Unleashing the Power of Interactive Application Security Testing (IAST)

Amol Gangurde

OWASP Pune Chapter Meet

January 20, 2024
Amol Gangurde

Introduction

- 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.
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.
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

<table>
<thead>
<tr>
<th></th>
<th>SAST</th>
<th>DAST</th>
<th>IAST</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Static analysis security testing</strong></td>
<td>is a technique to analyse source code, binary and byte code for security vulnerabilities without running code/binary/byte code</td>
<td>Dynamic Analysis Security Testing is a technique to analyze the running application for security vulnerabilities.</td>
<td>Interactive application Security Testing analyzes the behavior of the application at runtime and also performs a static analysis of the source code</td>
</tr>
<tr>
<td><strong>Takes the developer approach</strong></td>
<td>testers have access to underlying framework, design and implementation</td>
<td>Takes the hacker approach</td>
<td>Hybrid approach that combines both dynamic and static analysis methods</td>
</tr>
<tr>
<td><strong>White box</strong></td>
<td></td>
<td>Black box</td>
<td>Grey Box</td>
</tr>
<tr>
<td><strong>Requires source code or binary, doesn’t require program execution</strong></td>
<td>Execution of program required, don’t need access to code or binary</td>
<td></td>
<td>IAST tests the application while it is running, providing real-time analysis of the security vulnerabilities</td>
</tr>
</tbody>
</table>
Overview of IAST

Steps in IAST Process

1. Sensors/Agents
   - Embedded in code, track behavior and data flow

2. Data Transmission
   - Send data to a central server or dashboard

3. Analysis
   - Server/dashboard analyzes data to identity vulnerabilities

4. 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, HCL Software, 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
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