HOW TO BUILD AWESOME SECURITY INSTRUMENTATION TO AUTOMATE APPSEC TESTING AND PROTECTION

Jeff Williams | CTO and cofounder | jeff.williams@contrastsecurity.com
THE GOAL IS SIMPLE:

DELIVER VALUE TO CUSTOMERS, FASTER

DEVELOPMENT’S RESPONSE:

FLOW | FEEDBACK | CONTINUOUS EXPERIMENTATION & LEARNING

MAJOR INITIATIVES:
- SECURITY MODERNIZATION
- DEVOPS MANDATE
- DIGITAL TRANSFORMATION
- LEGACY TOOL REVIEW
- BREACH PREVENTION
- AUTOMATION INITIATIVES

DEV SEC OPS

SECURITY’S CHALLENGE:

HOW DO YOU SECURE MODERN SOFTWARE WITH LEGACY TOOLS?

LEGACY OUTCOMES:
1. SECURITY SLOWS VALUE DELIVERY STREAM
2. DEVELOPMENT BYPASSES SECURITY & RELEASE INSECURE CODE
3. RUN SECURITY ASYNCHRONOUS TO SDLC AS A COMPLIANCE CHECK

MODERN SECURITY CAN BE MORE!
IS IT WORKING YET?

2002 - Average serious vulnerabilities: 26.7

2019 - Average serious vulnerabilities: 26.7

Formal Methods
- SW CMM
- Vulnerability Disclosure
- Common Criteria
- OWASP T10
- WAF
- Developer Training
- DevOps
- Shift Left


- TCSEC DOD 5200.28
- SSE-CMM
- Penetration Testing
- Dynamic Scanning
- Static Analysis
- Compliance
- BSIMM OpenSAMM
- DevSecOps
DEVSECOPS IS ABOUT CHANGING *SECURITY* NOT *DEVOPS*
INSTRUMENTATION CHANGES EVERYTHING
WHAT IS SECURITY INSTRUMENTATION?

Platform

Agent | Transformer | Loader

Add sensors here!

Original Code

Instrumented Code

Run instrumented code!
1. SECURITY TESTING WITH INSTRUMENTATION

```java
package com.contrastsecurity;

import static net.bytebuddy.matcher.ElementMatchers.*;

public class UnsafeSQLAgent {

    public static void premain(String arg, Instrumentation inst) throws Exception {
        System.out.println( "UnsafeSQLAgent installed" );
        new AgentBuilder.Default()
            .ignore( none() )
            .type( hasGenericSuperType( named("java.sql.Statement")) )
            .transform((b, td, cl, m) -> b.visit(Advice.to(SecurityAdvice.class)
                .on( named("execute") ).or( named("executeQuery") )))
            .installOn( inst );
    }

    public static class SecurityAdvice {
        @Advice.OnMethodExit
        public static void exit(@Advice.Argument(0) Object p) throws Exception {
            System.out.println( "WARNING: Unparameterized SQL -> " + p );
            new Throwable().printStackTrace();
        }
    }
}
```

- Put SecurityAdvice inside Statement.execute
- Report rule violation when it happens
Now your normal test cases fail for security reasons if you have a vulnerability!
TYPICAL “SCANNING” APPROACH

SCANNER SIMULATES ATTACKS TO DETECT VULNERABILITIES

ATTACKS

HTTP RESPONSES
• HTTP RESPONSE

PROBLEMS WITH SPEED, SCALE, AND ACCURACY
TESTING WITH AN AGENT ON THE INSIDE!

DEVELOPER USES APPLICATION NORMALLY

NORMAL APP USE

AGENT OBSERVES APPLICATION BEHAVIOR DIRECTLY

SECURITY SPECIALISTS TRANSLATE NEW THREATS INTO RULES

DETAILED SECURITY ANALYSIS RESULTS

- EXACT LINE OF CODE, FULL QUERY, FULL HTTP REQUEST, CONTROL FLOW, DATA FLOW, LIBRARIES, CONFIGURATION, ETC...
DEMO:

“NO CODE” SECURITY OBSERVABILITY!
DEMO:

TESTING ACCESS CONTROL!

<table>
<thead>
<tr>
<th>Test Coverage Matrix</th>
<th>Guest</th>
<th>UserB</th>
<th>UserC</th>
<th>UserD</th>
<th>UserE</th>
</tr>
</thead>
<tbody>
<tr>
<td>/ticketbook/accessA.jsp</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>/ticketbook/accessB.jsp</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/accessC.jsp</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/accessD.jsp</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>/ticketbook/accessE.jsp</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/architecture.jsp</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/cmd.jsp</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/forward.jsp</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/hash.jsp</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/redirect.jsp</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/xss.jsp</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/xxe.jsp</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Access Control Matrix</th>
<th>RoleA</th>
<th>RoleB</th>
<th>RoleC</th>
<th>RoleD</th>
<th>RoleE</th>
</tr>
</thead>
<tbody>
<tr>
<td>/ticketbook/accessA.jsp</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/accessB.jsp</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/accessC.jsp</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/accessD.jsp</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>/ticketbook/accessE.jsp</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
DEMO:

MAKING SECURITY OBSERVABLE WITH INSTRUMENTATION

- name: "get-ciphers"
  description: "Identifies encryption ciphers"
  methods:
  - "javax.crypto.Cipher.getInstance"
  captures:
  - "#P0"
  matchers:
  - "!null"

- name: "native-libraries"
  description: "Identifies the use of native libraries"
  methods:
  - "java.lang.System.load"
  - "java.lang.System.loadLibrary"
  - "java.lang.System.mapLibraryName"
  captures:
  - "#P0"
  matchers:
  - "!null"

- name: "get-unsafe-queries"
  description: "Identifies unparameterized database queries"
  methods:
  - "java.sql.Statement.execute"
  - "java.sql.Statement.addBatch"
  - "java.sql.Statement.executeQuery"
  - "java.sql.Statement.executeUpdate"
  excludes:
  - "java.sql.PreparedStatement"  #these calls are harmless in PreparedStatement
  captures:
  - "#ARGS"
## DEMO: MAKING SECURITY OBSERVABLE WITH INSTRUMENTATION

<table>
<thead>
<tr>
<th>Test Coverage Matrix</th>
<th>Guest</th>
<th>UserA</th>
<th>UserB</th>
<th>UserC</th>
<th>UserD</th>
<th>UserE</th>
</tr>
</thead>
<tbody>
<tr>
<td>/ticketbook/accessA.jsp</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>/ticketbook/accessB.jsp</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>/ticketbook/accessC.jsp</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>/ticketbook/accessD.jsp</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>/ticketbook/accessE.jsp</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>/ticketbook/check.jsp</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>/ticketbook/list.jsp</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>/ticketbook/profile</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Access Control Matrix</th>
<th>RoleA</th>
<th>RoleB</th>
<th>RoleC</th>
<th>RoleD</th>
<th>RoleE</th>
</tr>
</thead>
<tbody>
<tr>
<td>/ticketbook/accessA.jsp</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/accessB.jsp</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/accessC.jsp</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/accessD.jsp</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>/ticketbook/accessE.jsp</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>/ticketbook/check.jsp</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/list.jsp</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/profile</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cipher Matrix</th>
<th>AES</th>
<th>DES</th>
<th>DES/CBC/PKCS5Padding</th>
<th>DESede</th>
<th>PBEWithMD5AndTripleDES</th>
</tr>
</thead>
<tbody>
<tr>
<td>/ticketbook/accessA.jsp</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/accessB.jsp</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/accessC.jsp</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/accessD.jsp</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>/ticketbook/accessE.jsp</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>/ticketbook/check.jsp</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/list.jsp</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/ticketbook/profile</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>SELECT * FROM tickets WHERE ticket='OWASP'</td>
</tr>
<tr>
<td>INSERT INTO tickets(name,city,cc,ticket) VALUES('OWASP', 'Everywhere', '16/RPW701N3H9cJofut3ig==', '10006')</td>
</tr>
</tbody>
</table>
DEMO: PREVENTING COMMAND INJECTION EXPLOITS WITH INSTRUMENTATION!

sensors:
- name: "get-routes"
  description: "Identifies the route for this HTTP request"
  methods:
  - "javax.servlet.Servlet.service"
  captures:
  - "#P0.getRequestURI()"

- name: "block-native-process"
  description: "Blocks attempts to start native processes"
  methods:
  - "java.lang.ProcessBuilder.<init>"
  scopes:
  - "javax.servlet.Servlet.service"
  captures:
  - "#ARGS"
  exception: "Attempt to create ProcessBuilder from within Servlet.service() prevented by JST rule 'block-native-process'"

reports:
- name: "CMDi"
  type: "series"
  rows: "get-routes"
  cols: "block-native-process:13"
DEMO: PREVENTING COMMAND INJECTION

HTTP Status 500 – Internal Server Error

Type Exception Report

Message com.contrastsecurity.advice.SensorException: Attempt to create ProcessBuilder from within Servlet.service() prevented by JST rule 'block-native-process'

Description The server encountered an unexpected condition that prevented it from fulfilling the request.

Exception


org.apache.jasper.servlet.JspServletWrapper.handleJspException

org.apache.jasper.servlet.JspServletWrapper.service(JspServlet.java:741)

org.apache.jasper.servlet.JspServlet.serviceJspFile(JspServlet.java:427)

org.apache.jasper.servlet.JspServlet.service(JspServlet.java:381)

javax.servlet.http.HttpServlet.service(HttpServlet.java:741)

org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:53)
sensors:

- name: "get-routes"
description: "Identifies the route for this HTTP request"
methods:
  - "javax.servlet.Servlet.service"
captures:
  - "#P0.getRequestURI()"

- name: "sandbox-expressions"
description: "Prevents harmful methods from being used during expression evaluation"
methods:
  - "java.lang.ProcessBuilder.<init>"
  - "java.io.Socket.<init>"
scopes:
  - "javax.el.ValueExpression.getValue"
captures:
  - "#P0"
exception: "Attempt to escape expression language sandbox prevented by JST rule 'sandbox-expressions'"

reports:

- name: "Expression Language Injection Attempt Log"
type: "series"
rows: "get-routes"
cols: "sandbox-expressions:13"
DEMO: PREVENTING EXPRESSION LANGUAGE INJECTION

Runtime Protection!

HTTP Status 500 – Internal Server Error

Type Exception Report

Message javax.el.ELException: com.contrastsecurity.advice.SensorException: Attempt to escape expression language sandbox prevented by JST rule 'sandbox-expressions'

Description The server encountered an unexpected condition that prevented it from fulfilling the request.

Exception

org.apache.jasper.servlet.JspServletWrapper.handleJspException(JspServletWrapper.java:38)
org.apache.jasper.servlet.JspServletWrapper.service(JspServletWrapper.java:330)
org.apache.jasper.servlet.JspServlet.serviceJspFile(JspServlet.java:741)
org.apache.jasper.servlet.JspServlet.service(JspServlet.java:330)
javax.servlet.http.HttpServlet.service(HttpServlet.java:714)
org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)

Root Cause

javax.el.ELException: com.contrastsecurity.advice.SensorException: Attempt to escape expression language sandbox prevented by JST rule 'sandbox-expressions'

ProtocolHandler ["http-nio-8080"]
**IAST**
Interactive Application Security Testing detects vulnerabilities in both custom code and libraries during normal use.

**RASP**
Runtime Application Self-Protection detects attacks and prevents exploits in both custom code and libraries.
SECURITY WORKS BETTER FROM INSIDE-OUT

Yesterday: Scanning and firewalling at network layer

Today: Security Instrumentation means accuracy, speed, scalability
SECURITY INSTRUMENTATION STANDARDS

- **SA-11(9) | INTERACTIVE APPLICATION SECURITY TESTING**
  Require the developer of the system, system component, or system service to employ interactive application security testing (IAST) tools to identify flaws and document the results.

- **SI-7(17) | RUNTIME APPLICATION SELF-PROTECTION**
  Implement [Assignment: organization-defined controls] for application self-protection at runtime (RASP).

- **SSS 9.1 | RUNTIME APPLICATION SELF-PROTECTION**
  The software detects and alerts upon detection of anomalous behavior, such as changes in postdeployment configurations or obvious attack behavior.

- **SSS 10.2 | INTERACTIVE APPLICATION SECURITY TESTING**
  Vulnerabilities in the software and third-party components are tested for and fixed prior to release using ... [techniques including]... interactive application security testing (IAST) ....
TURNING SECURITY INTO CODE

...ACTUALLY, THE REAL VULNERABILITIES ARE IN THE PIPELINE ITSELF
APPSEC’S ABILITY TO DELIVER VALUE TO CUSTOMERS, FASTER

TRADITIONAL SECURITY (OUTSIDE – IN)

MODERN SECURITY (INSIDE – OUT)

Continuous automated security testing and exploit prevention

Accurate, Realtime Telemetry

Vulnerabilities

Attacks

Instrumentation Sensors

Self-Testing

Self-Protecting

CUSTOM CODE
LIBRARIES
FRAMEWORKS
APP SERVER
RUNTIME

DEV
TEST
STAGE
PROD

CI/CD

“AAAaah!”

Backlog

False Positives!

RUN SAST

TRIAGE

RUN SCA

CORRELATE

RUN DAST

PENTEST

WAF

FLOW
(BUSINESS VALUE)
COMMUNITY EDITION

A totally free and full-strength application security platform

CREATE FREE ACCOUNT

https://www.contrastsecurity.com/ce

AVAILABLE NOW
COMING SOON

Protect against attacks with RASP
Find vulnerabilities with IAST
Secure open-source with SCA
ASK ME ANYTHING

Jeff Williams, Cofounder and CTO
@planetlevel