

IBM Rational Application Security

Extending Application Security Testing across the SDLC

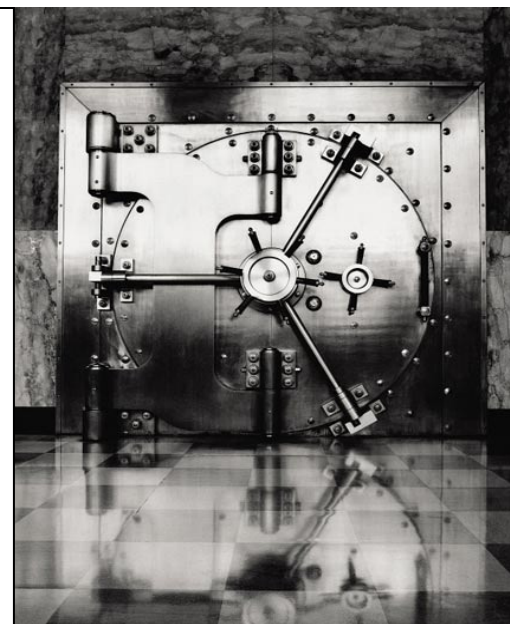
Hong Kong – Jan 20, 2011



Executive Summary

- Web applications are the **greatest source of risk** for organizations
- Rational Application Security enables organizations to **address root cause** of this risk
- AppScan leverages a mix of technologies (**static & dynamic**) to enable the **right use cases**
- AppScan is a key part of IBM Security's **full solution view of application security**

Rational AppScan Suite
enables
***Comprehensive Application
Vulnerability Management***



Rate of Security Breaches Steadily Increasing



Facebook to Encrypt UIDs After App Security Breach

By: *Chloe Albanesius*

10.21.2010



Data breach costs rise as firms brace for next loss

By Robert Westervelt, News Editor 02 Feb 2009 | SearchSecurity.com

According to Ponemon Institute, the total average costs of a data breach grew to \$202 per record compromised, an increase of 2.3% since 2007 (\$197 per record)



McAfee highlights perils of offshoring sensitive data

Phil Muncaster | Jan 30, 2009 9:59 AM

Global companies may have lost over US\$1 trillion worth of intellectual property last year owing to data theft, according to new research from McAfee presented today at the World Economic Forum.



January 27, 2009

Hackers steal details of 4.5 million in attack on Monster jobs site

The Costs from Security Breaches are Staggering

**285 MILLION RECORDS
COMPROMISED IN 2008**

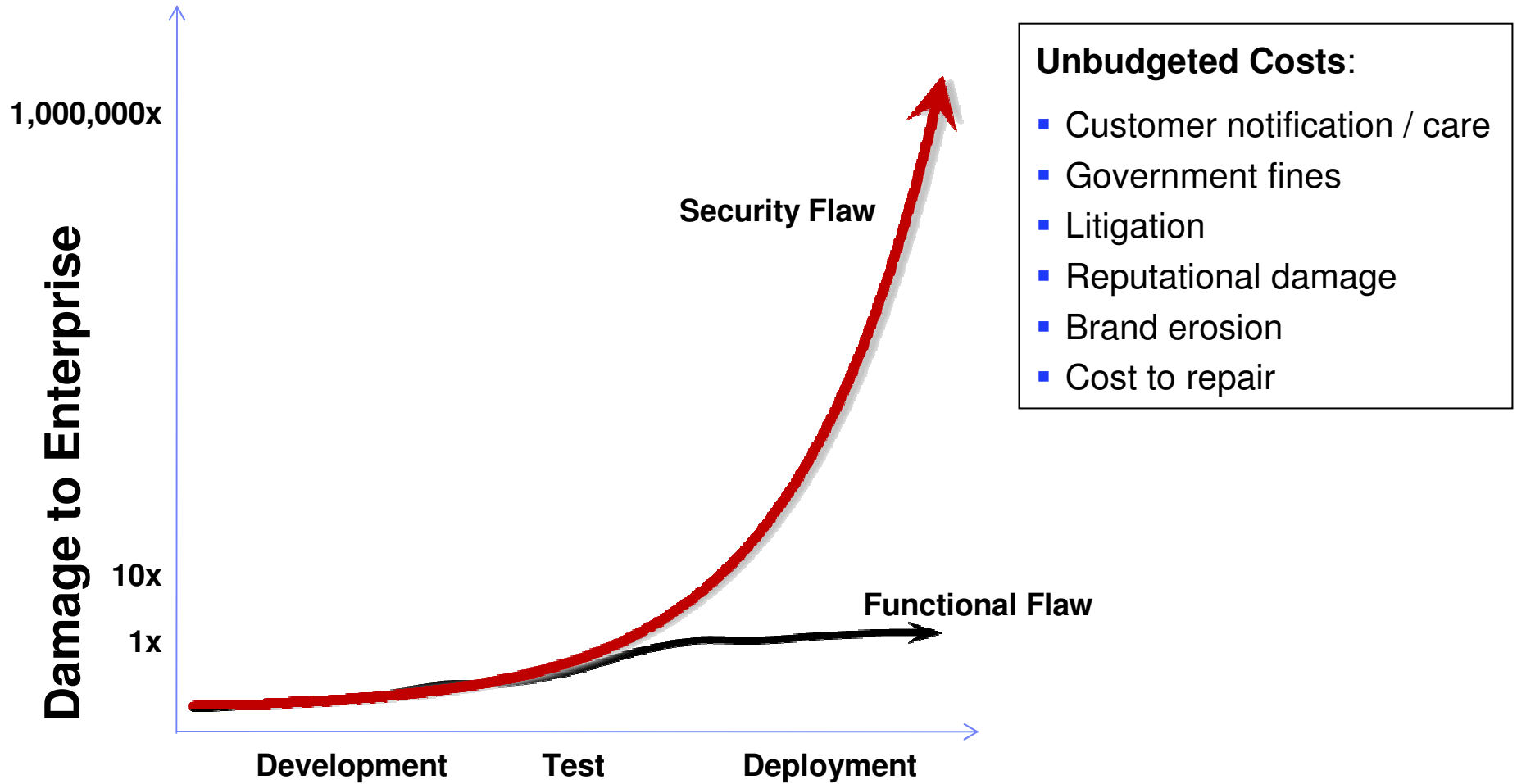
Verizon 2009 data Breach
Investigations Report

**\$204 COST PER
COMPROMISED
RECORD**

Ponemon 2009-2010 Cost
of a data Breach Report

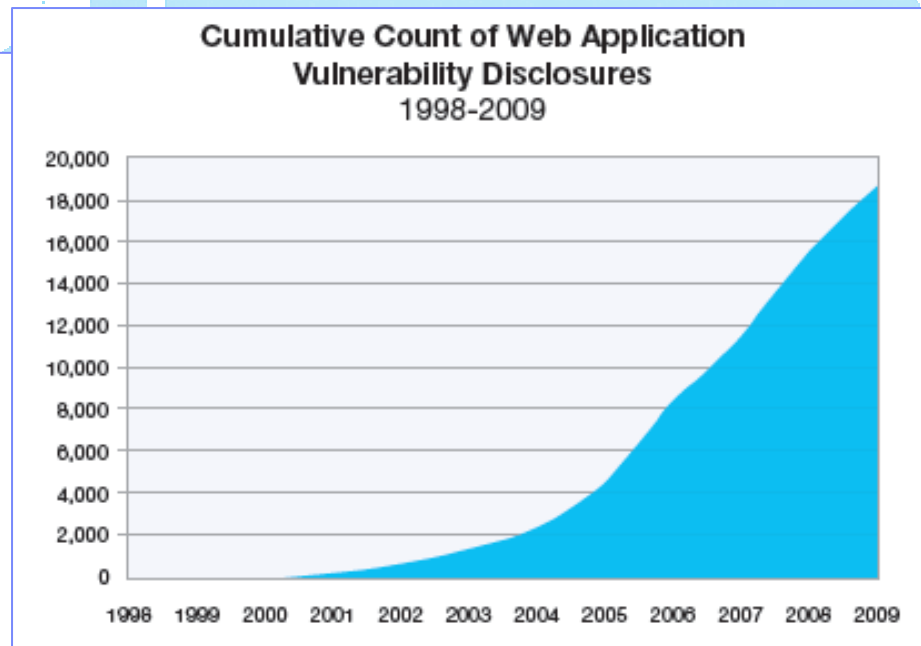
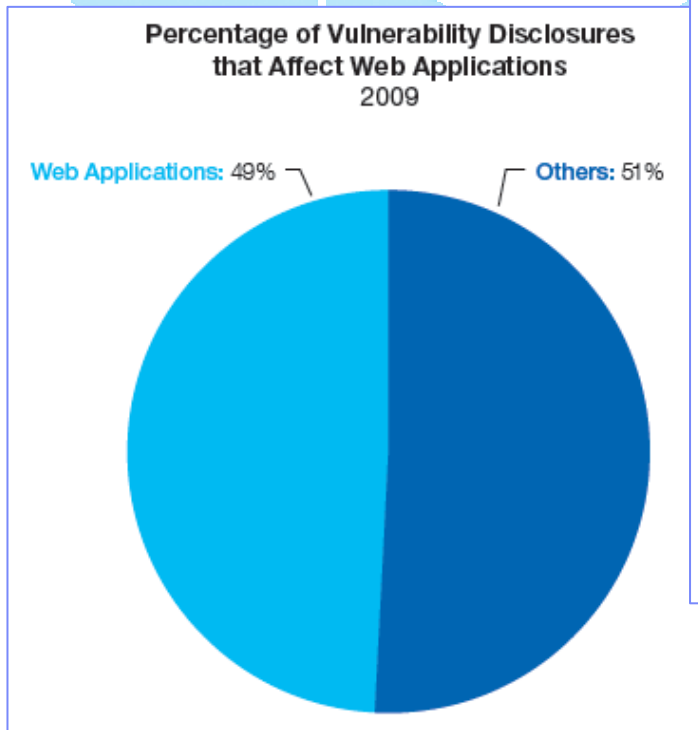
**TRANSLATES TO \$58.1B
COST TO CORPORATIONS**

Sources of Security Breach Costs



Web Applications are the greatest risk to organizations

- Web application vulnerabilities represented **the largest category** in vulnerability disclosures
- In 2009, 49% of all vulnerabilities were Web application vulnerabilities
- SQL injection and Cross-Site Scripting are neck and neck in a race for the top spot



IBM Internet Security Systems 2009 X-Force®
Year End Trend & Risk Report

Why are Web Applications so Vulnerable?

- Developers are mandated to deliver functionality on-time and on-budget - but not to develop secure applications
- Developers are not generally educated in secure code practices
- Product innovation is driving development of increasingly complicated software for a Smarter Planet
- Network scanners won't find application vulnerabilities and firewalls/IPS don't block application attacks



Volumes of applications continue to be deployed that are riddled with security flaws...

...and are non compliant with industry regulations

The Solution - Security for Smarter Products

- Smarter Products require secure applications
- Security needs to be built into the development process and addressed throughout the development lifecycle
- Providing security for smarter products requires comprehensive security solutions deployed in concert with application lifecycle management offerings that:
 - Provide **integrated testing** solutions for developers, QA, Security and Compliance stakeholders
 - Leverage **multiple appropriate testing technologies**
 - Provide **effortless security** that allows development to be part of the solution
 - Support **governance, reporting and dashboards**
 - Can **facilitate collaboration** between development and security teams

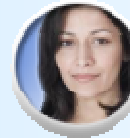


Pre-empt Costly Risk Mitigation

– *Become Secure, by Design*

Customer Speak!

Reduce today's most significant area of risk by adopting a cost effective and thorough application security program



"We have more of our business supported by our web presence – we can't afford the business risk of deploying unsecured applications and services"

Embed security early into the software delivery process to enable on time and on budget delivery of secure applications



"Embracing security in development allowed us to get ahead of schedule disruption and increased costs from security acceptance testing"

Establish a security blueprint to create and maintain security governance, manage risk and ensure compliance



"IBM has recognized this trend and has created comprehensive security packages that leverage various products to provide for multiple layers of security to customers"

Make Applications Secure, by Design

Cycle of secure application development

Design Phase

- Consideration is given to security requirements of the application
- Issues such as required controls and best practices are documented on par with functional requirements

Development Phase

- Software is checked during coding for:
 - Implementation error vulnerabilities
 - Compliance with security requirements

Build & Test Phase

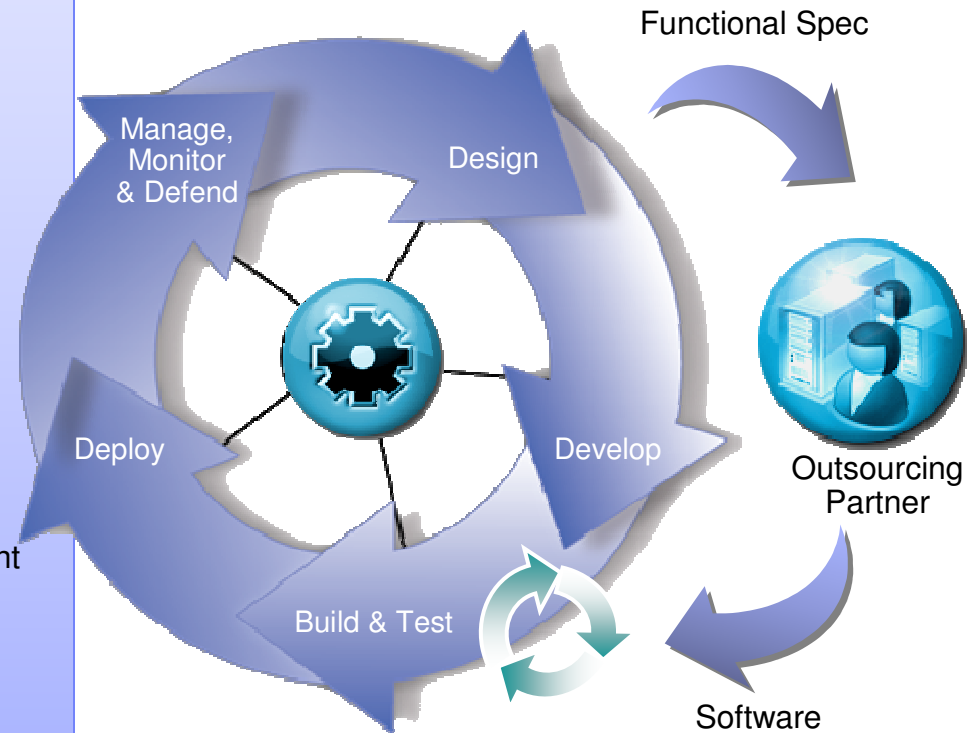
- Testing begins for errors and compliance with security requirements across the entire application
- Applications are also tested for exploitability in deployment scenario

Deployment Phase

- Configure infrastructure for application policies
- Deploy applications into production

Operational Phase

- Continuously monitor applications for appropriate application usage, vulnerabilities and defend against attacks



Security is an Enterprise Responsibility

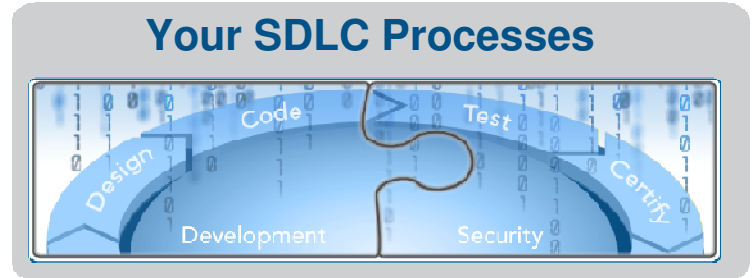


Securing Applications is a Challenge

Your Application Portfolio
Different types, different sources

Financial	HR	Logistics	Corp Internet
Outsourced	In-house	Legacy	Open Sourced

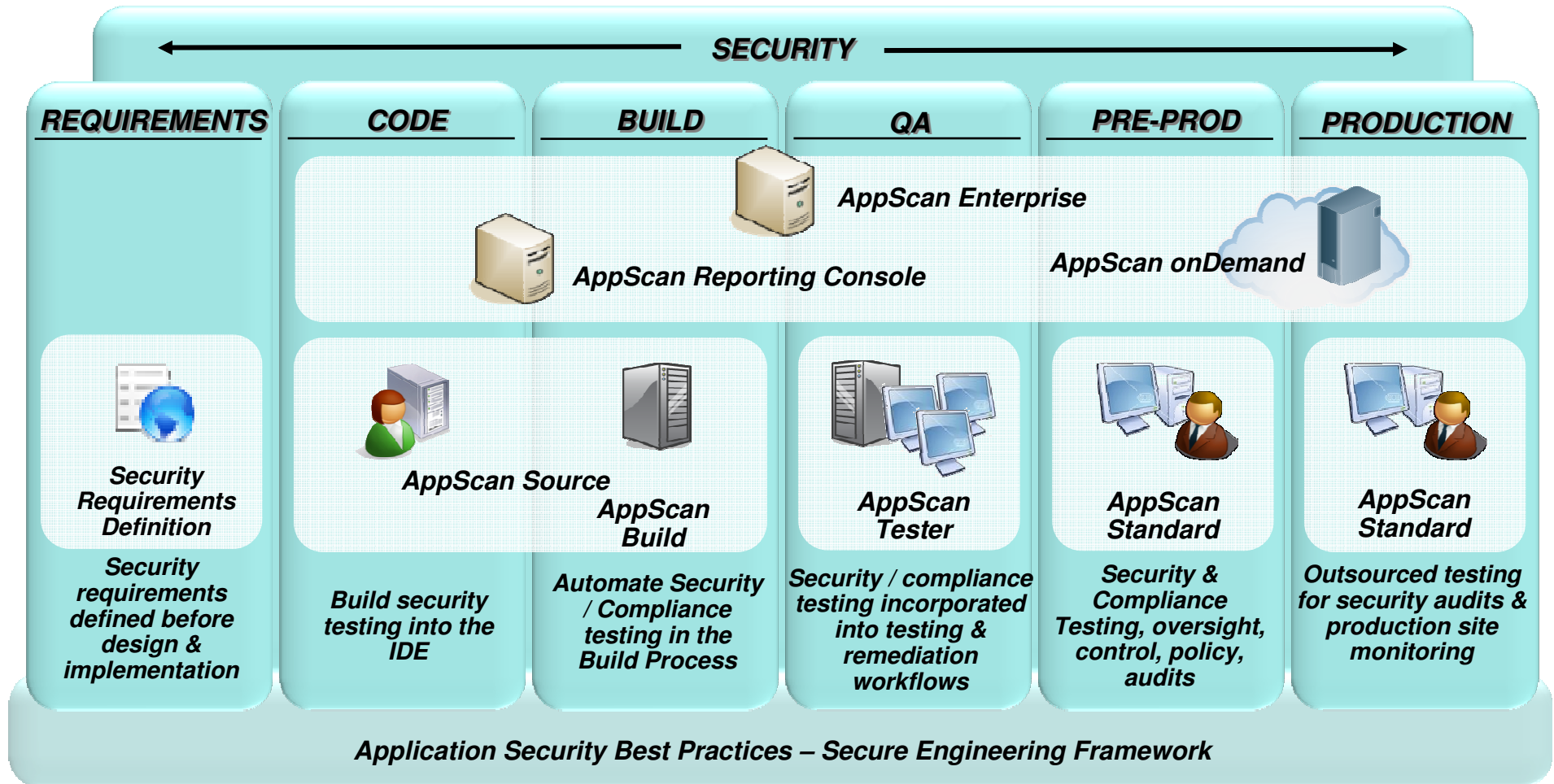
Your Policies
Data Privacy
Regulatory Compliance
Outsourcer Accountability



- Large and diverse application portfolios
- In-house and outsource development
- External & internal regulatory pressure
- Pockets of security expertise
- Yet another task for developers

Need an efficient, scalable, automated way to develop and deliver secure applications...

IBM Rational AppScan Suite – Comprehensive Application Vulnerability Management



What is AppScan Source Edition?

- A static code analysis security testing solution with centralized control of security policies
- Allows organizations to create, distribute and enforce consistent security policies
- Provides automated security testing by seamlessly integrating security source code analysis into the build process



Benefits:

- Enables security teams strengthen application security, protect confidential data and improve compliance
- Enables the cost effective remediation of vulnerabilities early in the development process to support on-time delivery of projects

IBM Rational AppScan Source Edition Solution

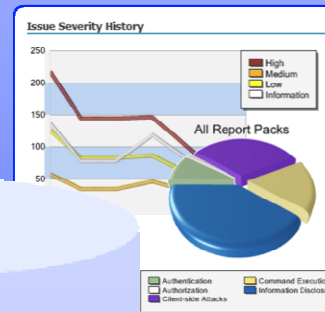
Security

- Configure Software
- Scan
- Triage Results
- Manage Security Policies

Reset	Vulnerability	Exceptions		Totals
		Type I	Type II	
High	198	310	16	524
Medium	198	99	8	305
Low	682	14		
Totals	1078	51		

Reporting Console

- Track Progress
- Compare Applications
- Customize Dashboards
- Manage Portfolio Risk
- Combine BB/WB results

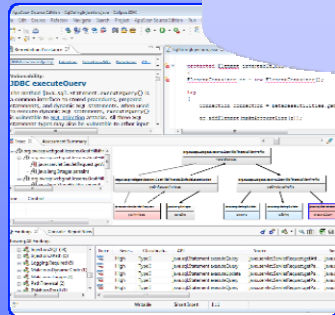


Core

- Knowledgebase
- Assessment Database
- Custom Rules

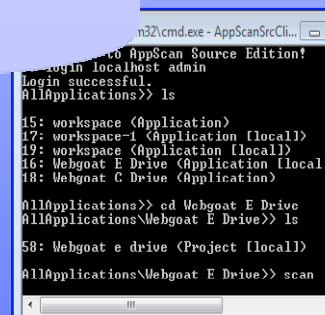
IDE Plug-Ins

- Investigate Flaws
- Remediate with Guidance
- Scan
- Confirm Fix



Automation

- Build integration
- Automate Scans
- ANT, Make, Maven integration
- Data Access API



Security Testing Technologies...

Combination Drives Greater Solution Accuracy

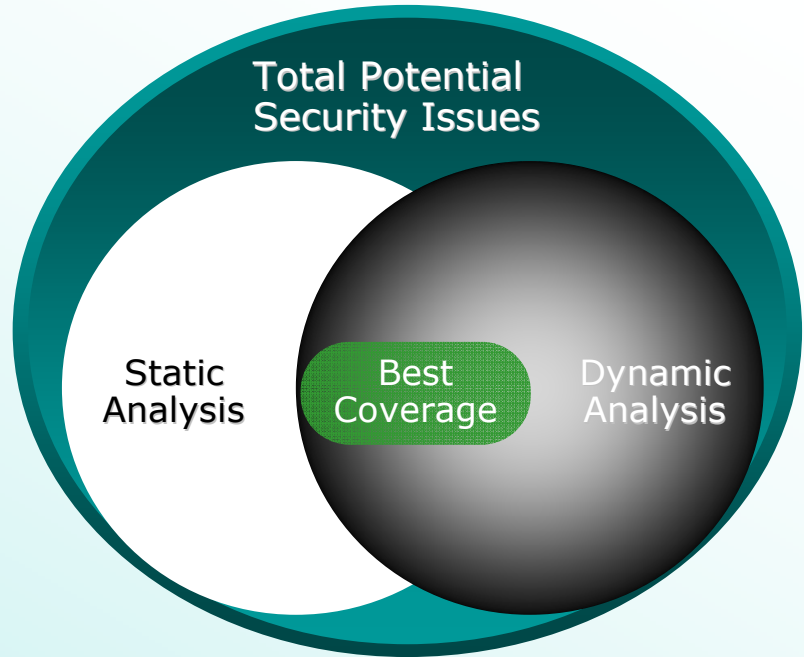
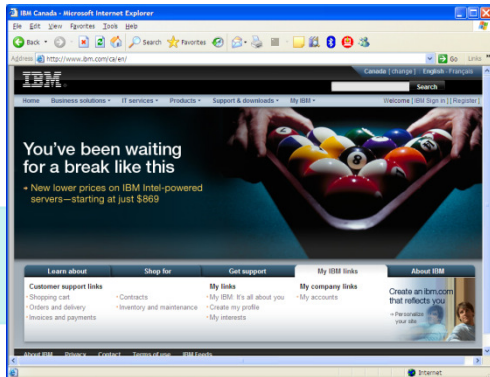
Static Code Analysis (Whitebox)

- Scanning source code for security issues

```
184 }  
185 }  
186 }  
187 }  
188 }  
189 }  
190 }  
191 }  
192 }  
193 }  
194 }  
195 }  
196 }  
197 }  
198 }  
199 }  
200 }  
201 }  
202 }  
203 }  
204 }  
205 }  
206 }  
207 }  
208 }  
209 }  
210 }  
211 }  
212 }  
213 }  
214 }  
215 }  
216 }  
217 }  
218 }  
219 }  
220 }  
221 }  
222 }  
223 }  
224 }  
225 }  
226 }  
227 }  
228 }  
229 }  
230 }  
231 }  
232 }  
233 }  
234 }  
235 }  
236 }  
237 }  
238 }  
239 }  
240 }  
241 }  
242 }  
243 }  
244 }  
245 }  
246 }  
247 }  
248 }  
249 }  
250 }  
251 }  
252 }  
253 }  
254 }  
255 }  
256 }  
257 }  
258 }  
259 }  
260 }  
261 }  
262 }  
263 }  
264 }  
265 }  
266 }  
267 }  
268 }  
269 }  
270 }  
271 }  
272 }  
273 }  
274 }  
275 }  
276 }  
277 }  
278 }  
279 }  
280 }  
281 }  
282 }  
283 }  
284 }  
285 }  
286 }  
287 }  
288 }  
289 }  
290 }  
291 }  
292 }  
293 }  
294 }  
295 }  
296 }  
297 }  
298 }  
299 }  
300 }  
301 }  
302 }  
303 }  
304 }  
305 }  
306 }  
307 }  
308 }  
309 }  
310 }  
311 }  
312 }  
313 }  
314 }  
315 }  
316 }  
317 }  
318 }  
319 }  
320 }  
321 }  
322 }  
323 }  
324 }  
325 }  
326 }  
327 }  
328 }  
329 }  
330 }  
331 }  
332 }  
333 }  
334 }  
335 }  
336 }  
337 }  
338 }  
339 }  
340 }  
341 }  
342 }  
343 }  
344 }  
345 }  
346 }  
347 }  
348 }  
349 }  
350 }  
351 }  
352 }  
353 }  
354 }  
355 }  
356 }  
357 }  
358 }  
359 }  
360 }  
361 }  
362 }  
363 }  
364 }  
365 }  
366 }  
367 }  
368 }  
369 }  
370 }  
371 }  
372 }  
373 }  
374 }  
375 }  
376 }  
377 }  
378 }  
379 }  
380 }  
381 }  
382 }  
383 }  
384 }  
385 }  
386 }  
387 }  
388 }  
389 }  
390 }  
391 }  
392 }  
393 }  
394 }  
395 }  
396 }  
397 }  
398 }  
399 }  
400 }
```

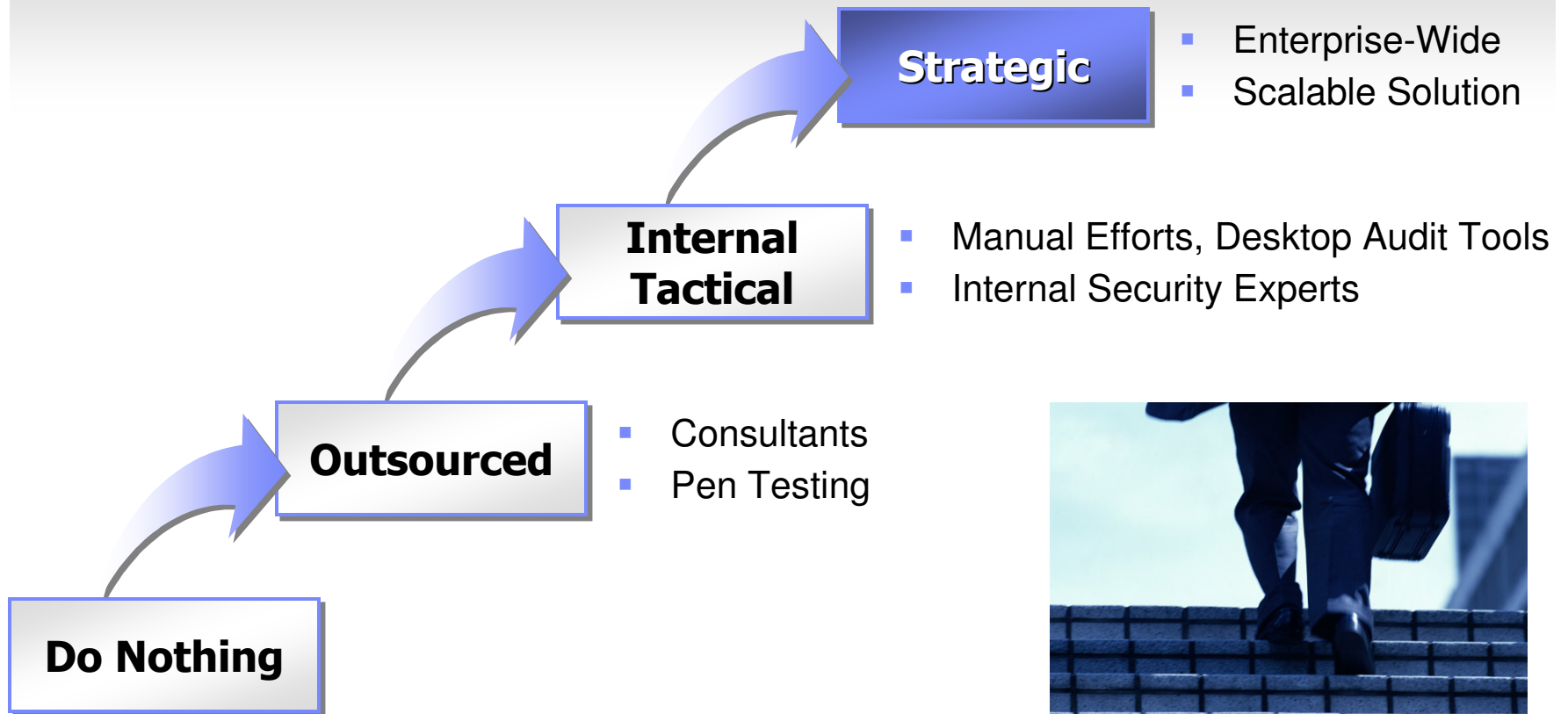
Dynamic Analysis (Blackbox)

- Performing security analysis of a compiled application

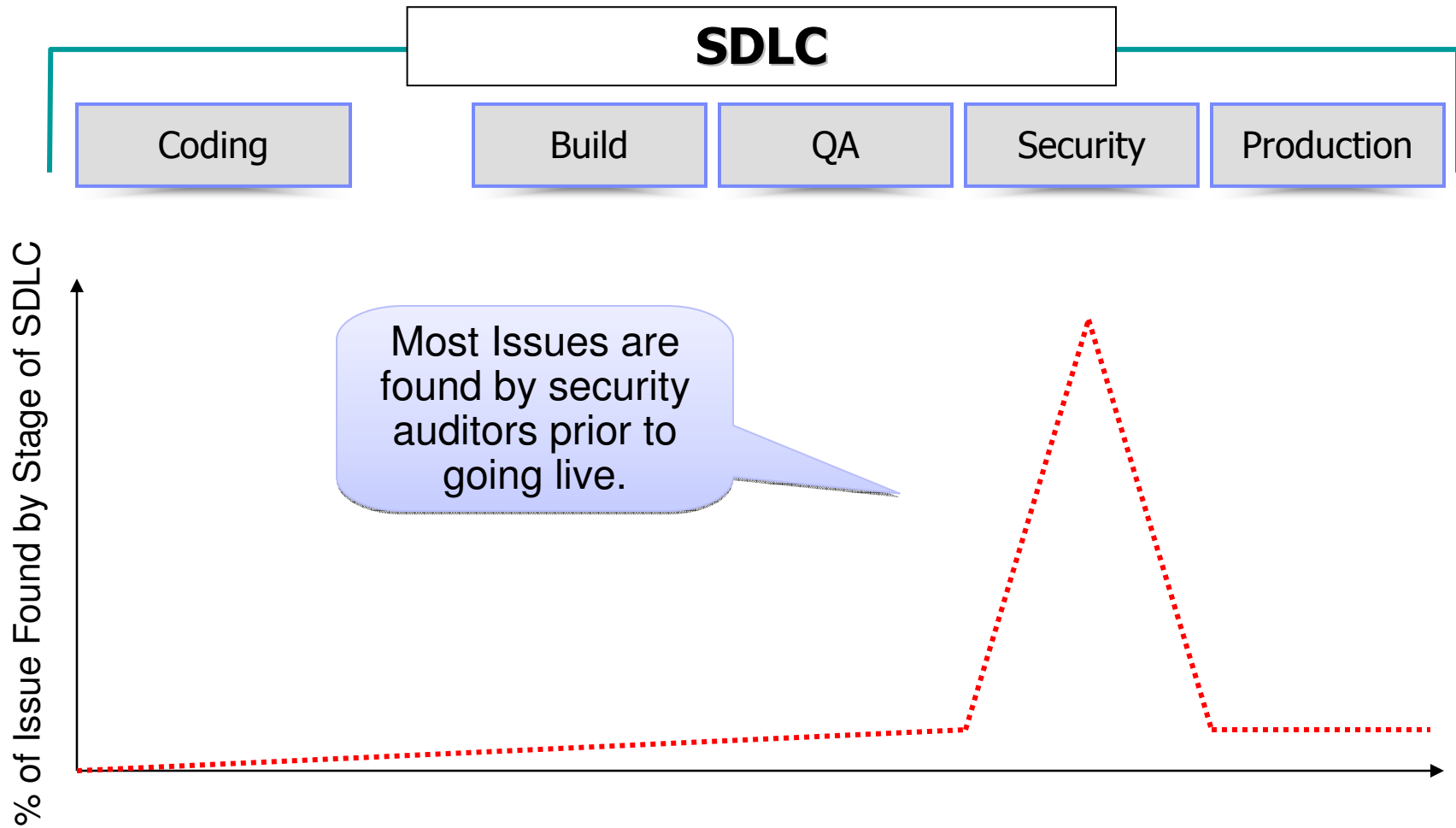


Addressing Web Application Security

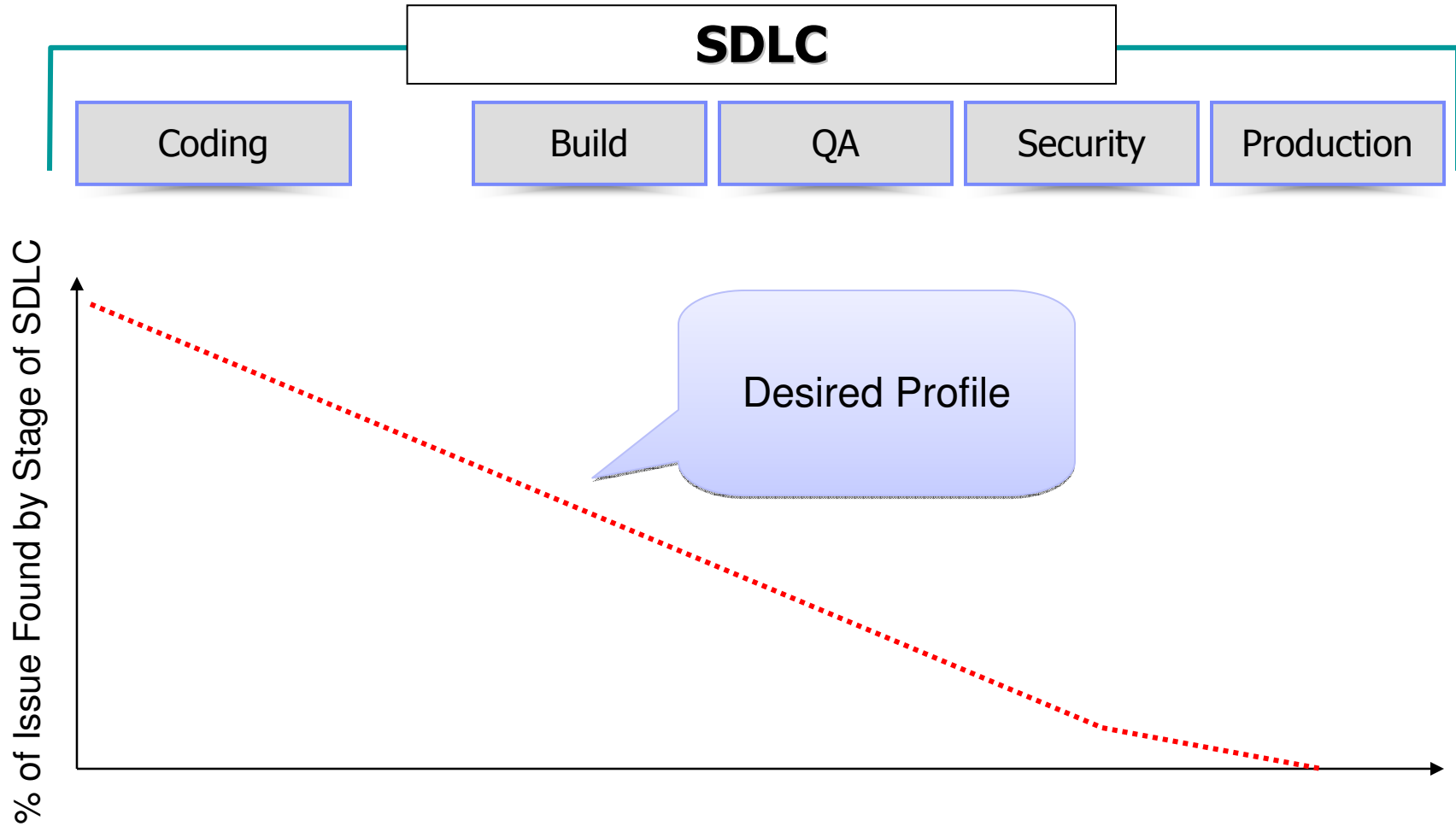
Approaches for addressing Web Application Security



Security Testing Within the Software Lifecycle

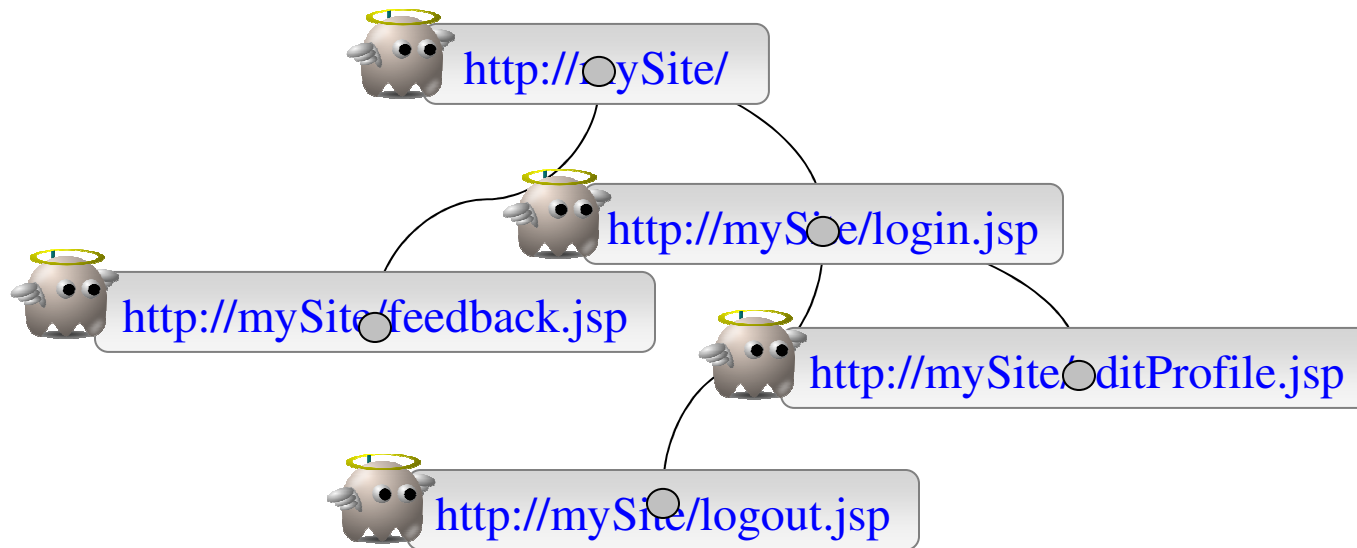


Security Testing Within the Software Lifecycle



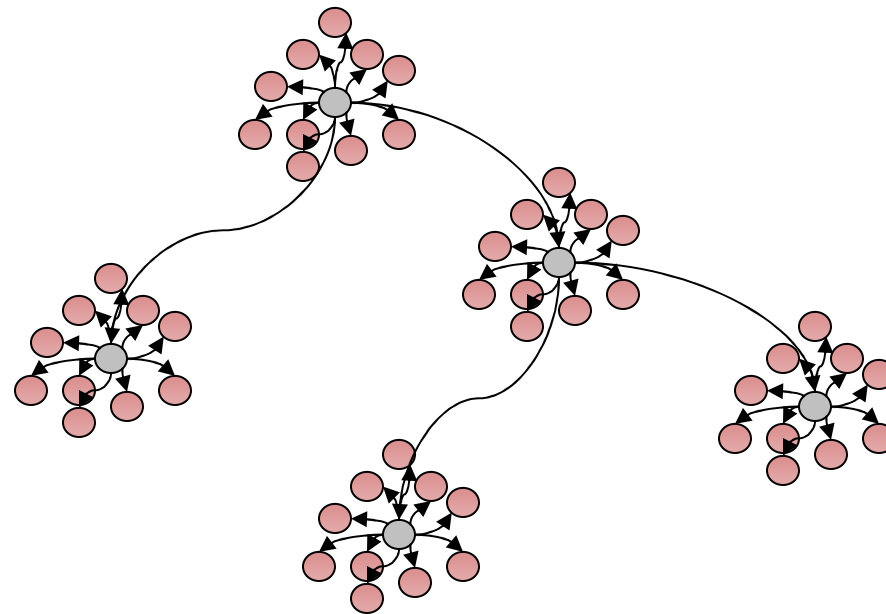
How BB Scanners Work

- Stage 1: Crawling as an honest user



How BB Scanners Work

- Stage 1: Crawling as an honest user
- Stage 2: Testing by tampering requests



White-box (Discovering SQL Injection)

```
// ...
String username = request.getParameter("username");
String password = request.getParameter("password");

// ...
String query = "SELECT * from tUsers where " +
    "userid='" + username + "'" +
    "AND password='" + password +

// ...
ResultSet rs = stmt.executeQuery(query);
```

Source – a method returning tainted string

User can change executed SQL commands

Sink - a potentially dangerous method

White-box (Discovering SQL Injection)



```
String username = request.getParameter("username");
```

```
// ...  
String username = request.getParameter("username");  
String password = request.getParameter("password");
```

```
// ...  
String query = "SELECT * from tUsers where " +  
    "userid='" + username + "'";
```

```
String query = "SELECT ..." + username
```

```
// ...  
ResultSet rs = stmt.executeQuery(query);
```

```
ResultSet rs = stmt.executeQuery(query);
```

A Common Fix (not always the best for SQL Injection)

```
// ...
String username = request.getParameter("username");
String password = request.getParameter("password");

// ...
String query = "SELECT * from tUsers where " +
    "userid='" + Encode(username) + "' " +
    "AND password='" + Encode(password) + "'";

// ...
ResultSet rs = stmt.executeQuery(query);
```



Sanitizer:
a method returning
a non-tainted string

Complementary Security Assessment



Static

- Findings directly tied to their locations in the source
- Test earlier in lifecycle
- Test sub-components of an application
- Easier automation
- Fast scanning
- Non-web-applications, infrastructure, middleware
- All control flows
- Illuminate architecture and logic
- Consistent Automation

Dynamic

- Simpler configuration
 - No cross-domain requirement
- Lower learning curve
- Findings include attack vectors
- Captures dynamic activity (Spring, Struts, CAB)
- Scan unsupported source languages
- 3rd party applications (no source)
- Find configuration vulnerabilities
- Smaller finding sets

Differences Between SAST and DAST Approaches

	 Static Analysis	 Dynamic Analysis
Scan input	Scans source code and bytecode for security and quality issues. Requires access to source or bytecode	Scans running web applications. Requires starting point URL, and login credentials where relevant
Assessment techniques	Uses “taint analysis” and pattern matching techniques to locate issues	Tampering of HTTP messages to locate application and infrastructure layer issues
Where does it fit in application development lifecycle	Early – fits best during application development and build automation	Later – fits best in QA and security verification of production applications
Results & Output	Results are presented by line of code, source to sink functions flow	Results are presented as HTTP messages (exploit requests)

Why IBM?

- **IBM continues to demonstrate leadership in security**
 - IBM Wins ‘Best Security Company’
 - IBM recognized as International Association of Privacy Professionals “[Top Privacy Innovators](#)” in 2009
- **Rational is #1 in Application Security Testing Market Share**
 - According to Gartner and IDC
- **Complete security from IBM Security Solutions**
 - 5 Security Pillars
 - Enterprise-wide coverage of application security from design through development and into production
 - [Secure Engineering Framework](#) – security practices employed by IBM and for customers (Redbook)
- **Rational AppScan breadth of technologies and offerings**
 - Solutions for all SDLC stakeholder use cases
 - Leverages best-of-bread static and dynamic analysis
 - Over 60 application vulnerability management innovations patented or publically disclosed
- **Commitment to customer success – R&D Investment**
 - More than 100 resources, 6 labs, plus extended R&D teams

**Excellence Award:
2010 Best Security Company**

2007 Best Security Company – Watchfire
2006 Best Security Company - ISS

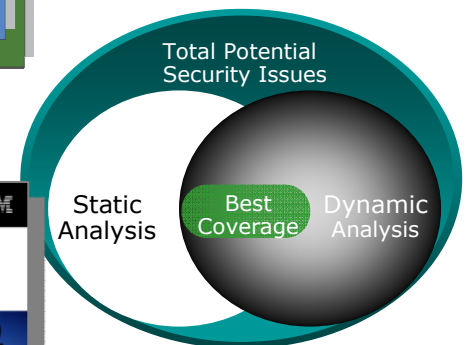


Security in Development: The IBM Secure Engineering Framework

Redbook
For DevOps
Security Affairs
The IBM Group
an IBM Business
and Service

- Investigating common development processes and the IBM Integrated Product Development process
- Embedding security awareness and requirements in the software development process
- Developing and automating assessments

Redbooks



IBM Investment & Commitment to Customer Success



Acquisitions:

- Watchfire acquisition 2007
- Ounce acquisition 2009

Global R&D Team

- Hawthorn NY research lab
- Tokyo research lab
- Israel research lab
- Ottawa development lab
- Toronto development lab
- Boston development lab

Product Team:

- 90 people in Rational development
- 13 people in IBM Research

Extended Team:

(enabling us to tackle broader security requirements)

- ISS team, including X-Force research
- Tivoli team
- Datapower team
- Optim team
- GBS team
- Guardium team

Technology Innovation in Application Vulnerability Management

- Over 60 innovative concepts patented or publically disclosed
- Patents covering:
 - Static Analysis, Blackbox Analysis, Automated Hybrid Analysis as well as general security-related innovations
- Key patents in the area of
 - Basic static analysis
 - Basic blackbox security scanning
 - Basic hybrid analysis
 - Automatic correction of security vulnerabilities
 - Elimination of false positives
 - Application state tracking
 - Web 2.0 application scanning

*Automated Hybrid
Analysis introduced
in Fall 2008*

AppScan Source Edition Patents

- Patent 7240332: Method and System for Detecting Vulnerabilities in Source Code
- Patent 7398516: Method and System for Detecting Race Condition Vulnerabilities in Source Code
- Patent 7398517: Method and System for Detecting Vulnerabilities in Source Code (Continuance)
- Patent 7418734: Method and System for Detecting Privilege Escalation Vulnerabilities in Source Code

Thank
YOU