Unleashing the Power of Interactive Application Security Testing (IAST)

Amol Gangurde

OWASP Pune Chapter Meet

January 20, 2024

Introduction

Amol Gangurde

- *AppSec Team at Global Payments*
- *13+ years of Experience in IT*
- *Web and API Penetration Testing*
- *DevSecOps*
- *Tools Integration and Security Automation*

What is Interactive Application Security Testing (IAST)?

- It is a security testing method that analyzes the application while it is running
- It uses sensors or agents that are embedded in the application code
- It monitors the application behavior and data flow
- It reports security vulnerabilities in real-time

IAST

Traditional vs. IAST

Static Application Security Testing (SAST)

What it does?

It analyzes the source code of an application without actually running it.

How it works?

Scans the code for patterns and known vulnerabilities.

Strengths

Can catch a wide range of vulnerabilities early in the development process,often at lower cost

Weaknesses

May produce many false positives, can't detect runtime issues, requires access to source code.

Traditional vs. IAST

Dynamic Application Security Testing (DAST)

What it does?

Simulates attacks on a running application from an external perspective.

How it works?

Sends test inputs to the application and observes its responses.

Strengths

Can detect vulnerabilities that SAST cannot, like logic flaws and real-world attack scenarios.

Weaknesses

May miss vulnerabilities not exposed by specific tests, can be resource-intensive and slow down testing.

Traditional vs. IAST

IAST (Interactive Application Security Testing)

What it does

Combines features of both SAST and DAST by analyzing code and monitoring runtime behavior.

How it works

Instruments the application code to track data flow and execution, then analyzes potential security risks.

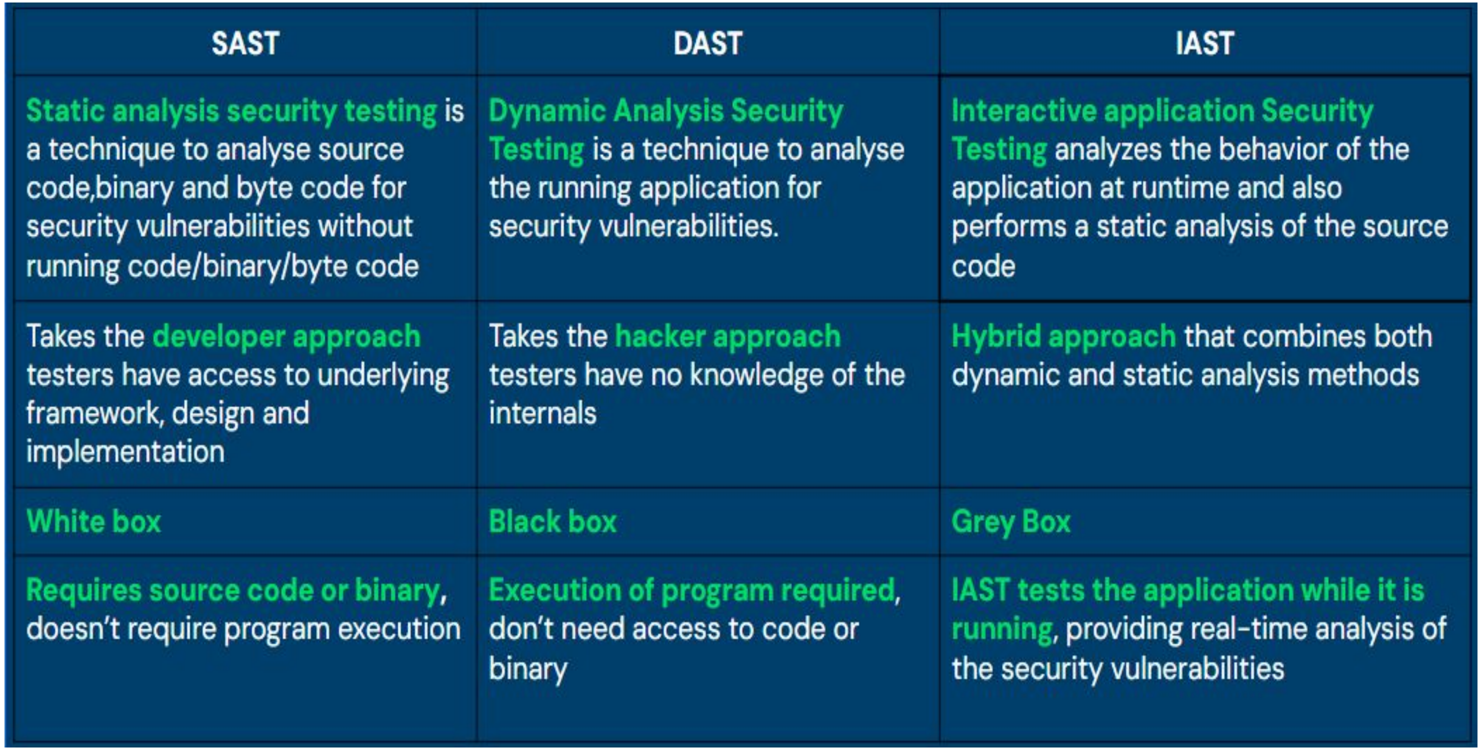
Strengths

Provides **more accurate** vulnerability detection than SAST and DAST, can detect runtime issues and false positives.

Weaknesses

May **impact application performance** to some degree, as it requires agent deployment on servers.

Traditional vs. IAST



SAST	DAST	IAST
Static analysis security testing is a technique to analyse source code,binary and byte code for security vulnerabilities without running code/binary/byte code	Dynamic Analysis Security Testing is a technique to analyse the running application for security vulnerabilities.	Interactive application Security Testing analyzes the behavior of the application at runtime and also performs a static analysis of the source code
Takes the developer approach testers have access to underlying framework, design and implementation	Takes the hacker approach testers have no knowledge of the internals	Hybrid approach that combines both dynamic and static analysis methods
White box	Black box	Grey Box
Requires source code or binary, doesn't require program execution	Execution of program required, don't need access to code or binary	IAST tests the application while it is running, providing real-time analysis of the security vulnerabilities

Overview of IAST

Steps in IAST Process

Sensors/Agents

Embedded in code, track behavior and data flow

Data Transmission

Send data to a central server or dashboard

Analysis

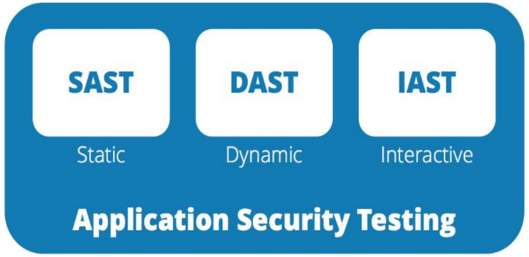
Server/dashboard analyzes data to identity vulnerabilities

Alerts

Alerts developers or security team in real time about vulnerabilities

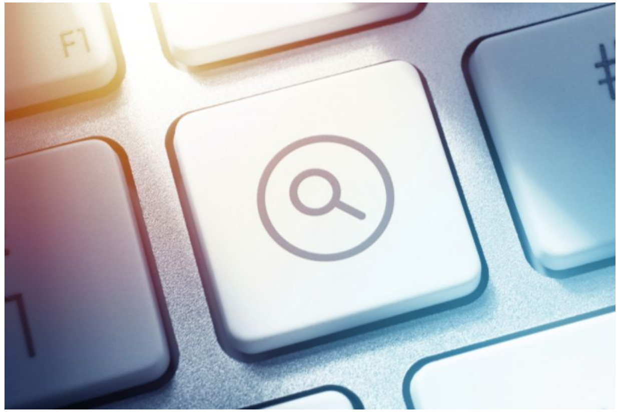
Advantage of IAST

- IAST can detect vulnerabilities that are missed by static analysis (SAST) and/or dynamic analysis (DAST)
- IAST can test APIs and microservices, which are often hard to scan with other methods
- IAST can integrate with automated functional tests or manual tests
- IAST can provide continuous feedback to developers and security teams
- IAST can reduce false positives and false negatives



Key features of IAST

- Promotes shift left approach by integrating easily into CI/CD
- Provides accurate results for fast triage
- Pinpoints the source of vulnerabilities
- Allows for earlier, less costly fixes



Challenges and Considerations

- Complex application environments
- Programming-language dependent
- Time intensive
- Doesn’t have 100% code coverage
- Might impact performance

IAST tools available in market

Acunetix

Checkmarx

CONTRAST SECURITY

Fortify

HCLSoftware

SYNOPSYS®

Invicti

VERACODE

Best practices for IAST

- Deploy IAST in a QA environment with automated functional tests running
- Educate Development Teams
- Automate Testing Workflows
- Establish clear protocols

DEMO
LOADIND...

IAST using Contrast

- Contrast Community Edition (CE) – Fully featured version for 1 app and up to 5 users (some Enterprise features disabled). Contrast CE supports Java and .NET only.

Thank **You**

amol24by7@gmail.com

@hackwithamol