

Understanding IAST

“Intrinsic” Application Security Testing

Jeff Williams, CEO

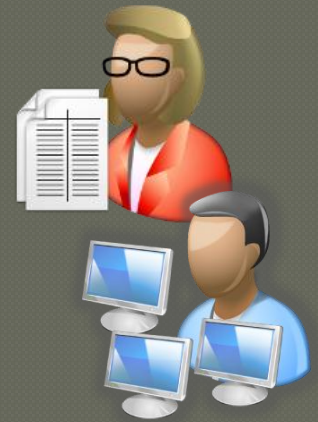
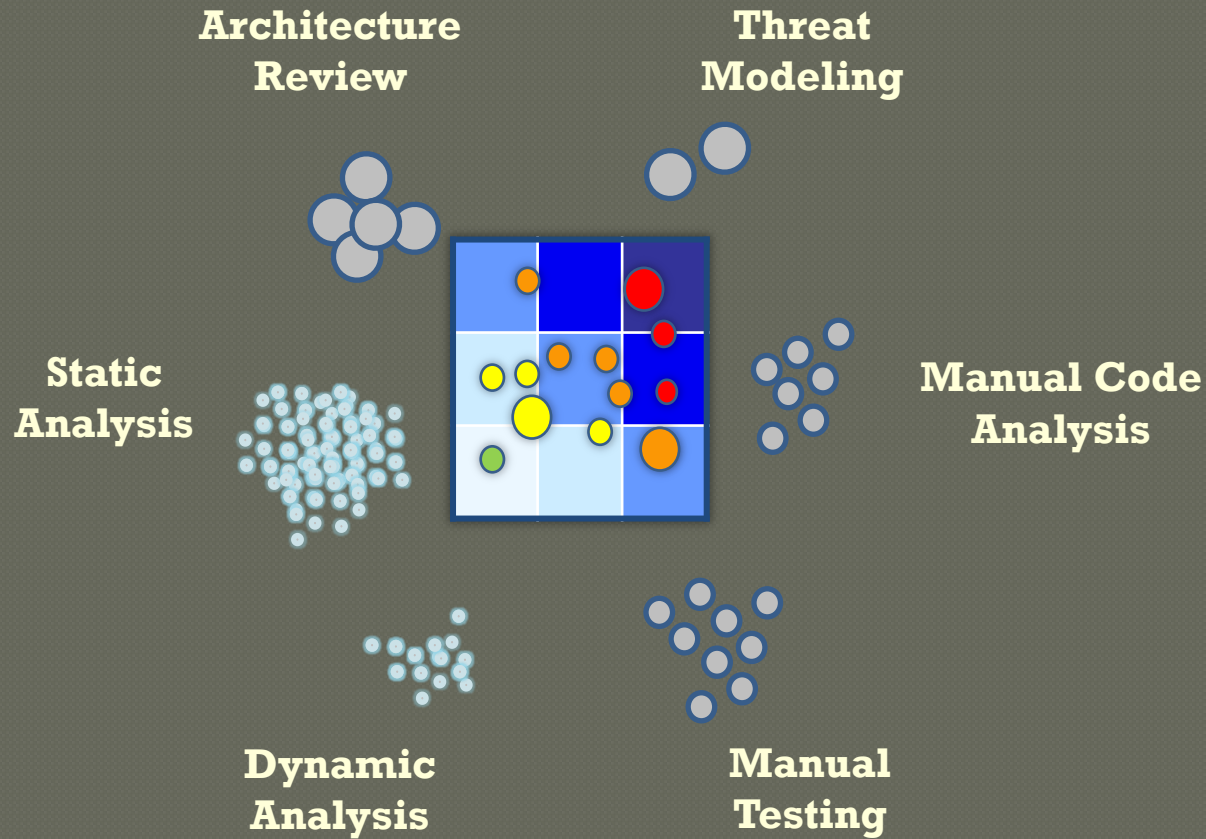
ASPECT **SECURITY**

Application Security Experts

OWASP AppSec DC

April 4, 2012





How do
we find
vulns?

Portfolio Assurance Strategies



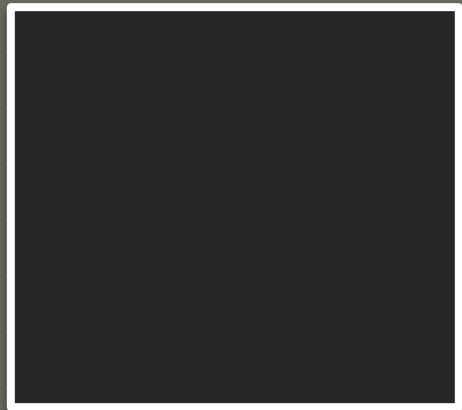
Scan



Manual



Spend



Pray

```
5|0|8|http://tester:8888/testapp  
/|9E4CB3D5635C548906BFB576DD18C7  
10|com.test.app.client.GreetingS  
ervice|greetServer|[Ljava.lang.S  
tring;/2600011424|hi|there|blah|  
1|2|3|4|1|5|5|3|6|7|8|%26ping%20  
-n%2020%20127.0.0.1%26
```

* GWT message courtesy GDS

Ajax

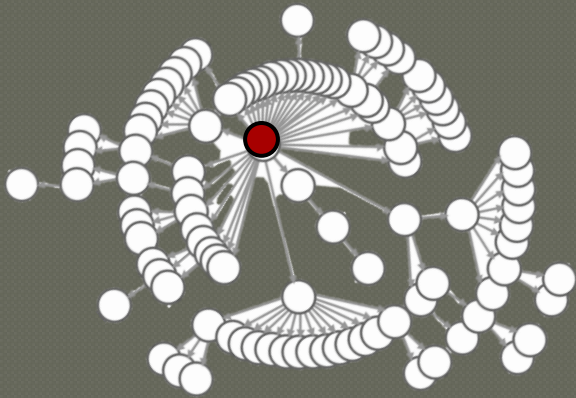
Web
Services

Serialized
Objects

Mobile

WebSocket

**Scanning and
pentesting are about
to get a LOT harder.**



Lines of
Code

Libraries
and
Frameworks

AOP

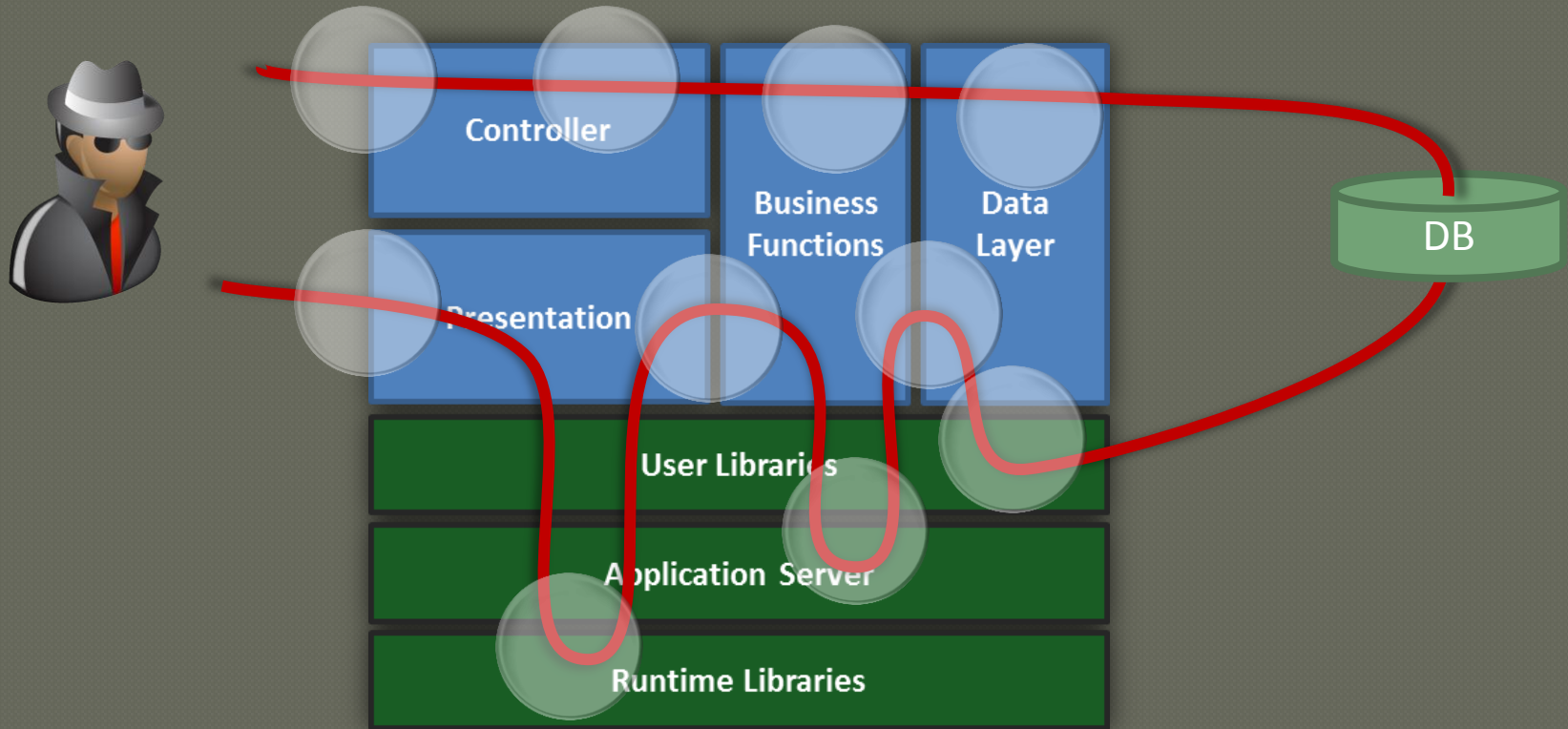
Custom
Controls

DevOps

**Static analysis and
code review are about
to get a LOT harder.**

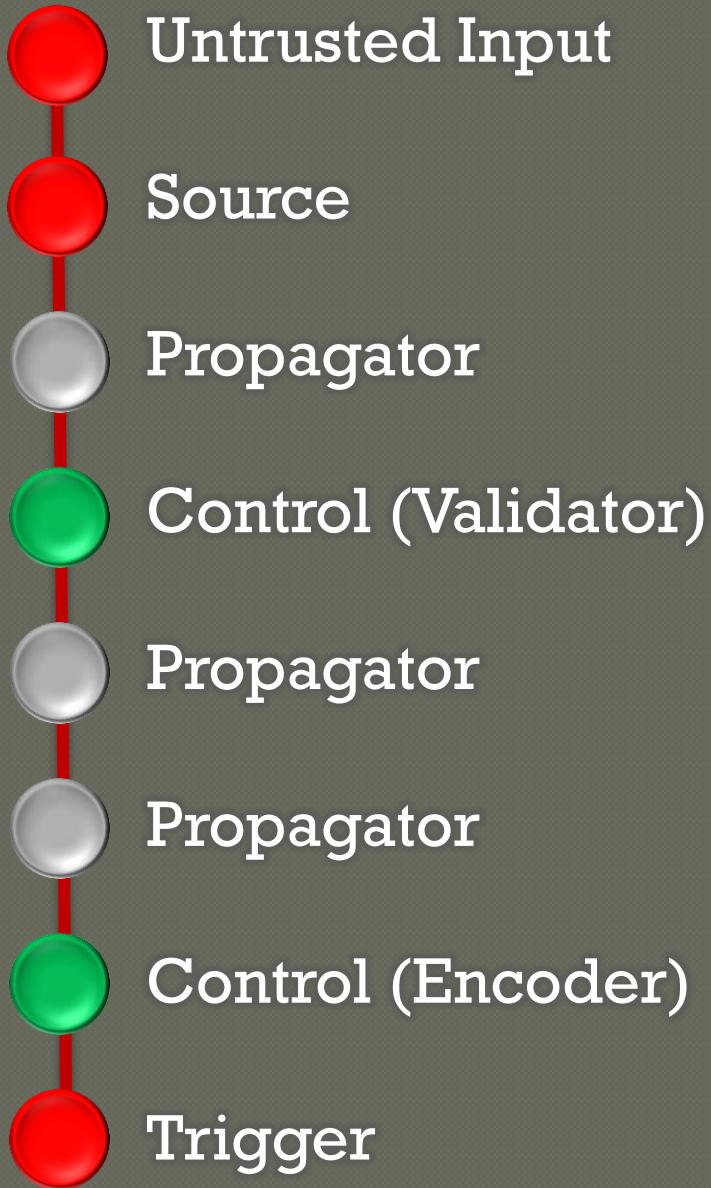
**We can do
better.**

**We have to do
better.**



What does a vuln look like?

Vulnerability Trace





The diagram features a central light pink circle labeled 'Problems'. To its right, a light pink curved line with three red circular markers points towards the circle. Each marker is associated with a text block describing a limitation of a specific security testing method.

Problems

Manual
pentesting
and DAST
can't see in

SAST and
code review
can't see out

No way to
map code to
HTTP

**Reimagining
the pentest.**



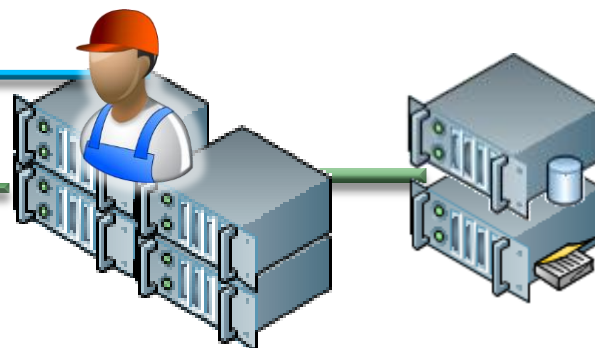
Parameter use
Session update
Dangerous call

“Manual” IAST



Security Intel

Application Tests



Test for XSS...
...HTML

Chrome File Edit View History Bookmarks Window Help

localhost:8080/WebGoat/attack?Screen=65&menu=1200&spy

ASDC12 2 of 8

ASPECT SECURITY
Application Security Experts

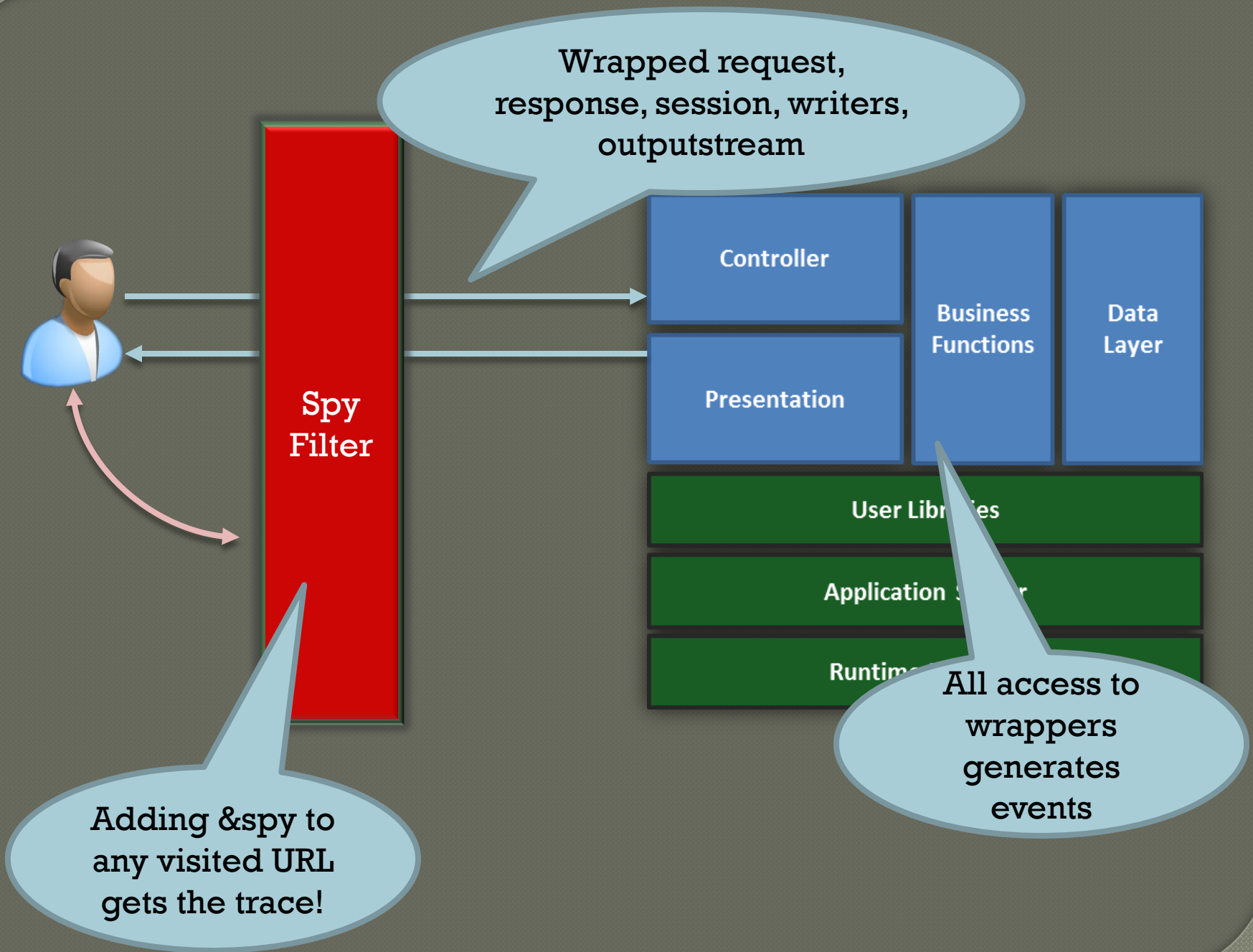
SpyFilter - a simple IAST demo

Events for /WebGoat/attack?Screen=65&menu=1200

METHOD	PARAMETER	RETURN VALUE	TRACE
session.getAttribute	Session	org.owasp.webgoat.session.WebSession@322b2057	javax.servlet.http.HttpServlet.service(HttpServlet.java:722)
request.getParameterValues	Screen	[65]	org.owasp.webgoat.session.ParameterParser.getIntParameter(ParameterParser.java:479)
request.getParameterValues	account_name	ASDC12	org.owasp.webgoat.lessons.SqlStringInjection.makeAccountLine(SqlStringInjection.java:203)
request.getParameterValues	Screen	[65]	org.owasp.webgoat.HammerHead.doPost(HammerHead.java:171)
request.getParameterValues	SUBMIT	[Go!]	org.owasp.webgoat.HammerHead.doPost(HammerHead.java:171)
request.getParameterValues	account_name	ASDC12	org.owasp.webgoat.HammerHead.doPost(HammerHead.java:171)
request.getParameterValues	menu	[1200]	org.owasp.webgoat.HammerHead.doPost(HammerHead.java:171)
request.getHeader	user-agent	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_7_3) AppleWebKit/535.19 (KHTML, like Gecko) Chrome/18.0.1025.142 Safari/535.19	javax.servlet.http.HttpServlet.service(HttpServlet.java:722)
session.setAttribute	websession	org.owasp.webgoat.session.WebSession@322b2057	javax.servlet.http.HttpServlet.service(HttpServlet.java:722)
session.setAttribute	course	org.owasp.webgoat.session.Course@2d58497c	javax.servlet.http.HttpServlet.service(HttpServlet.java:722)
session.getAttribute	welcomed	true	javax.servlet.http.HttpServlet.service(HttpServlet.java:641)
session.getAttribute	course	org.owasp.webgoat.session.Course@2d58497c	javax.servlet.http.HttpServlet.service(HttpServlet.java:722)
session.getAttribute	websession	org.owasp.webgoat.session.WebSession@322b2057	javax.servlet.http.HttpServlet.service(HttpServlet.java:722)
session.setHeader	Content-Length	329	javax.servlet.http.HttpServlet.service(HttpServlet.java:641)
servletWriter.print		<form accept-charset='UNKNOWN' method='POST' name='form' action='attack?Screen=65&menu=1200' enctype='><p>Enter your last name: <input name='account_name' type='TEXT' value='ASDC12'><input name='SUBMIT' type='SUBMIT' value='Go!'><pre>SELECT * FROM user_data WHERE last_name = 'ASDC12'</pre>No results matched. Try Again.</form>	org.owasp.webgoat.HammerHead.doPost(HammerHead.java:194)

Another free and open tool!

<https://www.aspectsecurity.com/spyfilter/>



**Better
scanning.**



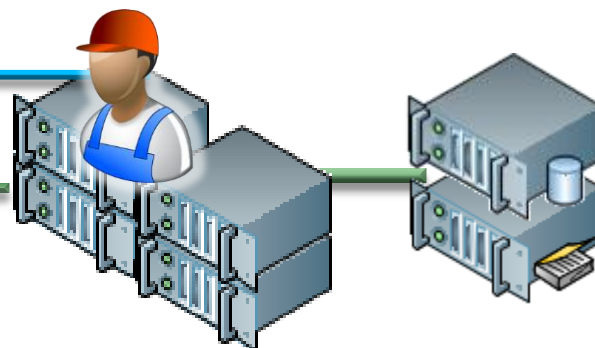
Queries,
Exceptions,
Logs...

“Basic” IAST



Results

Application Tests



Test for SQLi...
...HTML

“Wrap the Sink”

- HP WebInspect SecurityScope
- IBM GlassBox
- Acunetix

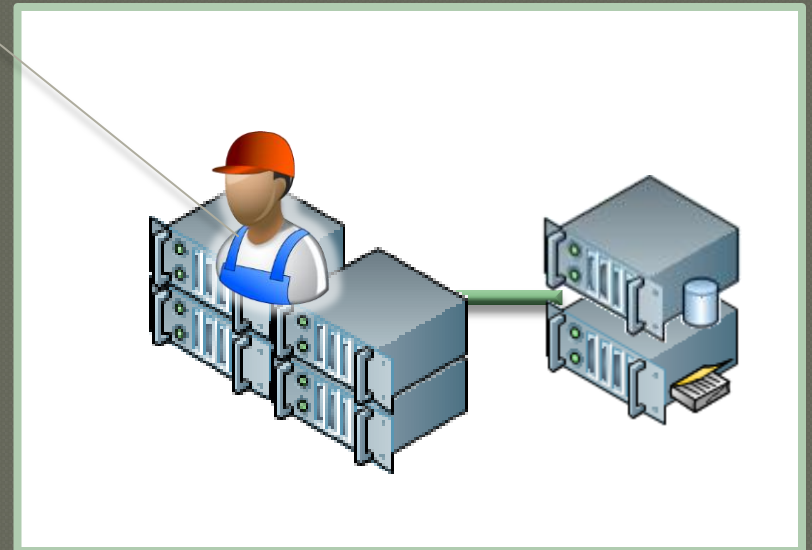


Basic IAST Benefits

1. Improve DAST Coverage

2. Validate DAST Vulnerabilities

3. Correlate with Code for DAST Findings

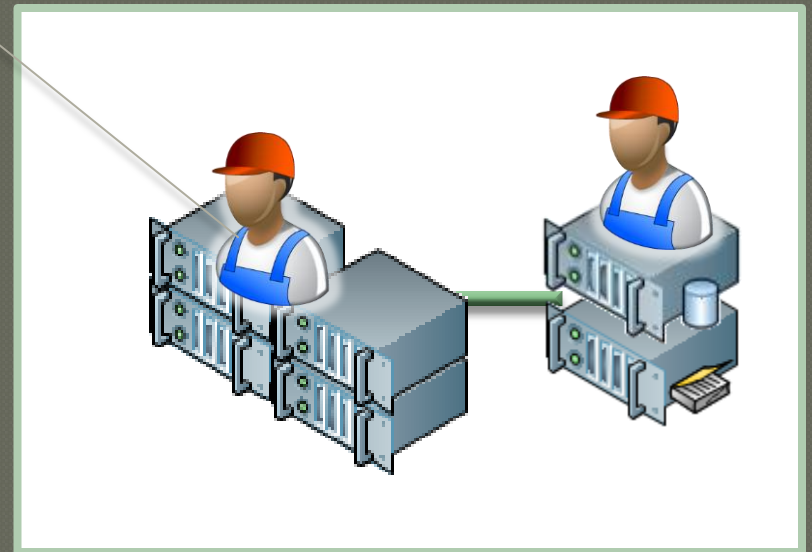


**Architecture
review?**

“Advanced” IAST

Basic IAST plus:

- All libraries used
- Exact SLOC count
- Backend connections
- System configuration
- Security controls
- Directory structure
- Entry points



Instrumentation Techniques

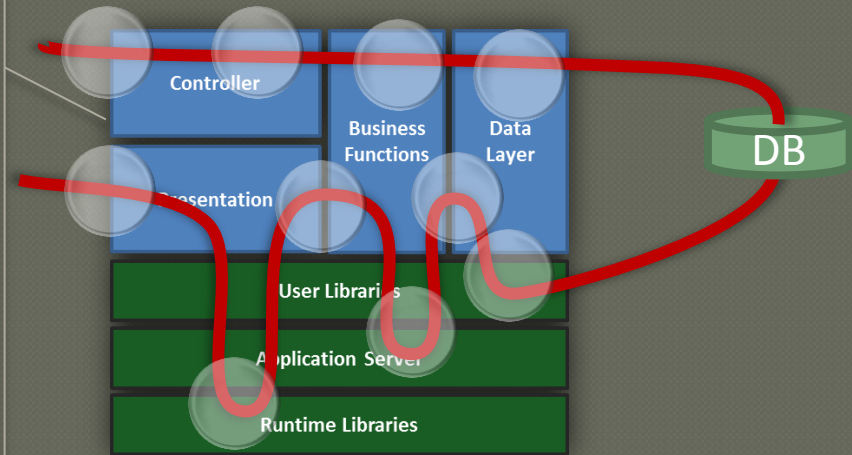
- ◉ Add calls to source code
- ◉ Use Aspect-oriented programming
- ◉ Modify class files on disk
- ◉ Modify bytecode of running application with “Instrumentation API”

The Future!

“Pure” IAST

Detailed IAST plus:

- No SAST/DAST
- Powerful rule engine
- Easy install
- Data flow analysis
- Continuous security
- Leverage QA testers



Aspect
“Contrast” in
private beta

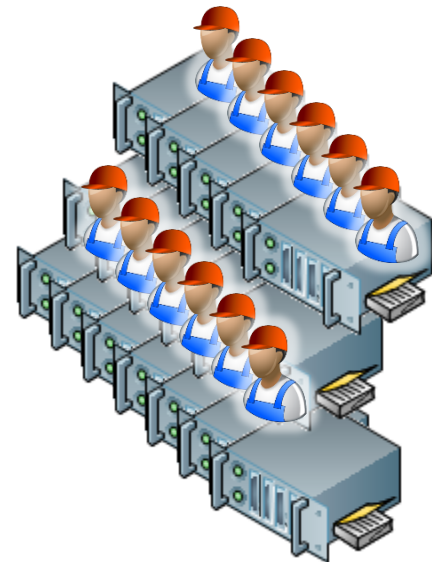
Continuous Security!

The Future of IAST

Automatic:

- Portfolio (prioritized)
- Libraries (analyzed)
- Architecture (summary)
- Vulnerabilities (traced)

GOAL: continuous testing with an enterprise ruleset!



Instrumented
Enterprise

Jeff Williams

Aspect Security CEO

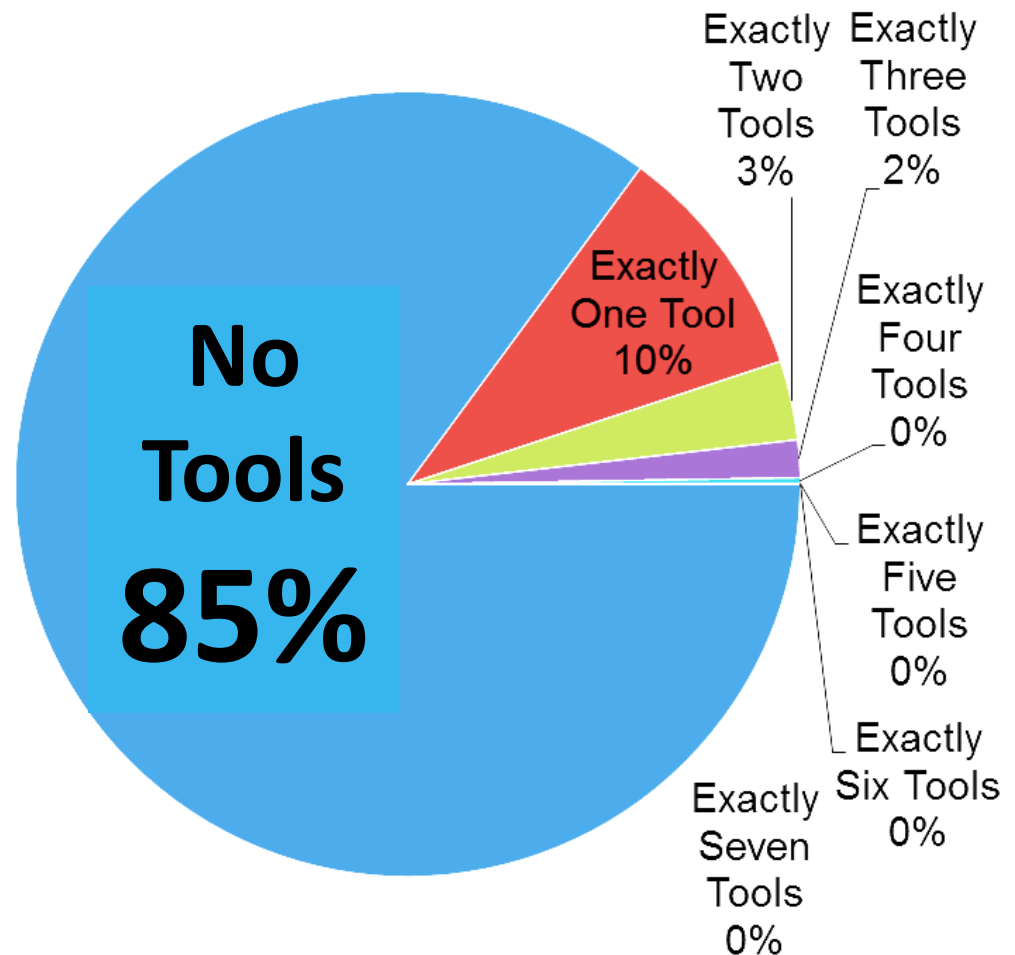
jeff.williams@aspectsecurity.com

<http://www.aspectsecurity.com>

NSA Center for Assured Software



- Seven tools
- 13,801 Test Cases
- 527 flaw types
- Various data and control flows
- 85% of problems were not “discriminated” by ANY tools



<http://www.appsecusa.org/p/nsacas.pdf>

Results with False Alarms

