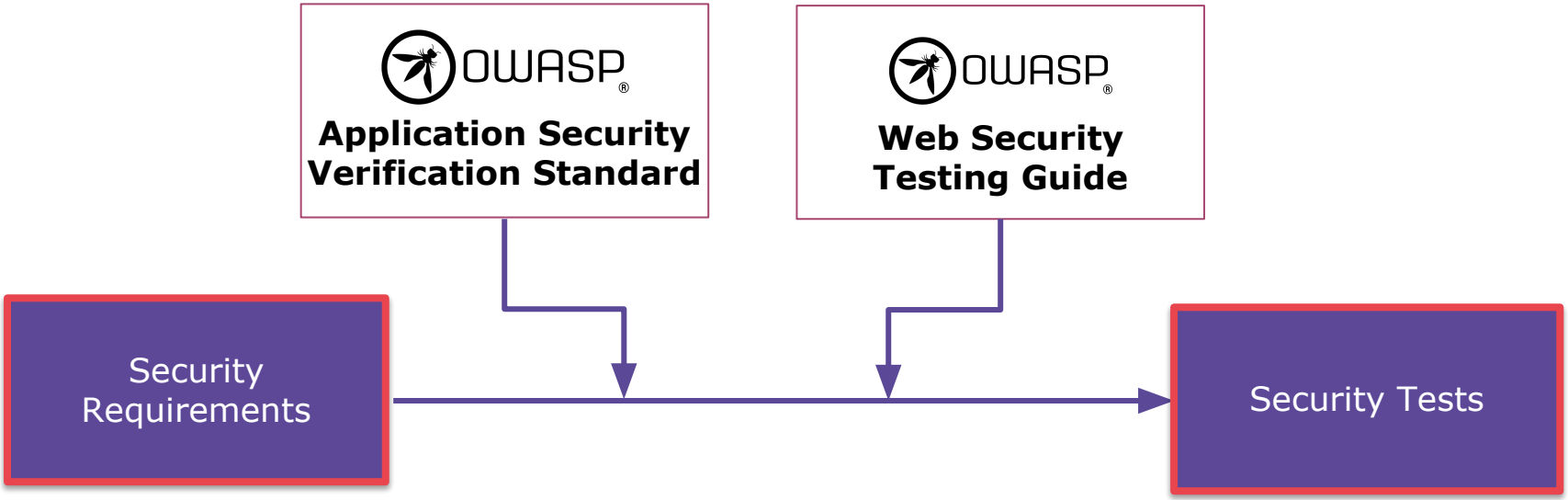
NOVATEC

OWASP Top 10 Web Application Security Risks

- Top 1: **Injection**
- Top 2: **Broken Authentication**
- Top 3: **Sensitive Data Exposure**
- Top 4: **XML External Entities (XXE)**
- Top 5: **Broken Access Control**
- Top 6: **Security Misconfiguration**
- Top 7: **Cross-Site Scripting (XSS)**
- Top 8: **Insecure Deserialization**
- Top 9: **Using Components with Known Vulnerabilities**
- Top 10: **Insufficient Logging & Monitoring**

https://owasp.org/www-project-top-ten/2017

Derive Tests from Security Requirements



https://owasp.org/www-project-application-security-verification-standard
https://owasp.org/www-project-web-security-testing-guide

NOVATEC

Application Security Verification Standard (1)

- V1: **Architecture, Design and Threat Modeling**
- V2: **Authentication**
- V3: **Session Management**
- V4: **Access Control**
- V5: **Validation, Sanitization and Encoding**
- V6: **Stored Cryptography**
- V7: **Error Handling and Logging**

https://github.com/OWASP/ASVS

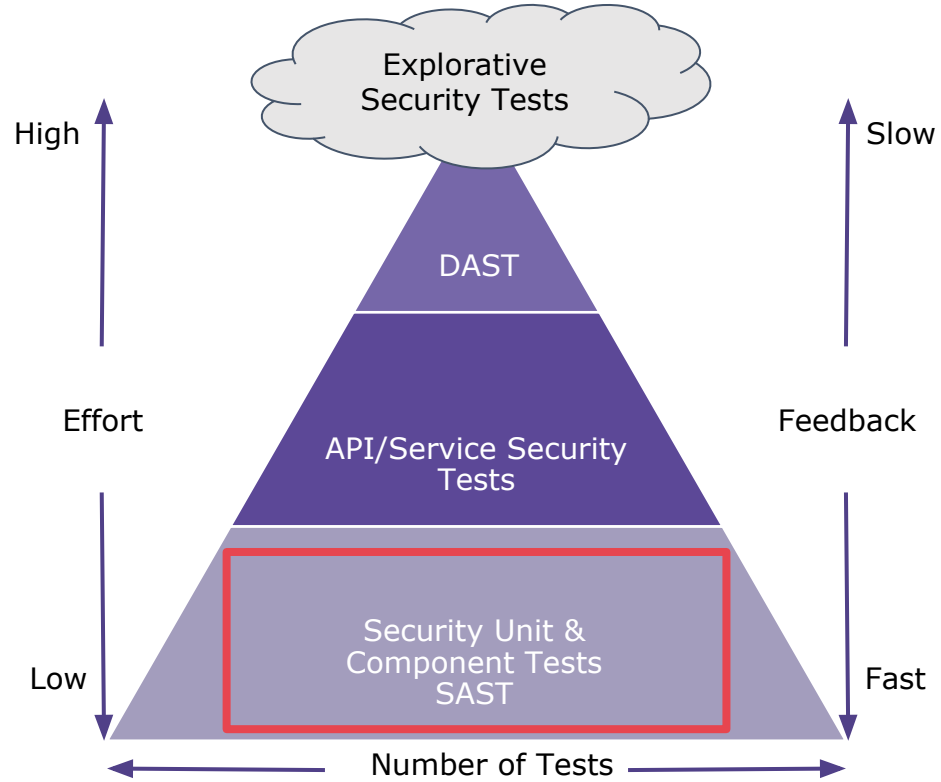
Application Security Verification Standard (2)

- V8: **Data Protection**
- V9: **Communications**
- V10: **Malicious Code**
- V11: **Business Logic**
- V12: **File and Resources**
- V13: **API and Web Service**
- V14: **Configuration**

https://github.com/OWASP/ASVS

Unit-Test Layer

The Security-Testing Pyramid: Unit Test Layer



- Static Application Security Testing (SAST)
  - Static Code Analysis
- Using Components With Known Vulnerabilities
  - Dependency Check
  - Container Image Scan
- Unit / Component Tests
  - Injection (Input Validation)
  - Broken Authentication
  - Bypass Business Logic
  - Error Handling & Logging
  - Secure Architecture

NOVATEC

Static Application Security Testing

"Static application security testing (SAST) is a set of technologies designed to analyze application source code, byte code and binaries for coding and design conditions that are indicative of security vulnerabilities. SAST solutions analyze an application from the "inside out" **in a nonrunning state**."

— Gartner IT Glossary

- In other words, the artifacts themselves are inspected (e. g. source code instead of the program it represents)
- Presented SAST tools:
  - Find Security Bugs (SpotBugs)
  - SonarQube
  - OWASP Dependency Check

OWASP Top 10 (2017)

- A2:2017-Broken Authentication
  - Attack Vectors
    Attackers have access to hundreds of millions of valid username and password combinations for credential stuffing, default administrative account lists, automated brute force, and dictionary attack tools.
  - How to Prevent
    - Multi-factor authentication
    - No default credentials
    - Weak-password checks, such as testing new or changed passwords against a list of the top 10000 worst passwords.
    - Align password policies with NIST 800-63 B's guidelines
    - …

V2: Authentication (ASVS)

- V2.1 Password Security Requirements
  - 2.1.1 Verify that user set passwords are at least 12 characters in length
  - 2.1.2 Verify that passwords 64 characters or longer are permitted but may be no longer than 128 characters
  - 2.1.3 Verify that password truncation is not performed
  - 2.1.4 Verify that any printable Unicode character is permitted in passwords
  - 2.1.7 Verify that passwords are checked against a set of breached passwords

4.4 Authentication Testing (Testing Guide)

- Testing for Weak Password Policy

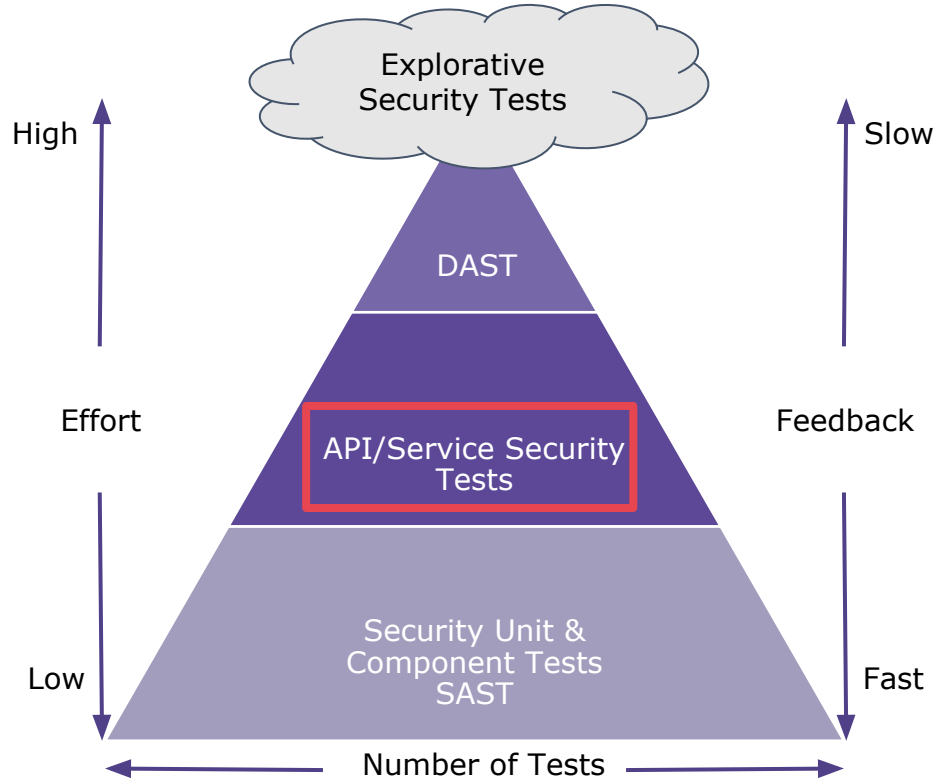
1. What characters are permitted and forbidden for use within a password? Is the user required to use characters from different character sets?
2. Is the user prevented from using his username or other account information (such as first or last name) in the password?
3. What are the minimum and maximum password lengths that can be set?
4. Is it possible set common passwords such as Password1 or 123456?

Demo

Unit Test Layer

Service-Test Layer

The Security-Testing Pyramid: API/Service Test Layer



- API / Service Tests
  - Input Validation
  - Authentication
  - Authorization
  - Session Management
  - Output Escaping (XSS)
  - Injection
  - Security Misconfiguration

NOVATEC

V4: Access Control (ASVS)

- V4.1 General Access Control Design
  - 4.1.1 Verify that the application enforces access control rules on a trusted service layer, especially if client-side access control is present and could be bypassed
  - 4.1.3 Verify that the principle of least privilege exists - users should only be able to access functions, data files, URLs, controllers, services, and other resources, for which they possess specific authorization
  - 4.1.5 Verify that access controls fail securely including when an exception occurs.

4.5 Authorization Testing (Testing Guide)

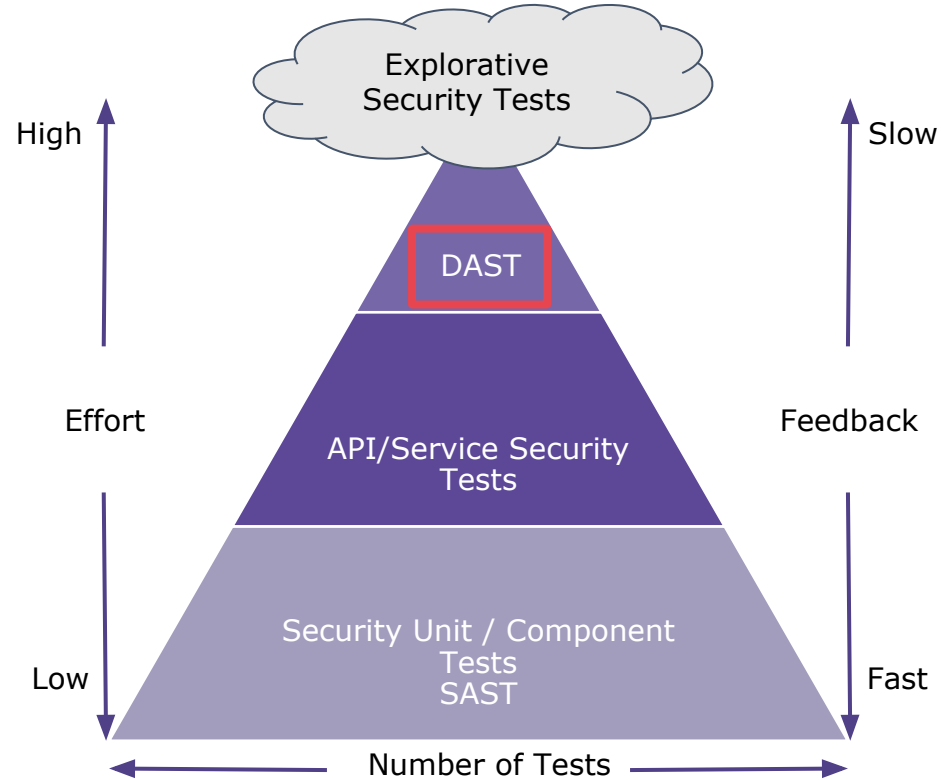
- Testing for Bypassing Authorization Schema
  1. Testing for Access to Administrative Functions
  2. Testing for Access to Resources Assigned to a Different Role
- Testing for Role/Privilege Manipulation
  - URL Traversal: Try to traverse the website and check if some of pages that may miss the authorization check.
  - WhiteBox: If the URL authorization check is only done by partial URL match, then it’s likely testers or hackers may workaround the authorization by URL encoding techniques.

Demo

Service Test Layer

NOVATEC

The Security-Testing Pyramid: DAST



- Dynamic Application Security Testing (DAST)
  - OWASP Zap / StackHawk
  - Portswigger Burp Suite
  - SQLMap
  - NMap
  - …

NOVATEC

Dynamic Application Security Testing


```
$ docker pull owasp/zap2docker-stable
```

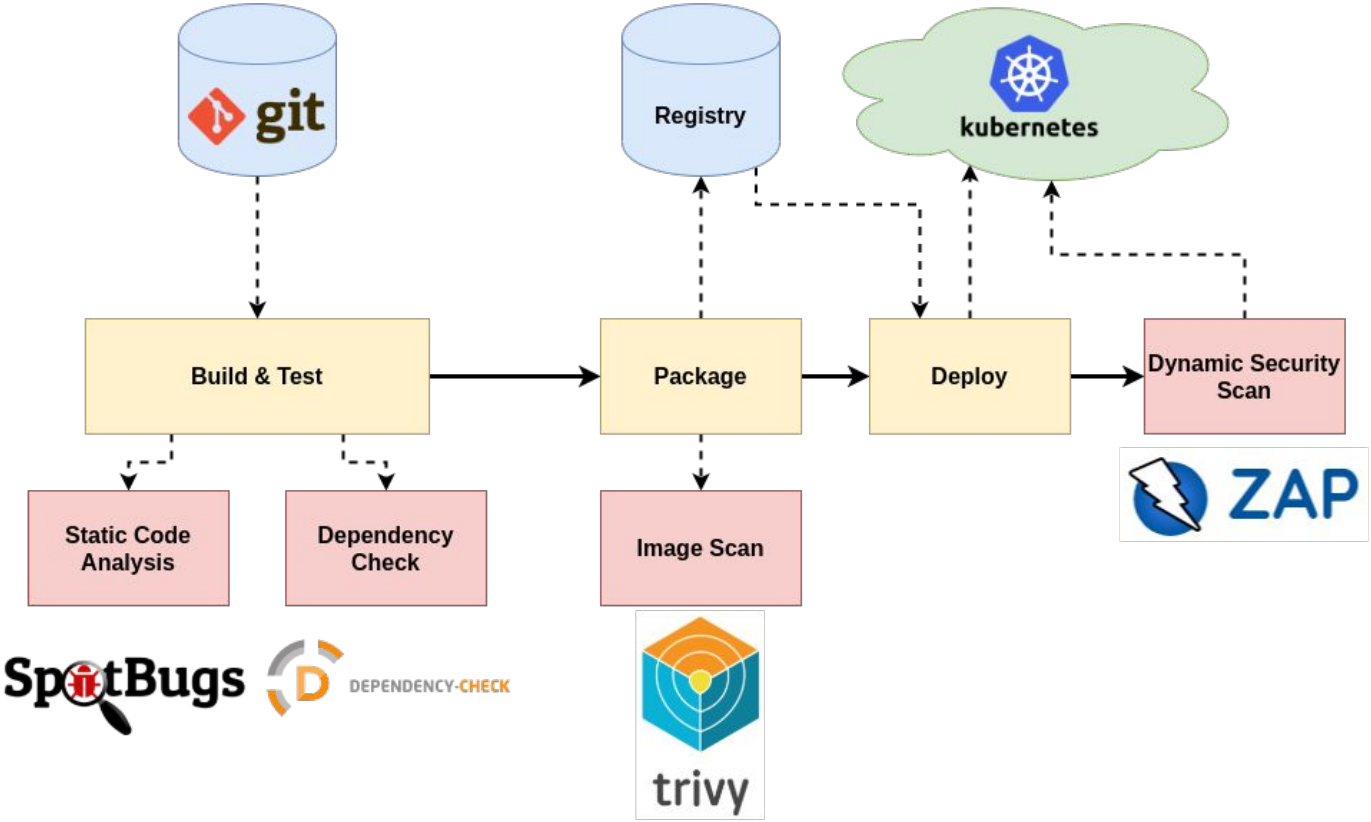
- Baseline Scan
  - Runs the spider and passive scanning:
    zap-baseline.py -t https://www.example.com
- API Scan
  - Performs active scan against APIs defined by OpenAPI
    zap-api-scan.py -t https://example.com/openapi.json -f openapi
- Full Scan
  - Runs the ZAP spider and a full active scan (+ opt. ajax scan)
    zap-full-scan.py -t https://www.example.com

https://www.zaproxy.org/docs/docker

Demo

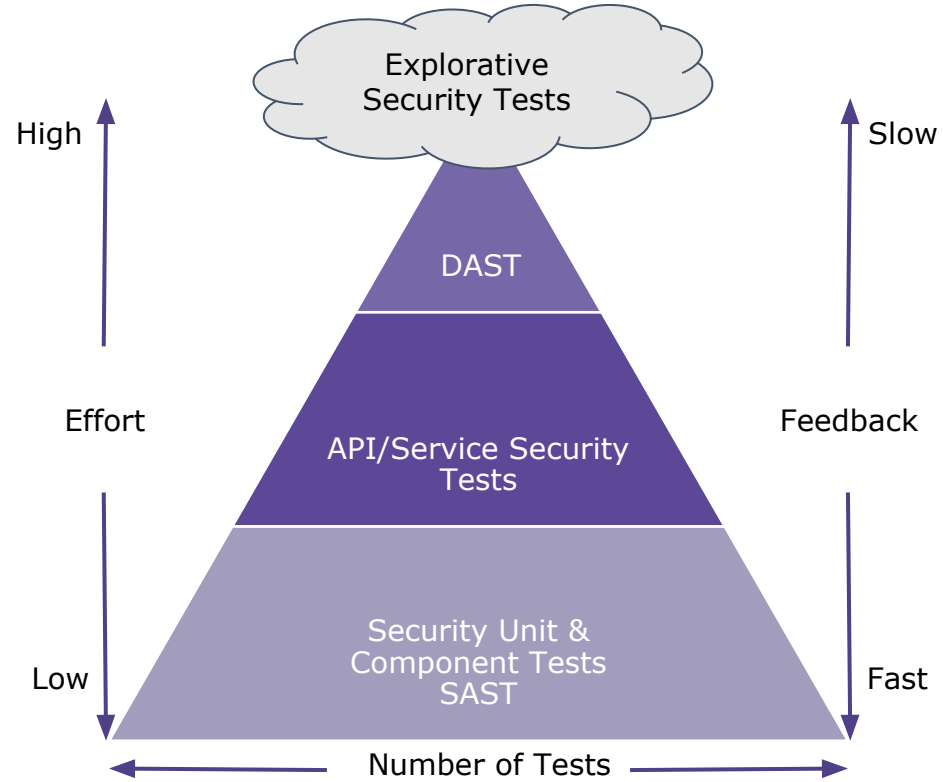
Dynamic Security Testing

Automate all the Security Things!



NOVATEC

Summary: The Security-Testing Pyramid



- Security Charters / Code Reviews / Pen-Tests
- Dynamic Application Security Testing (DAST)
- Input Validation, Authentication, Authorization, Session Management, Output Escaping (XSS), SQL Injection, Security Misconfiguration
- Input Validation, Broken Authentication, Secure Architecture, Bypass Business Logic, Error Handling & Logging
- Static Application Security Testing (SAST)

NOVATEC

Thanky You. Any Questions?

Feedback: https://feedback.andrena.de/f/7gf75q7z

NOVATEC

Thanks for your attention

Novatec Consulting GmbH
Dieselstraße 18/1
D-70771 Leinfelden-Echterdingen

T. +49 711 22040-700
info@novatec-gmbh.de
www.novatec-gmbh.de