SOFTWARE SECURITY ASSURANCE SUMMIT

December 1, 2010 | Westin Tysons Corner | Falls Church, VA

T.E.N. presented by FORTIFY An HP Company

Product Roadmap

Sushant Rao
Principal Product Manager
Fortify Software, a HP company

Agenda

- **Next Generation of Security Analysis**
- **Future Directions**

Currently under investigation and not guaranteed to be included in future releases

Next Generation of Security Analysis

A Key Element in SSA is Security Testing

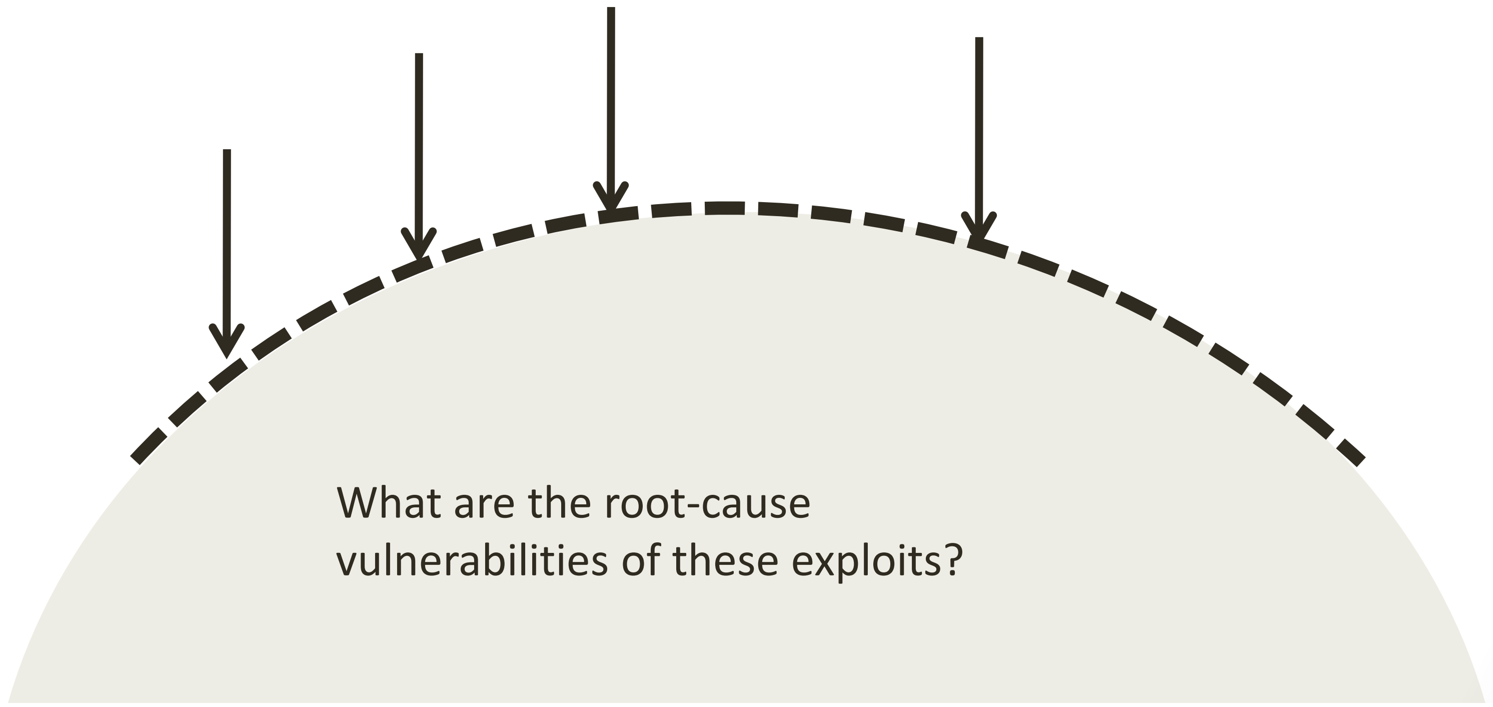
Which is the "best" Security Testing Methodology?

Dynamic Security Testing

Static Security Testing

Dynamic Testing

Dynamic Testing identifies Exploits



What are the root-cause vulnerabilities of these exploits?

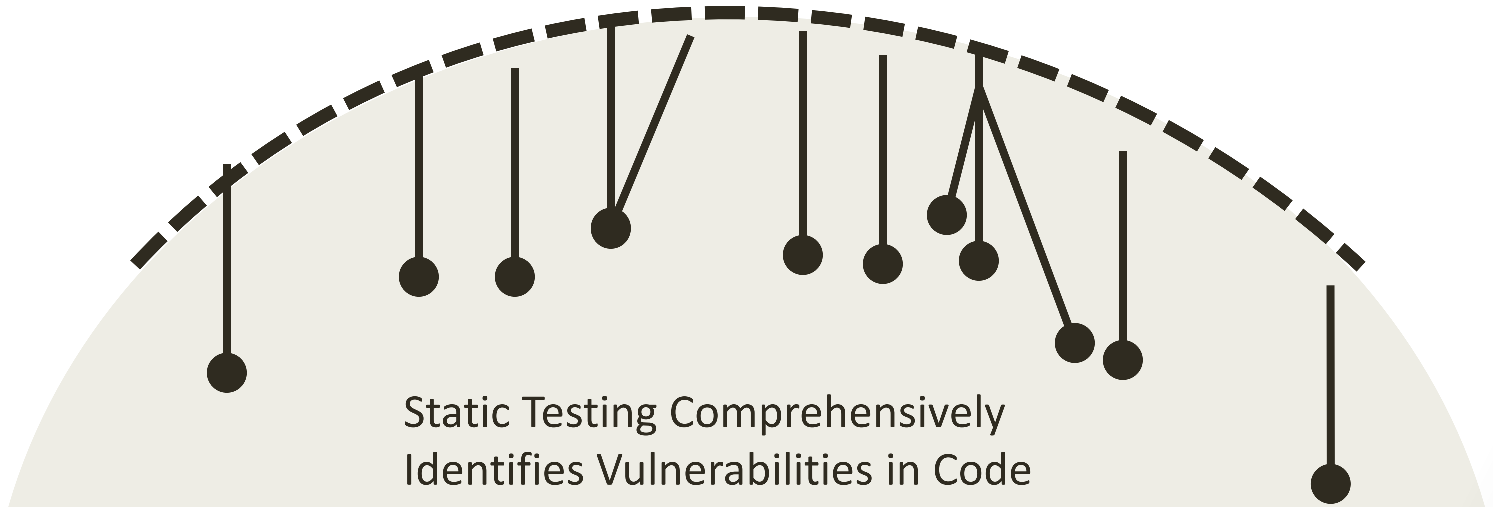
Dynamic Testing – Pros & Cons

Dynamic Security Testing

- Advantages
  - Concrete prioritization of results
  - Tests deployment environment
- Disadvantages
  - Little insight into root cause
  - Limited by functional coverage

Static Testing

Which vulnerabilities are accessible from the outside?



Static Testing Comprehensively Identifies Vulnerabilities in Code

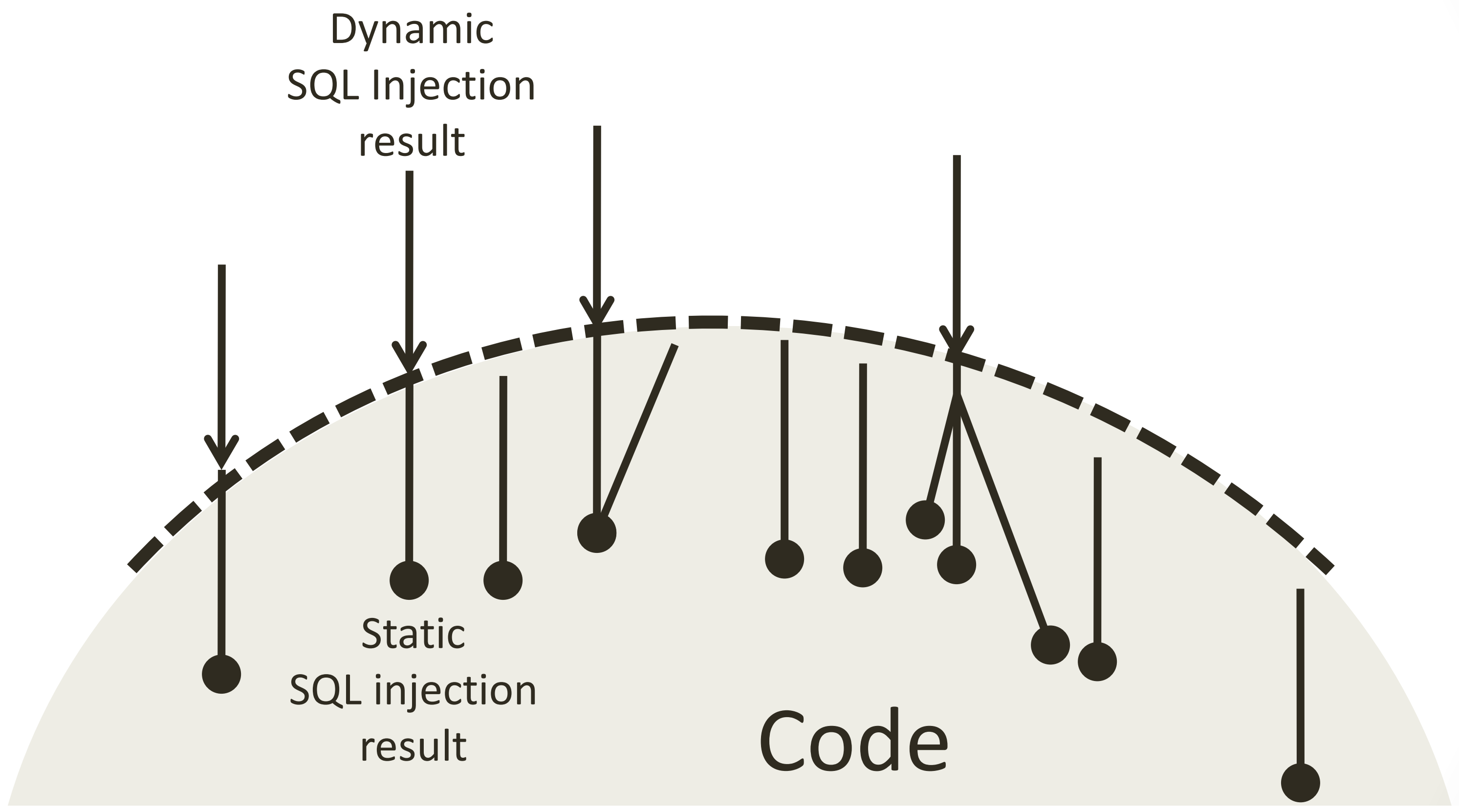
Static Testing – Pros & Cons

Static Security Testing

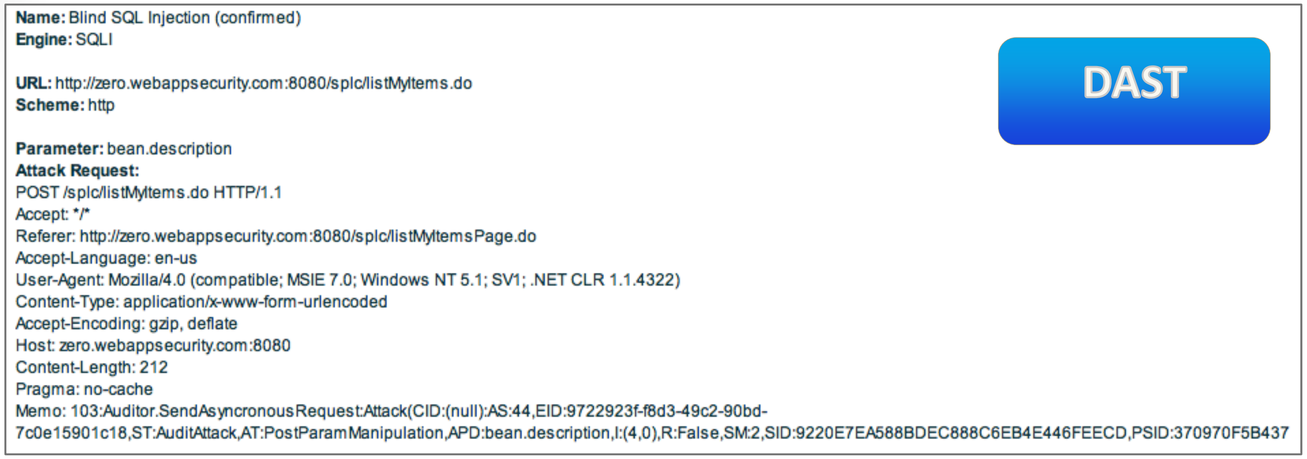
- Advantages
  - Comprehensive results
  - Source-level details
- Disadvantages
  - Exploits are difficult to provide
  - Prioritization difficult

Hybrid Technology

Correlates Exploits with Vulnerabilities



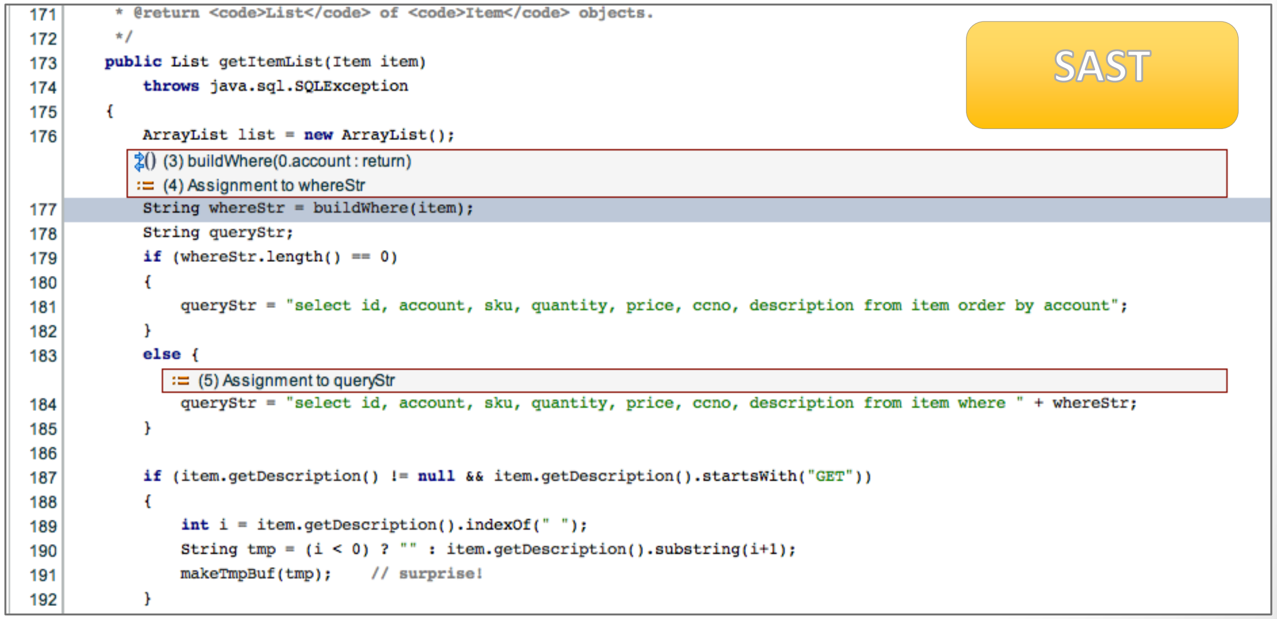
Challenge of Hybrid 1.0 Technology



Name: Blind SQL Injection (confirmed)
Engine: SQLI

URL: http://zero.webappsecurity.com:8080/splc/listMyItems.do
Scheme: http

Parameter: bean.description
Attack Request:
POST /splc/listMyItems.do HTTP/1.1
Accept: */*
Referer: http://zero.webappsecurity.com:8080/splc/listMyItemsPage.do
Accept-Language: en-us
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
Host: zero.webappsecurity.com:8080
Content-Length: 212
Pragma: no-cache
Memo: 103:Auditor.SendAsyncronousRequest:Attack(CID:(null):AS:44,EID:9722923f-f8d3-49c2-90bd-7c0e15901c18,ST:AuditAttack,AT:PostParamManipulation,APD:bean.description,I:(4,0),R:False,SM:2,SID:9220E7EA588BDEC888C6EB4E446FEECD,PSID:370970F5B437



```
     * @return <code>List</code> of <code>Item</code> objects.
     */
    public List getItemList(Item item)
        throws java.sql.SQLException
    {
        ArrayList list = new ArrayList();
          () (3) buildWhere(0.account : return)
          := (4) Assignment to whereStr
        String whereStr = buildWhere(item);
        String queryStr;
        if (whereStr.length() == 0)
        {
            queryStr = "select id, account, sku, quantity, price, ccno, description from item order by account";
        }
        else {
            := (5) Assignment to queryStr
            queryStr = "select id, account, sku, quantity, price, ccno, description from item where " + whereStr;
        }

        if (item.getDescription() != null && item.getDescription().startsWith("GET"))
        {
            int i = item.getDescription().indexOf(" ");
            String tmp = (i < 0) ? "" : item.getDescription().substring(i+1);
            makeTmpBuf(tmp);    // surprise!
        }
```

Correlating URLs (DAST) with Source Code (SAST) is difficult!

Problems With Hybrid 1.0

Inaccurate

- Correlation is difficult
- DAST provides URL, but SAST provides code-level data flow

Inefficient

- Securing applications become very time and resource intensive

Ineffective

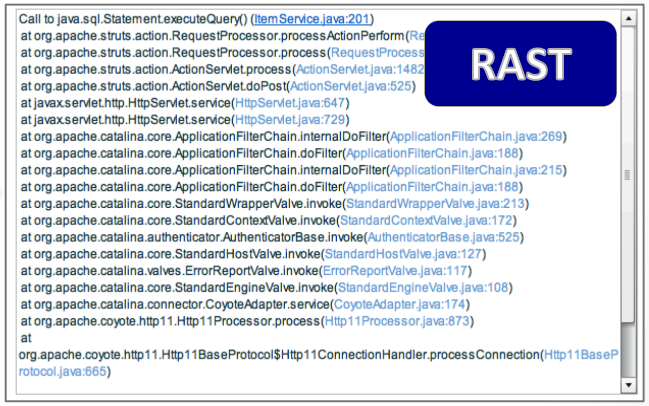
- No clear benefits to current approach
- As a result, users don't bother doing Hybrid Security Testing

Need a way to correlate Dynamic & Static testing

Introducing RAST for Intelligent Correlation

Runtime Security Testing

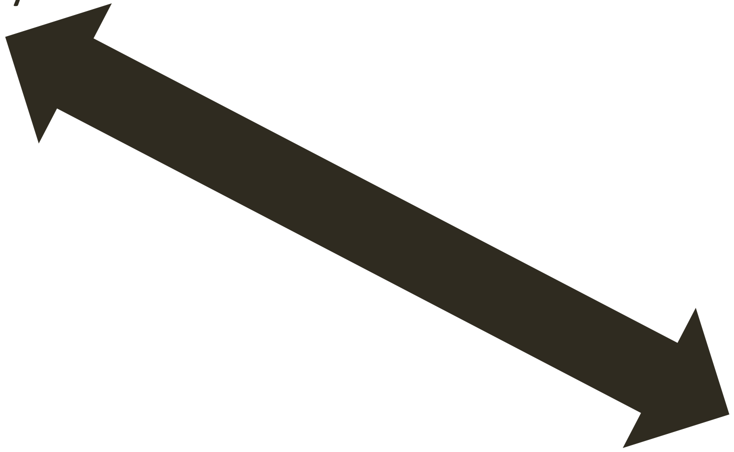
- Observe actual attacks
- Sidestep security controls
  - Obfuscation
  - Encryption



```
Call to java.sql.Statement.executeQuery() (ItemService.java:201)
 at org.apache.struts.action.RequestProcessor.processActionPerform(Re
 at org.apache.struts.action.RequestProcessor.process(RequestProcess
 at org.apache.struts.action.ActionServlet.process(ActionServlet.java:1482
 at org.apache.struts.action.ActionServlet.doPost(ActionServlet.java:525)
 at javax.servlet.http.HttpServlet.service(HttpServlet.java:647)
 at javax.servlet.http.HttpServlet.service(HttpServlet.java:729)
 at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:269)
 at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:188)
 at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:215)
 at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:188)
 at org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:213)
 at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:172)
 at org.apache.catalina.authenticator.AuthenticatorBase.invoke(AuthenticatorBase.java:525)
 at org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:127)
 at org.apache.catalina.valves.ErrorReportValve.invoke(ErrorReportValve.java:117)
 at org.apache.catalina.core.StandardEngineValve.invoke(StandardEngineValve.java:108)
 at org.apache.catalina.connector.CoyoteAdapter.service(CoyoteAdapter.java:174)
 at org.apache.coyote.http11.Http11Processor.process(Http11Processor.java:873)
 at
org.apache.coyote.http11.Http11BaseProtocol$Http11ConnectionHandler.processConnection(Http11BaseP
rotocol.java:665)
```

RAST is the key to correlation

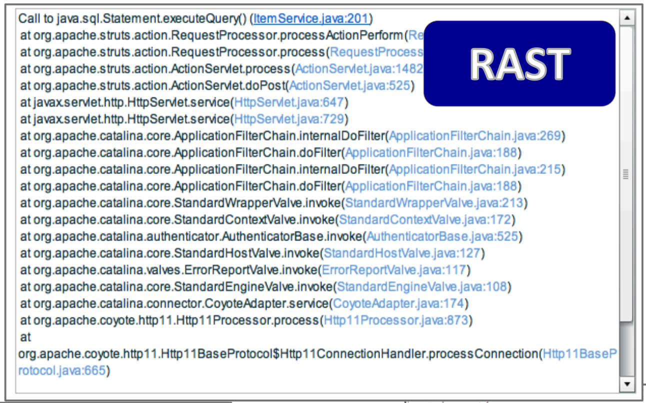
URL:
www.sales.company.
com



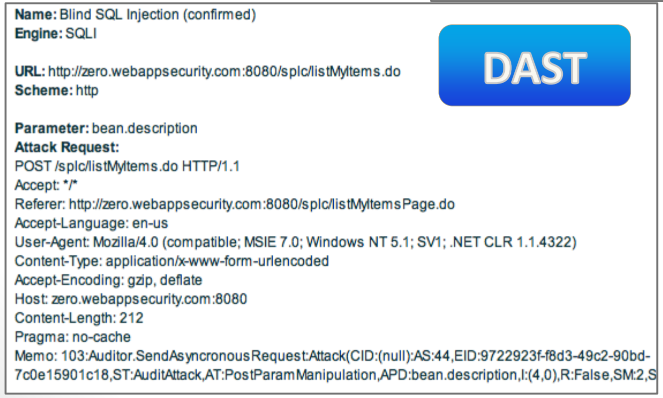
Source Code:
<java.sql.Connection
.xxx>

Introducing Hybrid 2.0 Technology

Hybrid 2.0 = HP WebInspect + Fortify SecurityScope + Fortify SCA



```
Call to java.sql.Statement.executeQuery() (ItemService.java:201)
 at org.apache.struts.action.RequestProcessor.processActionPerform(Re
 at org.apache.struts.action.RequestProcessor.process(RequestProcess
 at org.apache.struts.action.ActionServlet.process(ActionServlet.java:1482
 at org.apache.struts.action.ActionServlet.doPost(ActionServlet.java:525)
 at javax.servlet.http.HttpServlet.service(HttpServlet.java:647)
 at javax.servlet.http.HttpServlet.service(HttpServlet.java:729)
 at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:269)
 at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:188)
 at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:215)
 at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:188)
 at org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:213)
 at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:172)
 at org.apache.catalina.authenticator.AuthenticatorBase.invoke(AuthenticatorBase.java:525)
 at org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:127)
 at org.apache.catalina.valves.ErrorReportValve.invoke(ErrorReportValve.java:117)
 at org.apache.catalina.core.StandardEngineValve.invoke(StandardEngineValve.java:108)
 at org.apache.catalina.connector.CoyoteAdapter.service(CoyoteAdapter.java:174)
 at org.apache.coyote.http11.Http11Processor.process(Http11Processor.java:873)
 at
org.apache.coyote.http11.Http11BaseProtocol$Http11ConnectionHandler.processConnection(Http11BaseP
rotocol.java:665)
```

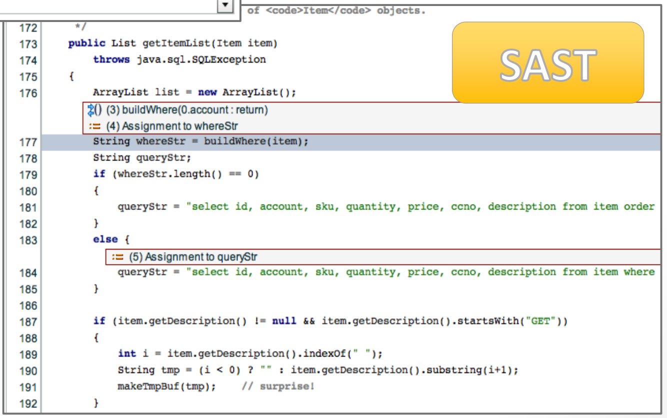


Name: Blind SQL Injection (confirmed)
Engine: SQLI

URL: http://zero.webappsecurity.com:8080/splc/listMyItems.do
Scheme: http

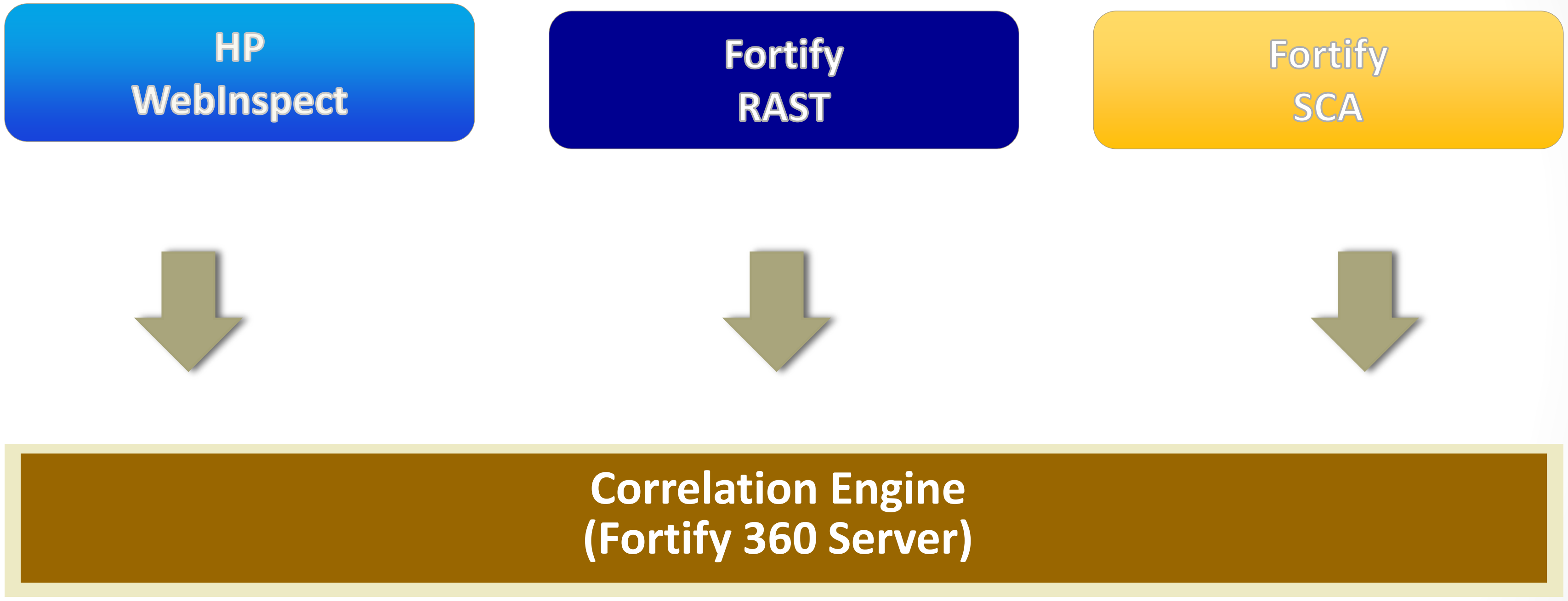
Parameter: bean.description
Attack Request:

```
POST /splc/listMyItems.do HTTP/1.1
Accept: */*
Referer: http://zero.webappsecurity.com:8080/splc/listMyItemsPage.do
Accept-Language: en-us
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
Host: zero.webappsecurity.com:8080
Content-Length: 212
Pragma: no-cache
Memo: 103:Auditor.SendAsyncronousRequest:Attack(CID:(null):AS:44,EID:9722923f-f8d3-49c2-90bd-
7c0e15901c18,ST:AuditAttack,AT:PostParamManipulation,APD:bean.description,I:(4,0),R:False,SM:2,S
```



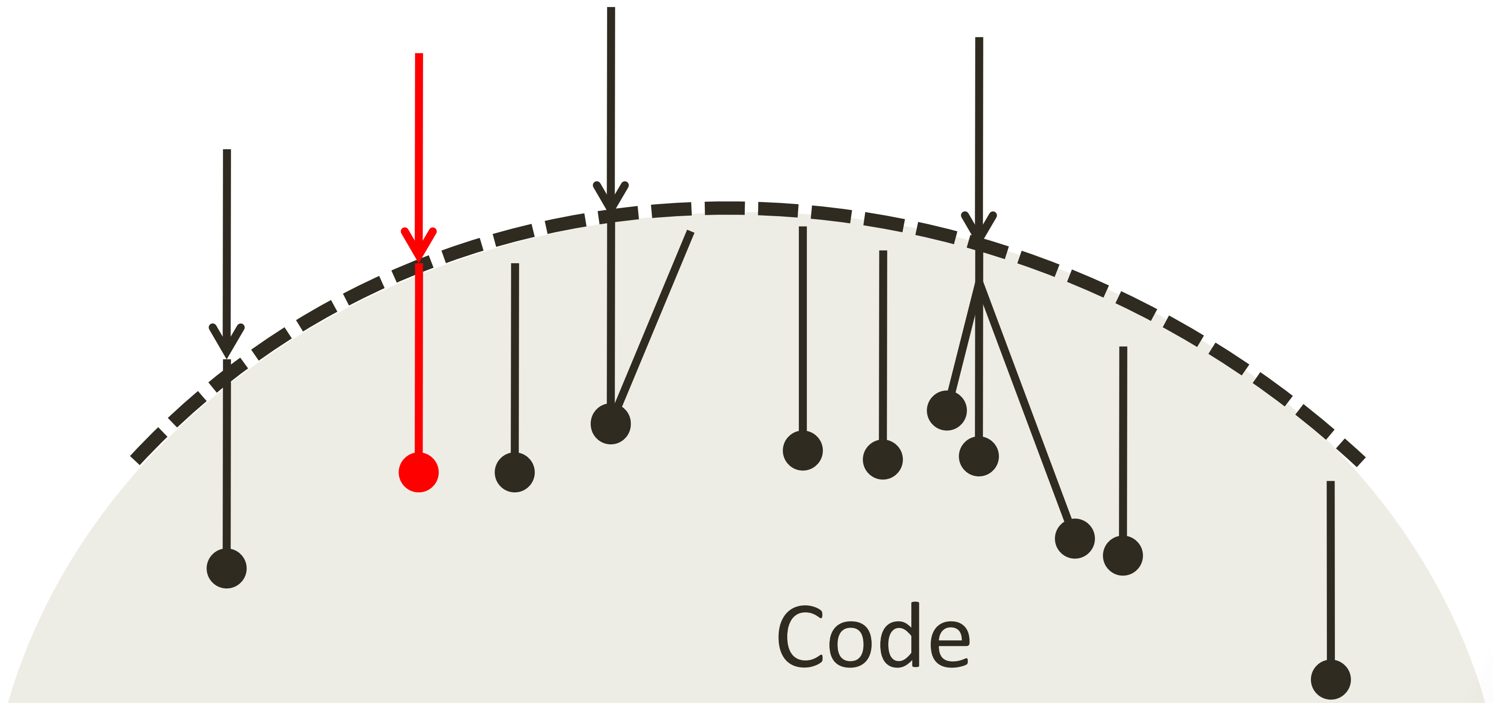
```
     of <code>Item</code> objects.
     */
    public List getItemList(Item item)
        throws java.sql.SQLException
    {
        ArrayList list = new ArrayList();
      () (3) buildWhere(0.account : return)
      := (4) Assignment to whereStr
        String whereStr = buildWhere(item);
        String queryStr;
        if (whereStr.length() == 0)
        {
            queryStr = "select id, account, sku, quantity, price, ccno, description from item order
        }
        else {
         := (5) Assignment to queryStr
            queryStr = "select id, account, sku, quantity, price, ccno, description from item where
        }

        if (item.getDescription() != null && item.getDescription().startsWith("GET"))
        {
            int i = item.getDescription().indexOf(" ");
            String tmp = (i < 0) ? "" : item.getDescription().substring(i+1);
            makeTmpBuf(tmp);    // surprise!
        }
```

Fortify Hybrid 2.0 Technology



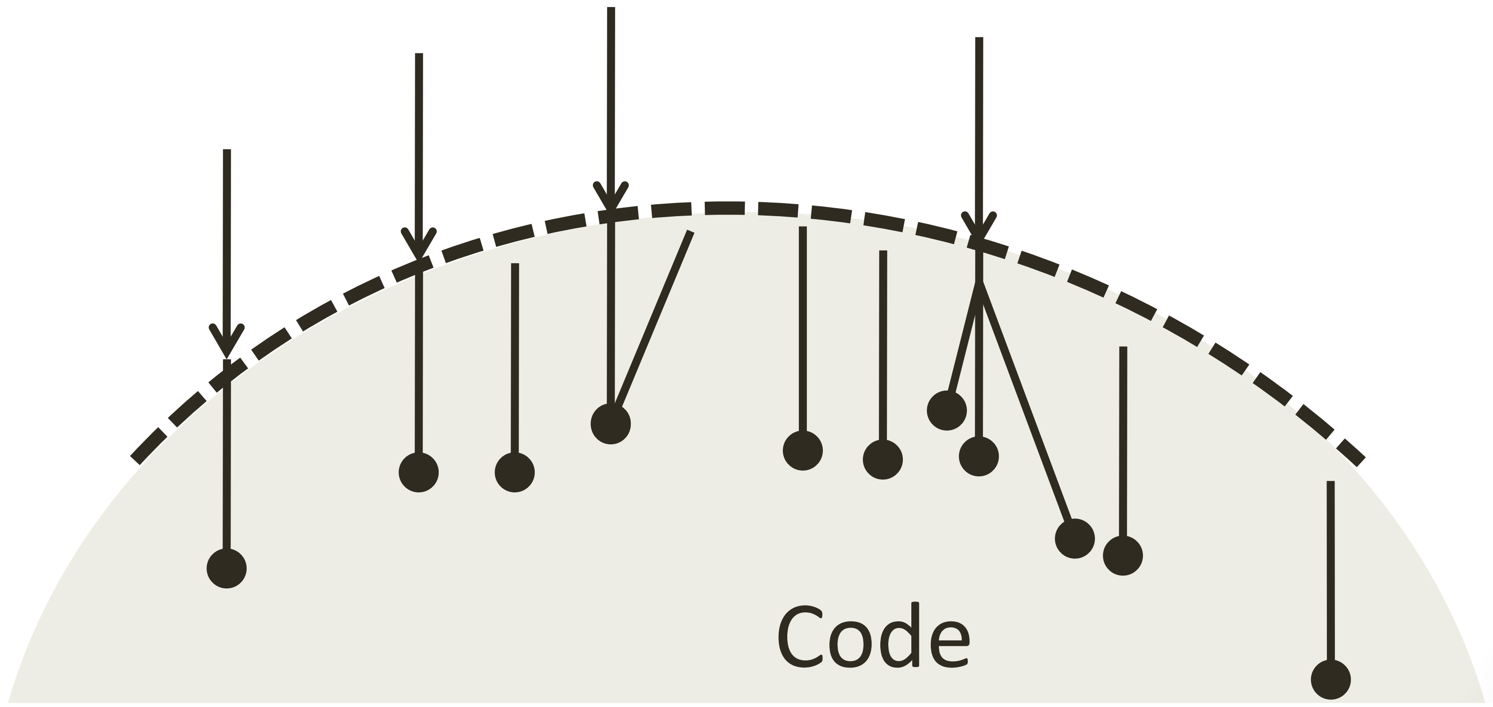
Hybrid 2.0 Technology

Directly links more vulnerabilities



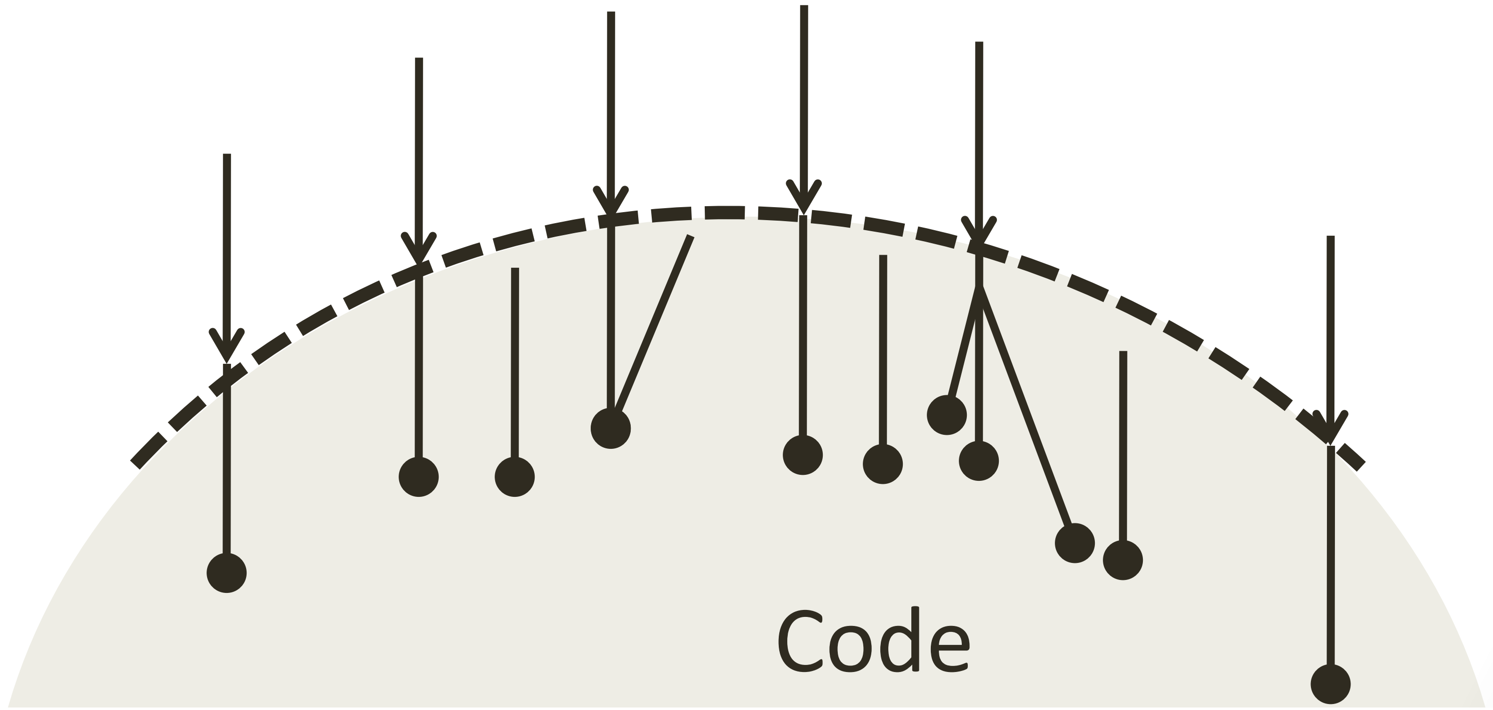
Hybrid 2.0 Technology

Correlation re-prioritizes riskier issues



Hybrid 2.0 Technology

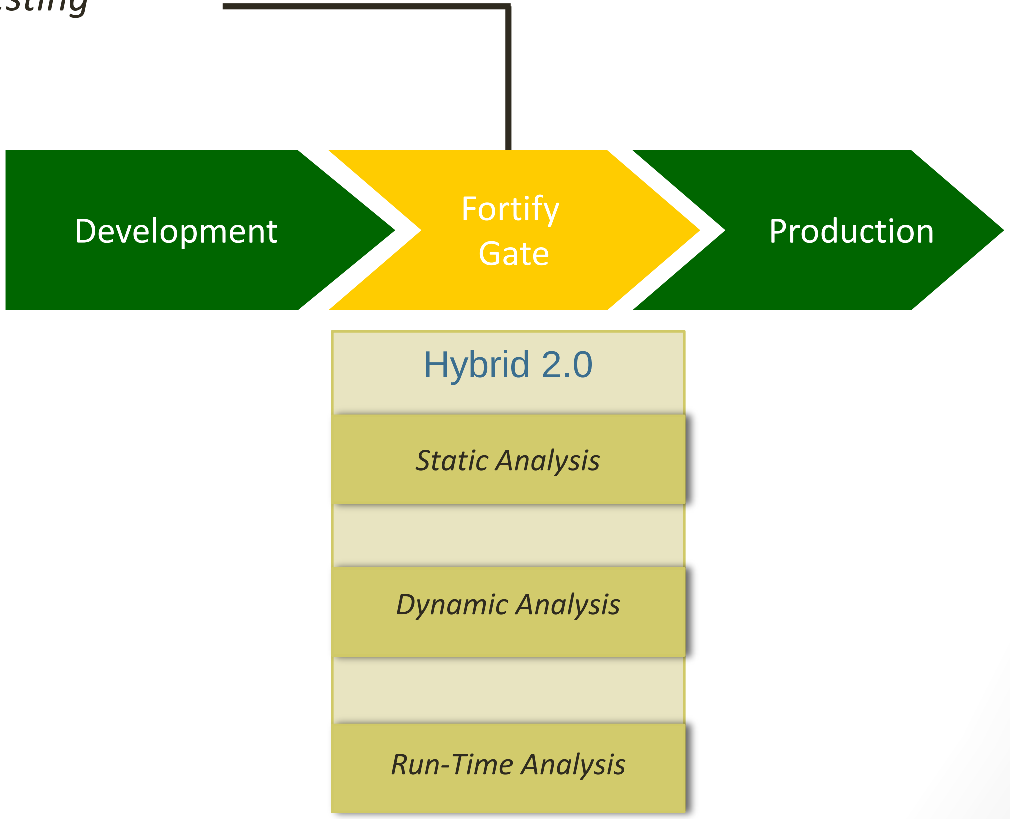
Direct dynamic testing



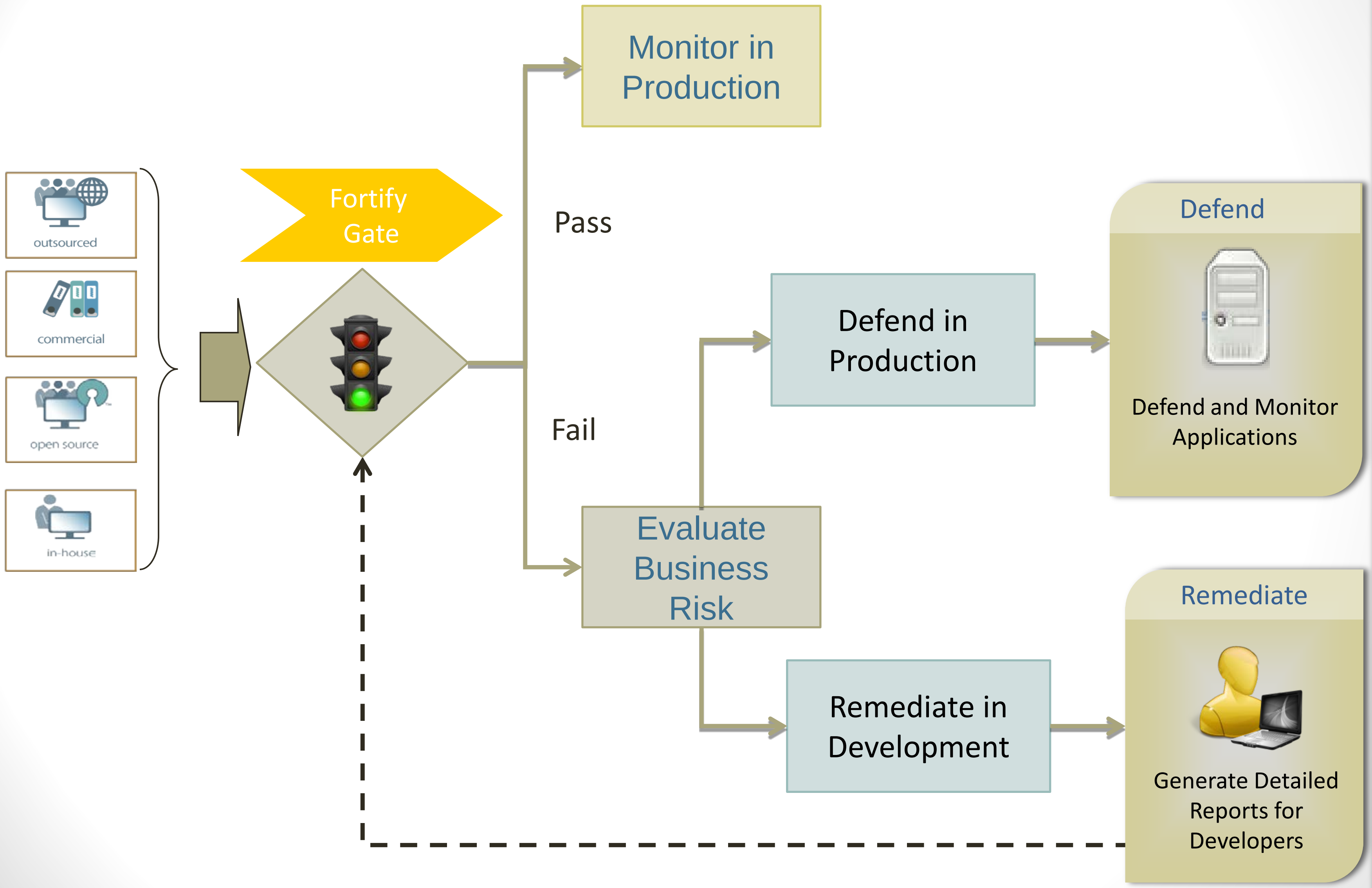
Deploying Hybrid 2.0

Step 1: Implement A Security Gate

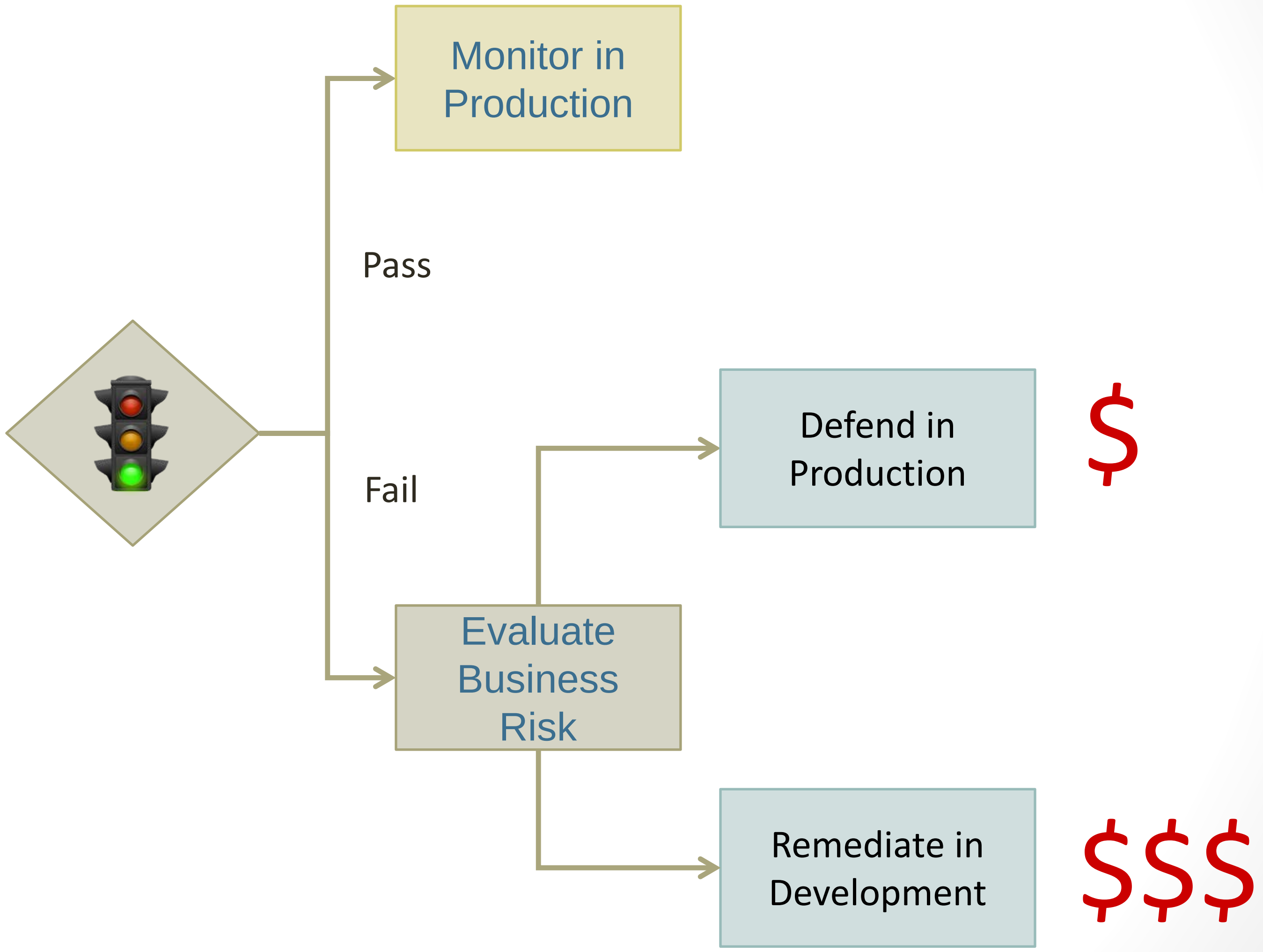
Security acceptance testing



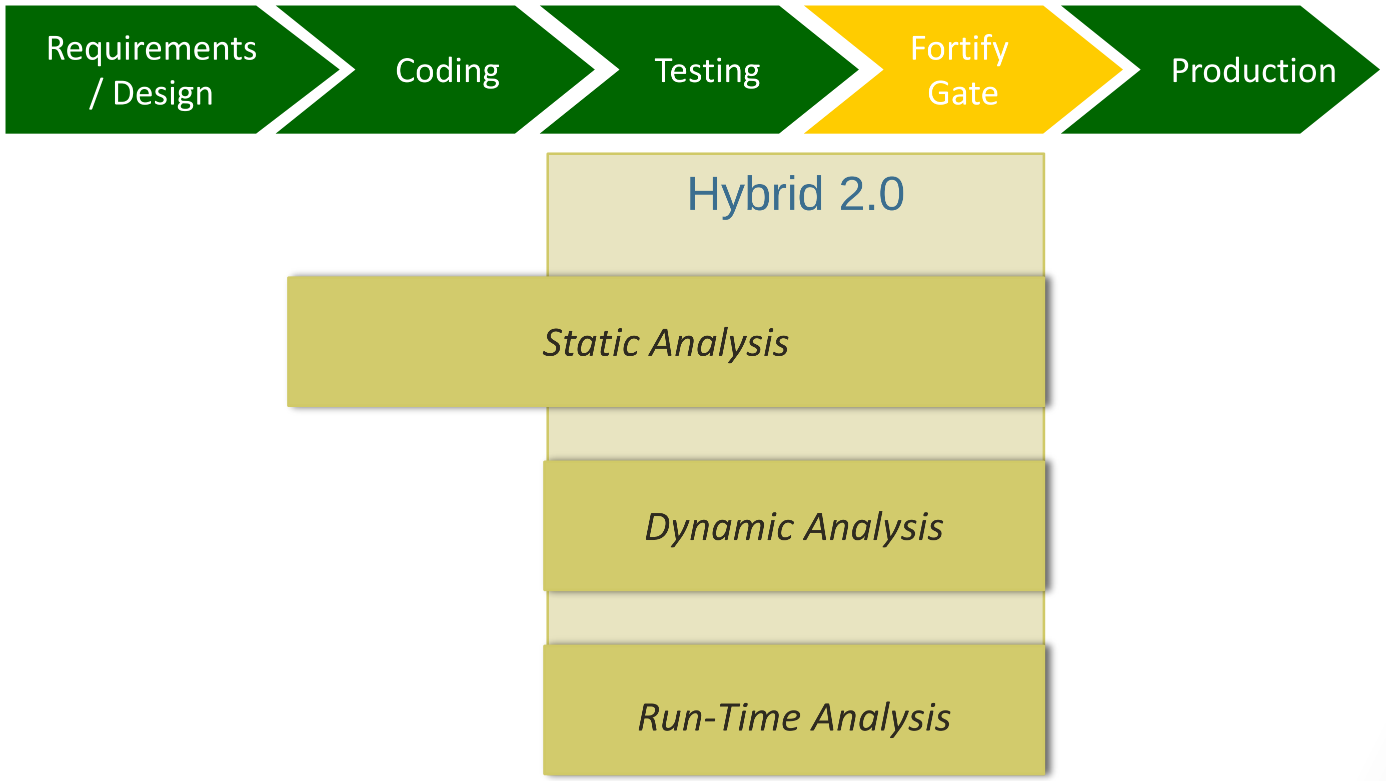
Fortify Security Gate with Hybrid 2.0



Issue with Step 1: Costs of Failing



Step 2: Expand to earlier stages in SDLC



Benefits of Fortify Hybrid 2.0

Relevance	▪ Find the root cause ▪ Understand the context of vulnerabilities
Importance	▪ Fix the most critical vulnerabilities ▪ Prioritize your resources and time
Speed	▪ Fix security issues fast ▪ Release secure applications to market quickly

Future Direction*

Currently under investigation and not guaranteed to be included in future releases

Security ▶ Languages

- Currently
  - Support 18 Languages: ASP.NET, VB.NET, C#, Java, JSP, C, C++, COBOL, Cold Fusion, T-SQL, PL/SQL, JavaScript / AJAX, Classic ASP, PHP, Python, VBScript, Visual Basic, XML / HTML
  - Under Development: SAP ABAP

Findings: Groups of Related Issues

- Correlation
  - Is a way to automatically group issues based on rules
- Findings
  - Will allow you to manually group issues during the audit process
  - Create your own findings (groups), drag and drop issues into them as you see fit
  - Correlation could turn into an initial seeding for findings
- Benefits
  - Save time by mass auditing issues
- Bugtrackers
  - Will be an important part of findings. We will provide an easy way to file a bug for several issues at once.

Security Education Plugin

- Working on a plugin that can alert you to security vulnerabilities in real time as you're developing code
  - i.e. when you start typing in "java.sql.Connection.PrepareCall()", you'll see a popup that alerts you to the security vulnerabilities that are related to that API
- Security information will come from our rules
  - Parsed/cached at plugin startup
- Looking at two different use cases: on-the-fly (alerts as you type), and on-demand (show all alerts for the current file)
- Several IDEs, will probably start with Eclipse
- Separate from our existing plugins, but can be used together

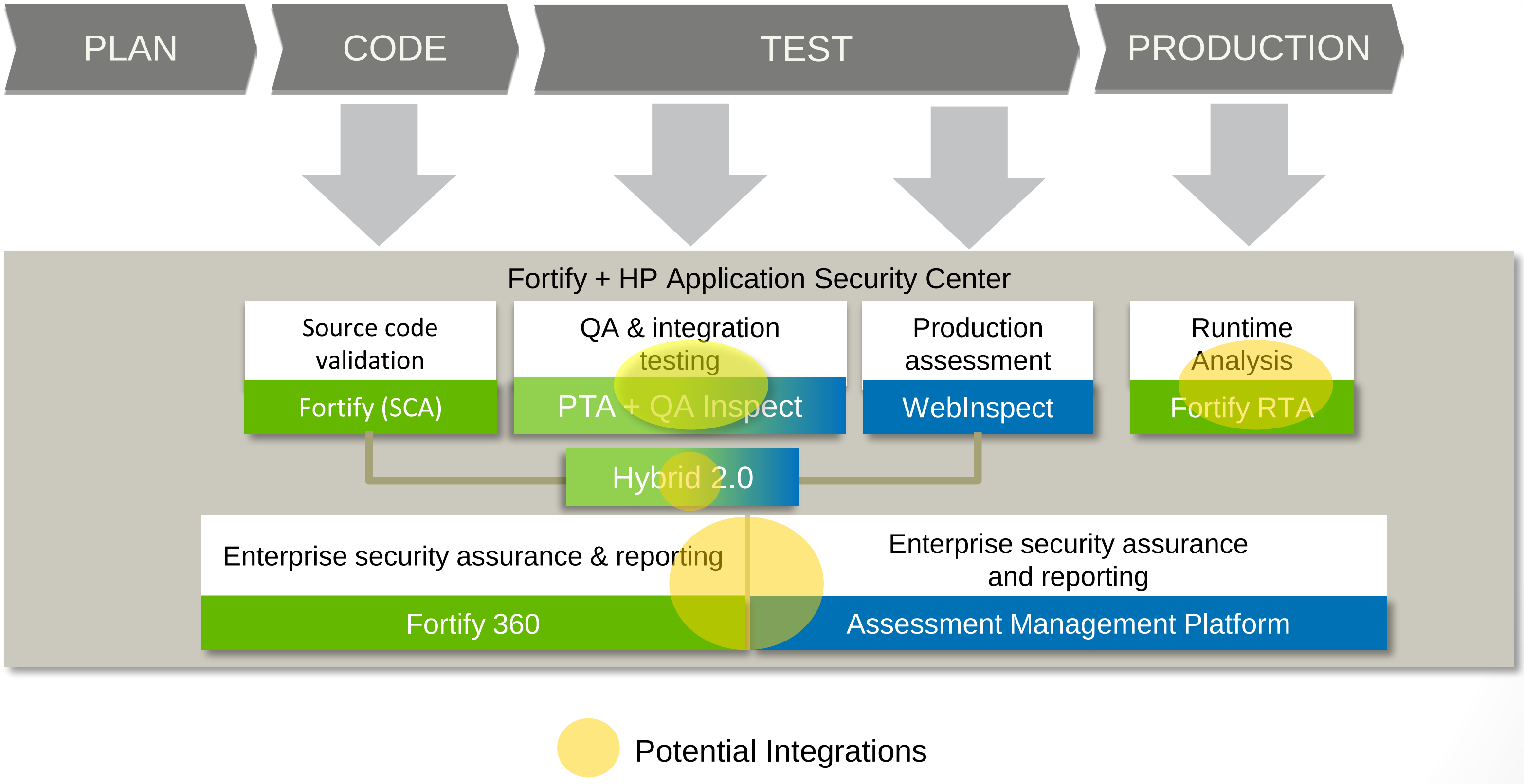
Easy & Fast

- Better Defect Tracking Integration
- Improved Scanning Performance
- Seamless Build Integration
- “Lighter-weight” plug-ins for Developer IDEs

Potential Fortify – HP Integrations

- Hybrid 2.0: DAST, SAST & RAST integration
- Defect Tracking: HP Quality Center & Fortify 360 Server
- Functional & Security Testing: HP QA Inspect & Fortify RAST
- Security Dashboard: Fortify 360 Server & HP AMP

Potential Fortify – HP Integrations



Thank you

Key Enhancements Released in 2010

- 2.6.0
  - RTA for Java 1.4
  - RTA for .NET 2.0, 3.0, and 3.5
  - IDE Plugin for Oracle Jdeveloper
  - User-extensible Vulnerability Descriptions and Recommendations
- 2.6.5
  - SCA for .NET 4.0
  - IDE Plugin support for Visual Studio 2010
  - SCA, IDE Plugins and Demo Suite for Windows 7
  - SCA, 360 Server and RTA for Windows 2008 Server R2

SAP ABAP Scanning

- SAP is used by many companies to “run” the company
  - Finance, Manufacturing, Marketing, HR, etc
- ABAP is SAP’s business processing language to customize SAP
- Fortify SAP ABAP scanning will analyze ABAP applications for vulnerabilities