Secure Web Applications

A Black AND White Approach

Presented to OWASP, Ottawa
July 15, 2008

John Linehan
Senior Security Consultant
John.linehan@armorize.com

armorize technologies
Secure Your Web Code

Who am I?

- **John Linehan**
- **Armorize Technologies**
  - Web Application Security
  - Senior Security Consultant
  - Santa Clara HQ
  - R&D center in Taipei
- **Eight years in Ottawa consulting market**
  - Systems and Network Security
  - Risk Management
  - Elytra Enterprises
  - DFAIT
  - OCIPEP
  - Private sector clients

Discussion

- Web Application Security
- ~~Black Box Vs White Box~~
- ~~Manual effort Vs Automation~~
- Black Box AND White Box
- Manual effort AND Automation
- Source Code Analysis
- Penetration Testing

Scenarios

- **Told a web application was vulnerable**
  - And said “now what?”
- **Found a vulnerable web application**
  - And heard “now what?”
- **Paid a security consultant**
  - And did nothing with report
- **Bought a security appliance**
  - And did nothing with it
- **Want Secure Web Applications**

Web Application Security

Web Application

- Software applications
- Interact with users or other applications
- HTTP or HTTPS

Programming Languages

- JAVA, PHP, .NET, etc.

Other Concerns

- Web 2.0, SOA, AJAX, Frameworks, etc.

Web Application Vulnerabilities

- Weakness in custom Web Application, architecture, design, configuration or Code

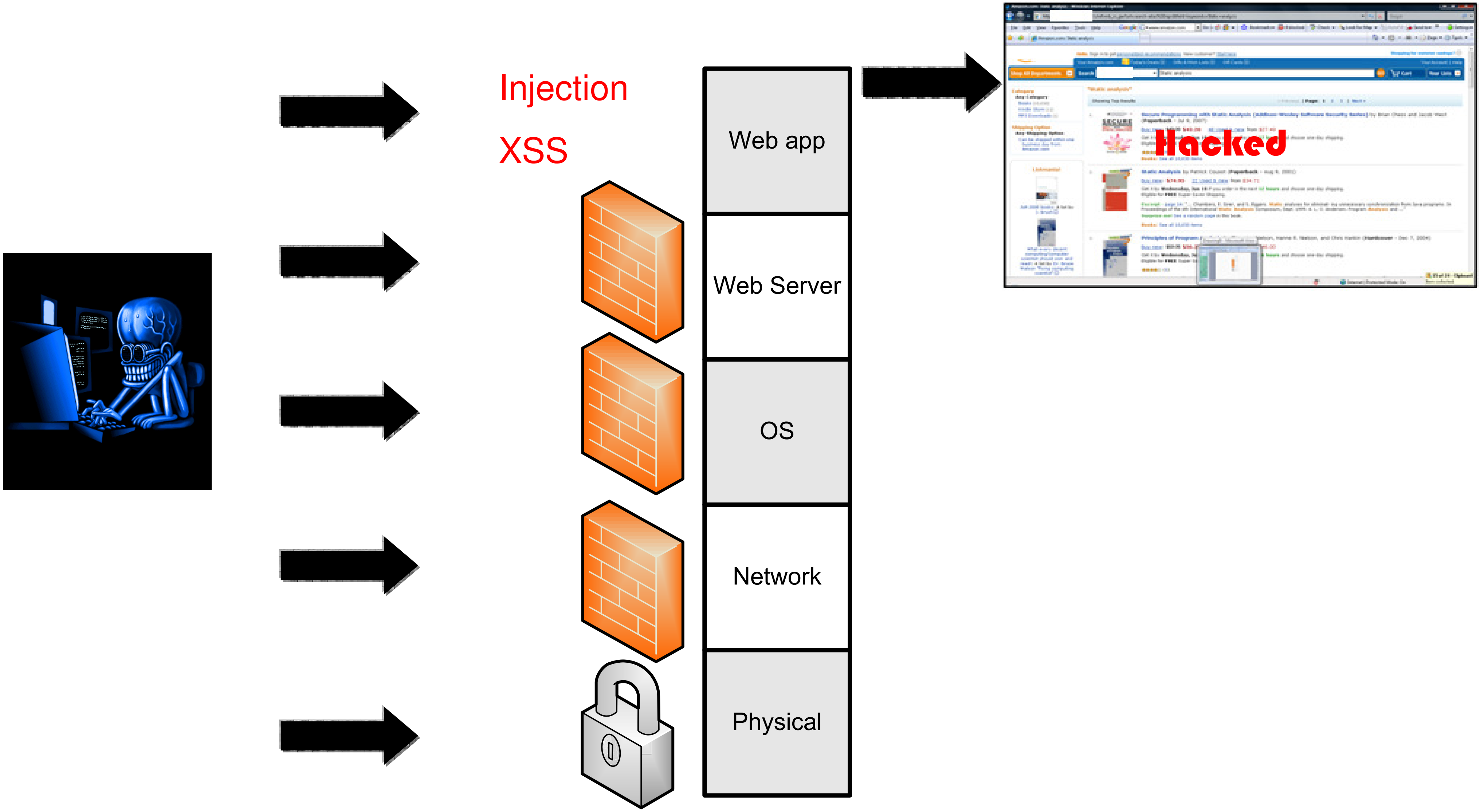
Web Application Security

- Focus higher in the stack
- Not Network, OS or Physical

Paradigm Shifts

- 1 - The Changing Face of Attacks
- 2 - The Changing Behavior of Attackers
- 3- Increasing Institutional Pressure

1 - The Changing Face of Attacks



Injection

XSS

Web app

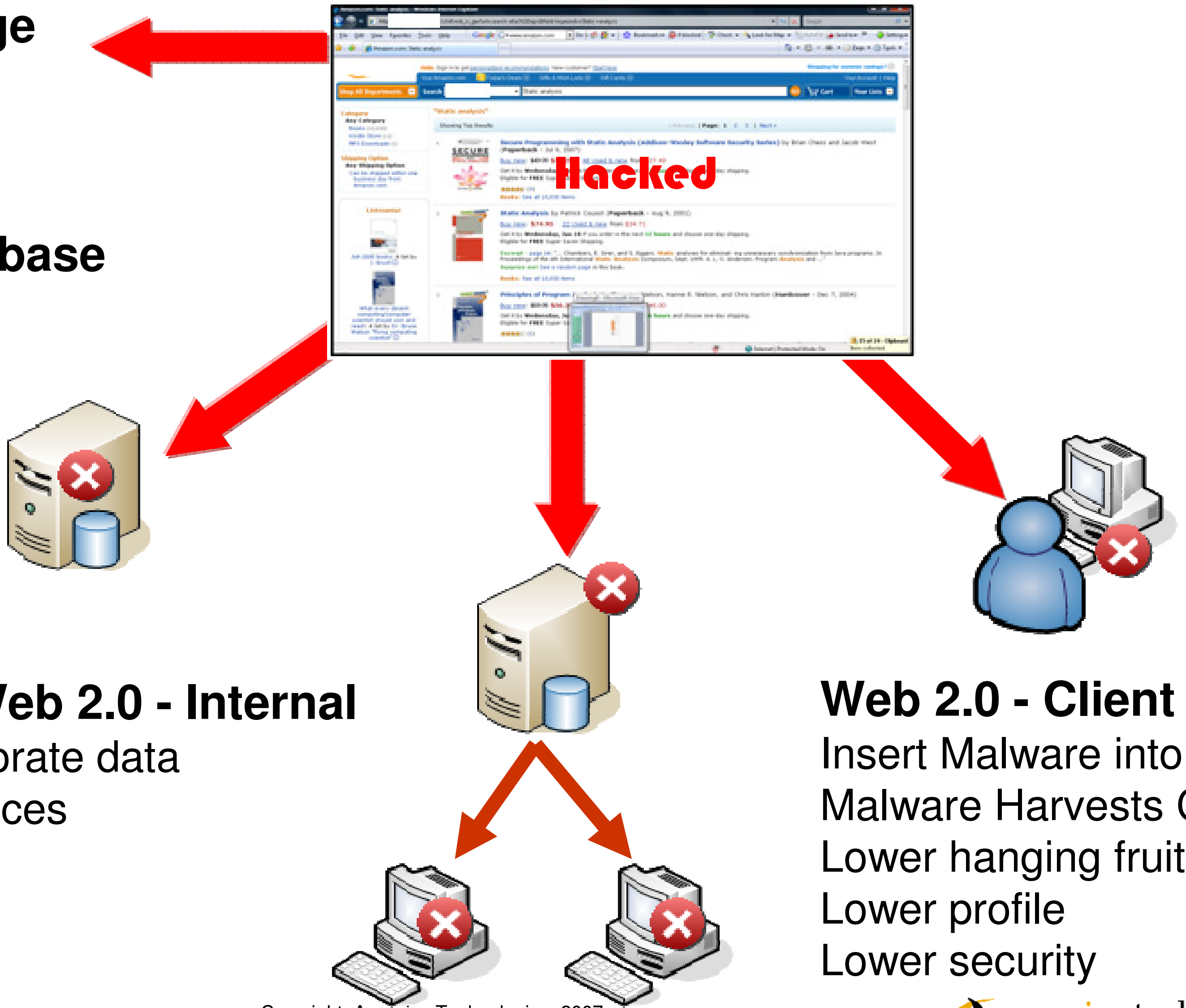
Web Server

OS

Network

Physical

2 - The behavior of Web Attacks



Web 1.0 – Web Page

Defacement

Pre-Web 2.0 - Database

Credit Cards
Health
Privacy

Pre-Web 2.0 - Internal

Corporate data
Finances

Web 2.0 - Client

Insert Malware into Websites
Malware Harvests Client Data
Lower hanging fruit
Lower profile
Lower security

armorize technologies
Secure Your Web Code

2 - The Changing behavior of Web Attacks

Edison Chen - Hong Kong Movie Star

- Targeted fans accessing legitimate sites and new sites with malware
- Attacked Protected Storage

China Mass SQL Injection

- Google Hacking / SQL Injection
- Targets Asian IE plug-ins

Bank of India and Russian Business Network

- Malicious code in iFrame
- Bank Customers redirected to sites
- Password stealing
- Zombies

Danmec/Asprox

- Google Hacking / SQL Injection
- Client downloads malicious JavaScript from direct84.com

2 - The behavior of Web Attacks

- **Public Safety Canada Canadian Cyber Incident Response Centre (CCIRC)**
- **IN08-002 (23 June 2008)**
  - Purpose:
    - Ensure web presence is not impacted by SQL injection attacks.
    - Unwittingly infect .. users visiting their site
    - Scripts inserted in the web pages html code.
  - Background:
    - Attacks plaguing the internet
    - Compromised sites unwittingly redirect client browsers
    - Malicious external domains .... compromise the visitor’s system.
  - Impact
    - Visitors to compromised web sites will be infected if not adequately protected.
- **TR08-001 (11 June 2008)**
  - Alleviating the Threat of Mass SQL Injection Attacks (PDF)
    - Talks about Security across stack
    - Secure coding and Penetration Testing
    - Refers to MSDN and OWASP

Hacking Motives

Pure Interest

- Antisocial geek in mother's basement is the least of our worries

Underground Economy

- Identity Theft, Phishing, Credit Card Information, Banking details
- Russian Business Network - The baddest of the bad (Verisign June 2006).
- Bank of India hack - injected malicious iFrame

Military Backed Operations

- *China seeks Taiwan spy for computer hacking*
  - International Herald tribune October 2007
- China Accuses Taiwan of owning thousands of their servers
  - China Times, October 2007
- *Estonia hit by Moscow cyber war*
  - BBC.co.uk May 2007
- *China's cyber army is preparing to march on America, says Pentagon*
  - timesonline.co.uk (Sept 2007)
- *Anti-Israel hackers deface central bank site*
  - register.co.uk April 2008
- *USAF Considers Creation of Military Botnet*
  - Slashdot May 12 2008

Military Backed Operations

- **Col. Charles W. Williamson III**
  - “The world has abandoned a fortress mentality in the real world, and we need to move beyond it in cyberspace.
  - “America needs a network that can project power by building a [botnet] that can direct such massive amounts of traffic to target computers that they can no longer communicate and become no more useful to our adversaries than hunks of metal and plastic.
  - “America needs the ability to carpet bomb in cyberspace to create the deterrent we lack.
  - “The time for fortresses on the Internet also has passed, even though America has not recognized it.
  - “Now, the only consequence for an adversary who intrudes into or attacks our networks is to get kicked out — if we can find him and if he has not installed a hidden back door.
  - “That is not enough. America must have a powerful, flexible deterrent that can reach far outside our fortresses and strike the enemy while he is still on the move”

Armed Forces Journal – Carpet bombing in cyberspace
http://www.armedforcesjournal.com/2008/05/3375884

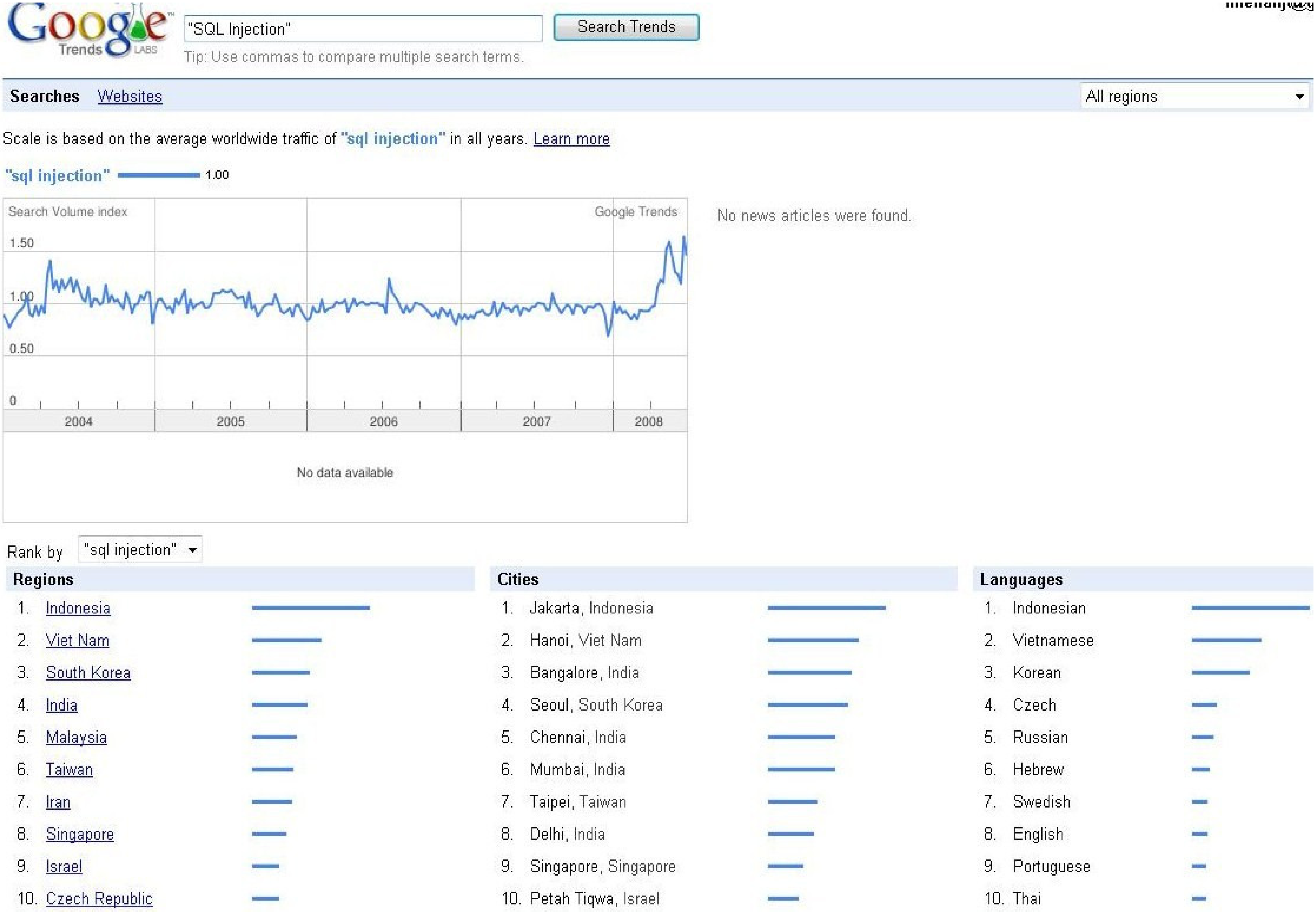
Taiwan Malware Report

- **135,000+ URLs**
- **582 pages with links to malicious code**
- **221 pages that actively push malicious code to browser**
- **72 different spyware types**
- **Source broken down by country**
  - Over 70% from one source

Jeremiah Grossman on Taiwan Cyber Issues

- “Taiwan cyber crime environment is MUCH different and WAY more serious than anything I’ve ever been exposed to in the U.S or elsewhere.
- “Experience thus far has everything to do with criminals attempting to monetize. In Taiwan it’s an environment of true military supported cyber warfare as a result of an intense political climate with China.
- “Both sides are extremely well organized, funded, motivated, their actions unrestricted.
- “Daily computing life filled with 0-days, single person target rootkits, trojan horses, malware-laced spam, and attacks designed not to monetize or embarrass but for militaristic espionage with command and control goals.
- “They view their exploit code more like weapons and munitions than anything else.
- “The private and government sectors are in close, open, and bi-directional communication. This might have something to do with their mandatory military service so relationships between the two are more natural”

http://jeremiahgrossman.blogspot.com/2007_09_01_archive.html



"SQL Injection" Search Trends

Tip: Use commas to compare multiple search terms.

Scale is based on the average worldwide traffic of "sql injection" in all years. Learn more

"sql injection" 1.00

No data available

No news articles were found.

Rank by "sql injection"

Regions	
1.	Indonesia
2.	Viet Nam
3.	South Korea
4.	India
5.	Malaysia
6.	Taiwan
7.	Iran
8.	Singapore
9.	Israel
10.	Czech Republic

Cities	
1.	Jakarta, Indonesia
2.	Hanoi, Viet Nam
3.	Bangalore, India
4.	Seoul, South Korea
5.	Chennai, India
6.	Mumbai, India
7.	Taipei, Taiwan
8.	Delhi, India
9.	Singapore, Singapore
10.	Petah Tiqwa, Israel

Languages	
1.	Indonesian
2.	Vietnamese
3.	Korean
4.	Czech
5.	Russian
6.	Hebrew
7.	Swedish
8.	English
9.	Portuguese
10.	Thai

3 – Increasing Institutional Pressure

Fear of non-compliance

- Compliance driven market
- Everyone has a silver bullet

Security

- People
- Processes
- Technology

Compliance

- People
- Processes
- Technology
- All doing least required amount of work

Don’t be driven by compliance

- Should fear lack of Security

Compliance

- **MITS 16.4.11**
  - OS and Application security best practices.
  - Must “harden” software exposed to the Internet
- **PCI 6.6 – Security**
  - Option 1:
    - Source code Analysis (Manual or Automated)
    - Vulnerability Assessment (Manual or Automated)
  - Option 2:
    - WAF
- **PCI - 11.3 – Penetration Testing**
  - Annually or after modifications
  - Network and Application Layer

Malware and our favorite search engine

- **Ghost in the Browser (May 2007)**
  - Google anti-malware team (Niels Provos)
- **All your iFrames point to us (Feb 2008)**
  - 3 million malicious URLs hosted on over 180,000 sites
  - 1.3% of incoming Search Queries return at least one URL with malicious code
- **Google Flagging malicious URLs from search**
  - Request http://www.stopbadware.org/ to remove Google warnings
  - If you are not in Google – you don't exist
- **What is the impact in your business if you do not show up a Google Search?**

Mandatory Industry Analyst Quotes

- **“More than 70% of attacks against a company are at the application layer, not the network layer”-** Gartner 2006
- **“Protecting networks is not enough. Applications are the real target for hackers” -** IDC 2006
- **Instead of bolting security on as an afterthought, Security 3.0 integrates compliance, risk assessment and business continuity into every process and application -** register.co.uk, 2007
- **“Developers don’t go to security conferences .... IT Security people expect developers to come to us and be shown the light, perhaps it should be the other way around -** Jeremiah Grossman June 16, 2008

Securing Web applications

armorize technologies
Secure Your Web Code

Securing Web applications

Security Testing

- Part of compliance process
- Often automated tool with human analysis
- Time saving should not be offset by False Positives or IT Overhead

Black Box Testing

- Assumes no prior knowledge of the infrastructure to be tested
- Emulate Hacker

White Box Testing

- Knowledge of the infrastructure to be tested
- Prepare for hacker

Gray Box

- Hybrid approach

Web Application Security - People

Advantages

- Human
- Expertise
- Flexible
- Validate results - Eliminate false positives (and false negatives)
- Show me what report actually means

Disadvantages

- Human
- Slow
- Expensive
- Flexible
- Not exactly repeatable
  - Unless you hire same consultant at same stage of learning curve

Web Application Security - Products

Advantages

- Machine
- Anyone can do it
- Fast
- Cheap
- Repeatable

Disadvantages

- Machine
- Anyone can do it
- False positives / negatives
- Report is meaningless unless it is understood (or at least read)
- Simply owning device does not make you secure

Black Box Vs White Box

Automated application vulnerability scanners

- Black Box - Penetration Testing Tools
- White Box - Source Code Analysis Tools
- http://en.wikipedia.org/wiki/Application_security

Black Box

- Watchfire, SPI Dynamics, Cenzic, N-Stalker
- Nikto, Wapiti
- Sandcat

White Box

- Armorize Technologies
- Fortify Software
- Ounce Labs

What am I trying to sell

- **Don’t fight automation**
  - Use it where appropriate
  - Automated hand in hand with Manual
- **Don’t limit yourself to one method**
  - White Box hand in hand with Black box
- **Source Code Analysis**
  - Develop secure applications
  - Absence of other safeguards
- **Pen Testing**
  - Audit/Test/Assess/Evaluate security
  - See application in real environment
- **Industry has promoted animosity**
  - Black Box Vs White Box
  - Consultants Vs Automate Process

The Bright Side of the Road

Static Source Code Analysis

- **White Box Testing - before deployment**
- **Analyze application source code without executing it**
- **Simulate all combinations of runtime behavior at compile time**
- **Create abstract program representation**
  - Symbolically executed
  - Generating warnings when anomalies are encountered
- **Everything a compiler does except create Binary**

Where to compile?

- **Integrate with compiler (e.g. on build server)**
  - Easier for vendor
  - Greater language support
  - Limited to information that compiler gives out
  - Installation and Maintenance overhead
- **Engine has own “compiler”**
  - Has algorithms to parse code and handle parser generated structure
  - Generates internal data for verification instead of binary executable
  - Very effective for languages that don’t have true compiler
  - Lower language coverage initially
    - As engine must have own compiler/interpreter
  - Lower overhead
    - No need to integrate with build server
    - Stand-alone system

False Alarms

False Negatives

- No product or person can get everything
- One product as a baseline
  - Compare whether other products find more or less
- If more, determine if they are false positives
- False negatives mean the product is not doing its job

False Positives

- Known code with known vulnerabilities
- Determine which product finds more vulnerabilities than there are
- False positives eat into time that should have been saved by automation

Trace Vulnerability through application

- **Trace from original flawed line of code to entry point**
  - Tainted Origin to Vulnerable Statement
- **Requires calculation of all possible states**
- **Backtracking does not work**
  - Incomplete Dataflow
- **Detailed trace back helps reduce false positives**
- **Without full trace back, white box is incomplete**

Code Types

Compiled Languages

- Java, C++
- Strong typing
- If verification fails then no binary (theory)
- Deterministic (somewhat) at run time
- “Easier” to analyze

Interpreted Languages

- PHP, Ruby on Rails, Python
- Dynamic interpretation at runtime
- More difficult to analyze
- Must interpret within analysis tool
- Greater accuracy on stand alone platform

Generations of SCA

1 – Soft Parsing

- Pattern Based
- Regular Expressions
- High False Positives and Negatives

2 – Software Checking

- Simple Verification Algorithms
- Heuristics
- Much lower false negatives but high false positives

3 – Software Verification

- Behavior based
- Simulate all possible run-time behaviors
- Built in compiler / interpreter
- Addresses doubling of state space with each conditional branch
- Trace each vulnerability back to line of code
- Control Flow Vs Data Flow

OWASP Top 10 and Source Code Analysis

A1. Cross Site Scripting (XSS)

A2. Injection Flaws

A3. Insecure Remote File Include

A4. Insecure Direct Object Reference

A5. Cross Site Request Forgery (CSRF)

A6. Information Leakage and Improper Error Handling

A7. Broken Authentication and Session Management

A8. Insecure Cryptographic Storage

A9. Insecure Communications

A10. Failure to Restrict URL Access

Integration and Potential Features

Appliance

- Browser or Client
- Enterprise level management
- Multiple projects / languages

Software

- Integration with build server
- Client Component

Service

- SaaS
- Source Code outside your control
- Binary Analysis

Repository Support

- Enterprise level scans

Integration and Potential Features II

IDE Integration

- Stand-alone IDE or Plug in for Eclipse, RAD, etc.
- Interface to engine or use local resources

Scheduling

- On-demand
- Integrate with check in process
- Automated

Policies and Reporting

- OWASP, CVE, (MITS?)
- Configurable
- Compliance based

Integrate with WAF

- May not be practical to rewrite
- Need to Mitigate

Source Code Analysis

- White box Testing
- Finds source code vulnerabilities
- Excellent way to address security early in development
- Bridges disconnect between security team and developers
- Once Vulnerabilities are identified
  - Rewrite Code
  - Or install WAF

Come over to the Dark Side

Penetration Testing

- **We are only looking at Web Application**
  - Overall testing should look at all OSI
- **Evaluate the security by simulating “hacker”**
  - Automated scanner
  - Accesses running application & environment through web interface
  - Identifies potential security weaknesses in web application
  - Detect the vulnerabilities by performing attacks
- **Commercial or Open Source**
  - It will have a Sales Pitch
- **Distinguish**
  - Penetration Testing and Vulnerability Management

False Alarms

False Negatives

- No product or person can get everything
- One product as a baseline
  - Compare whether other products find more or less
- If more, determine if they are false positives
- False negatives mean the product is not doing its job

False Positives

- Known application with known vulnerabilities
- Determine which product finds more vulnerabilities than there are
- False positives eat into time that should have been saved by automation

Common Disconnect

- **Penetration Testing Report**
  - Indicates SQL Injection vulnerabilities on specific pages
- **Dialog**
  - Security Team – “Fix them”
  - Developer – “Which one first?”
  - Security Team (after thinking) – “This one”
  - Developer “Which line of code should I change?”
  - Security Team – “I don’t know”
- **Do you have a product that can bridge that gap?**
- **Do you have an expert that can bridge that gap?**

What is a Pen Test Scanner Looking for? (far from exhaustive list)

- **Vulnerable Web Servers**
- **Dangerous HTTP methods**
- **Parameter Manipulation**
  - XSS, Injection, Redirection, etc.
  - This is actually a much longer list with significant crossover with Source Code Analysis
- **File/directory Checks**
  - Permissions, CVS, Backups
- **Known vulnerabilities in specific web applications**
- **Text Search (Directory listings, Source Code, Emails)**
- **Google Hacking Database**
- **Authentication attacks**

Penetration Testing

- **Black box Testing**
- **Detects impact of unresolved source code vulnerabilities**
  - Works really well in conjunction with manual tests
- **Vulnerabilities from configuration or architecture**
- **Disconnect**
  - No trace between entry point and vulnerable code
- **Once Vulnerabilities are identified**
  - Either rewrite application
  - Or install WAF

OWASP Top 10 and Pen Testing

A1. Cross Site Scripting (XSS)

A2. Injection Flaws

A3. Insecure Remote File Include

A4. Insecure Direct Object Reference

A5. Cross Site Request Forgery (CSRF)

A6. Information Leakage and Improper Error Handling

A7. Broken Authentication and Session Management

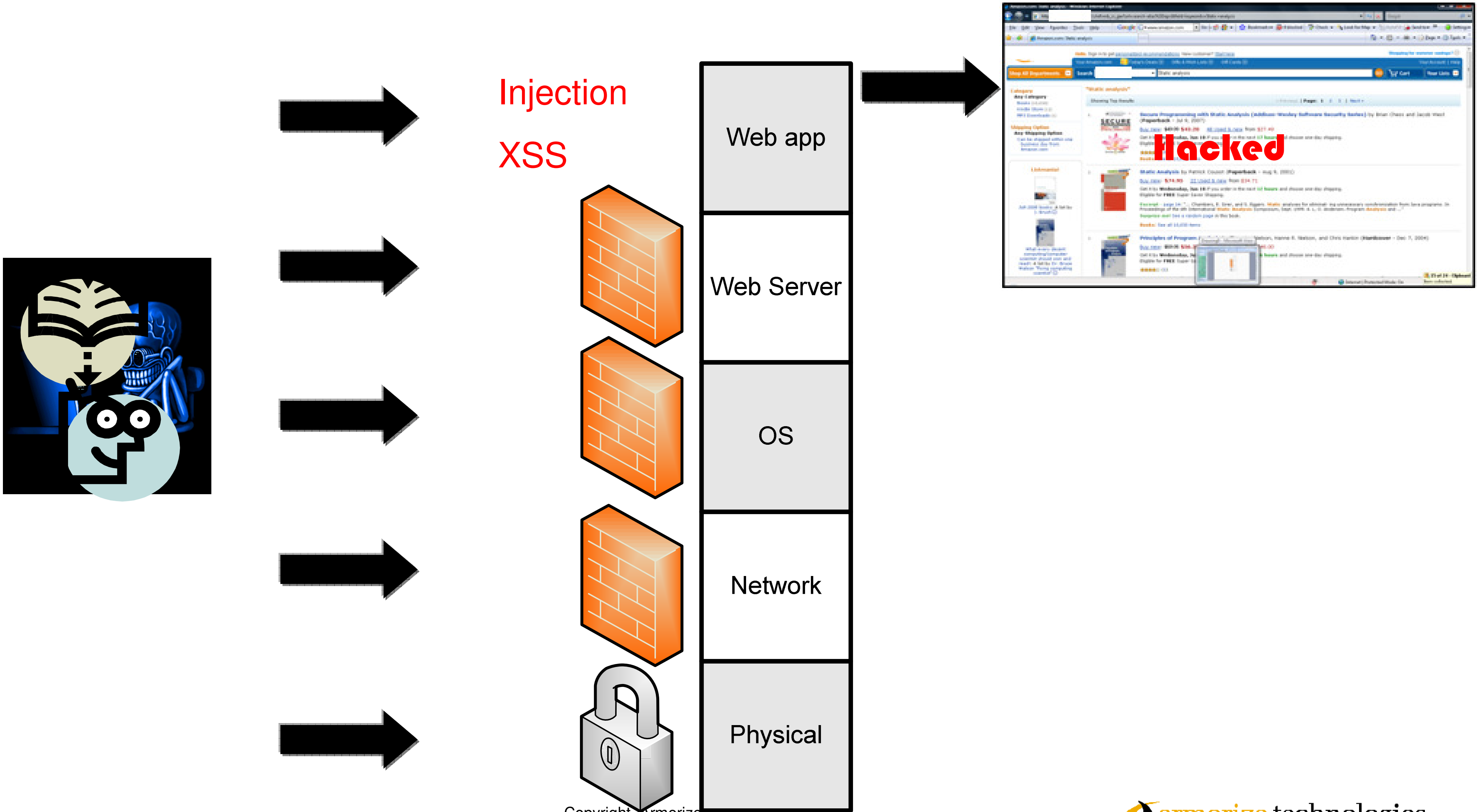
A8. Insecure Cryptographic Storage

A9. Insecure Communications

A10. Failure to Restrict URL Access

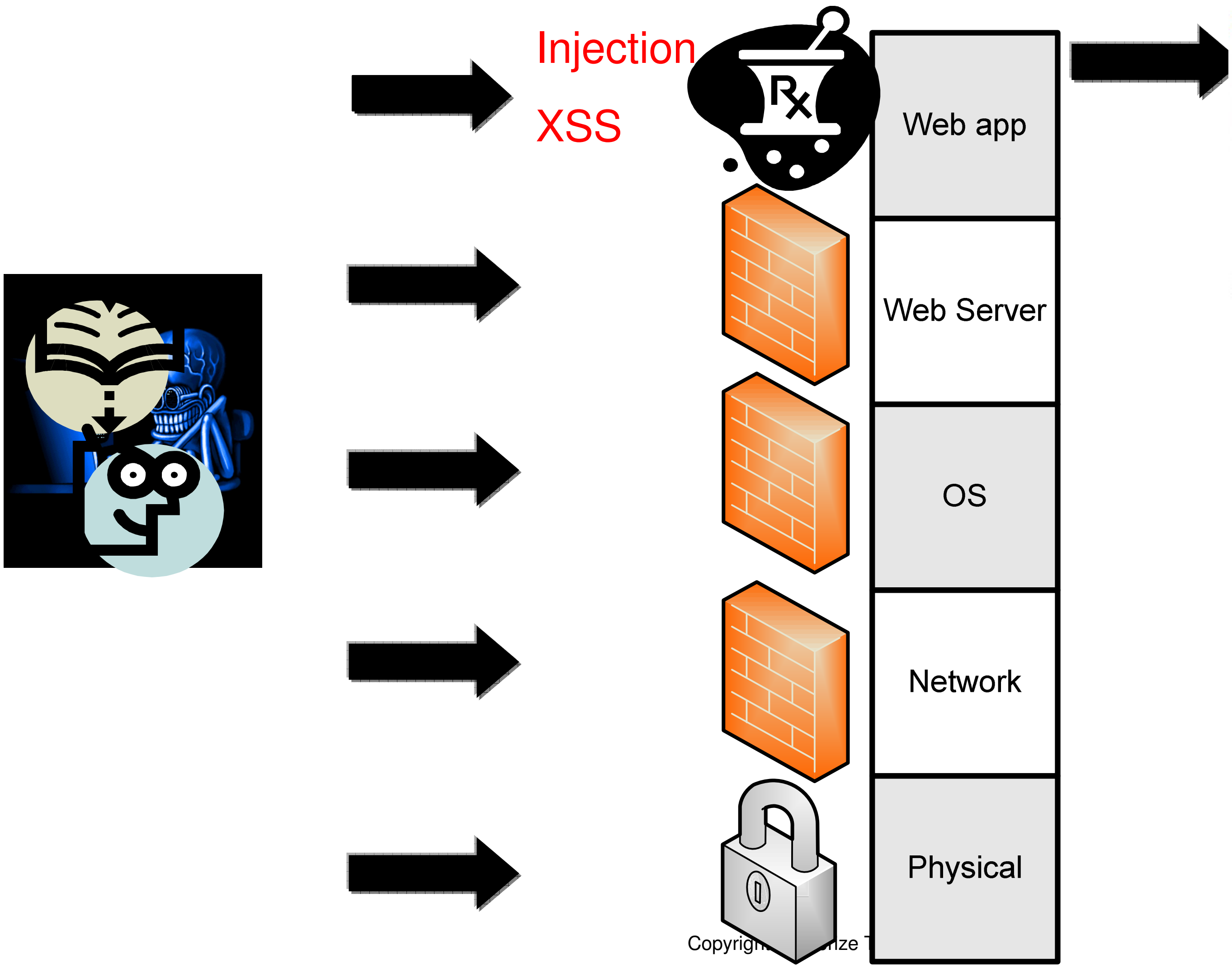
Tying it all Together

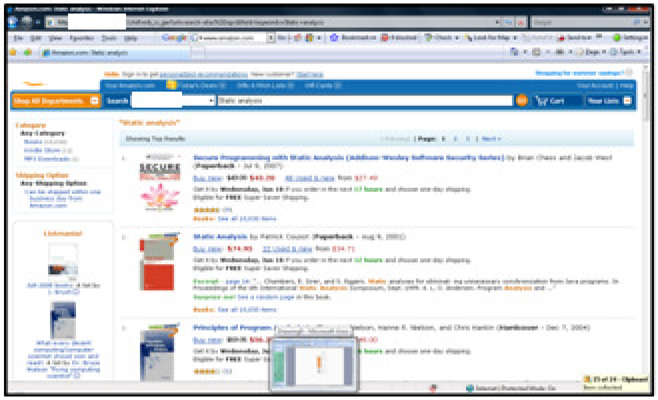
Remember our Hacker(s)?

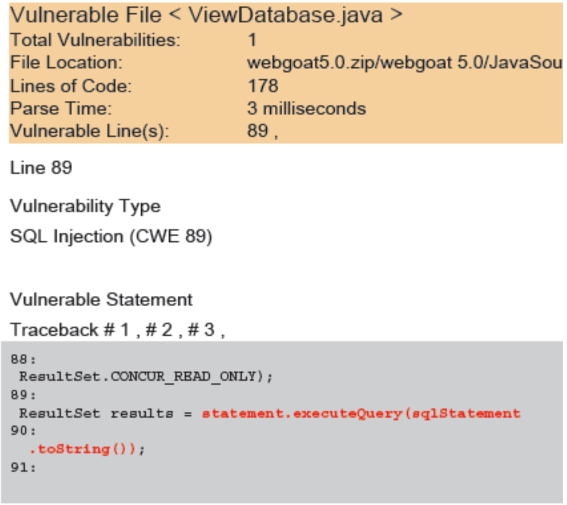


armorize technologies
Secure Your Web Code

Stop them with a securely built application







Vulnerable File < ViewDatabase.java >

Total Vulnerabilities:	1
File Location:	webgoat5.0.zip/webgoat 5.0/JavaSou
Lines of Code:	178
Parse Time:	3 milliseconds
Vulnerable Line(s):	89 ,

Line 89

Vulnerability Type

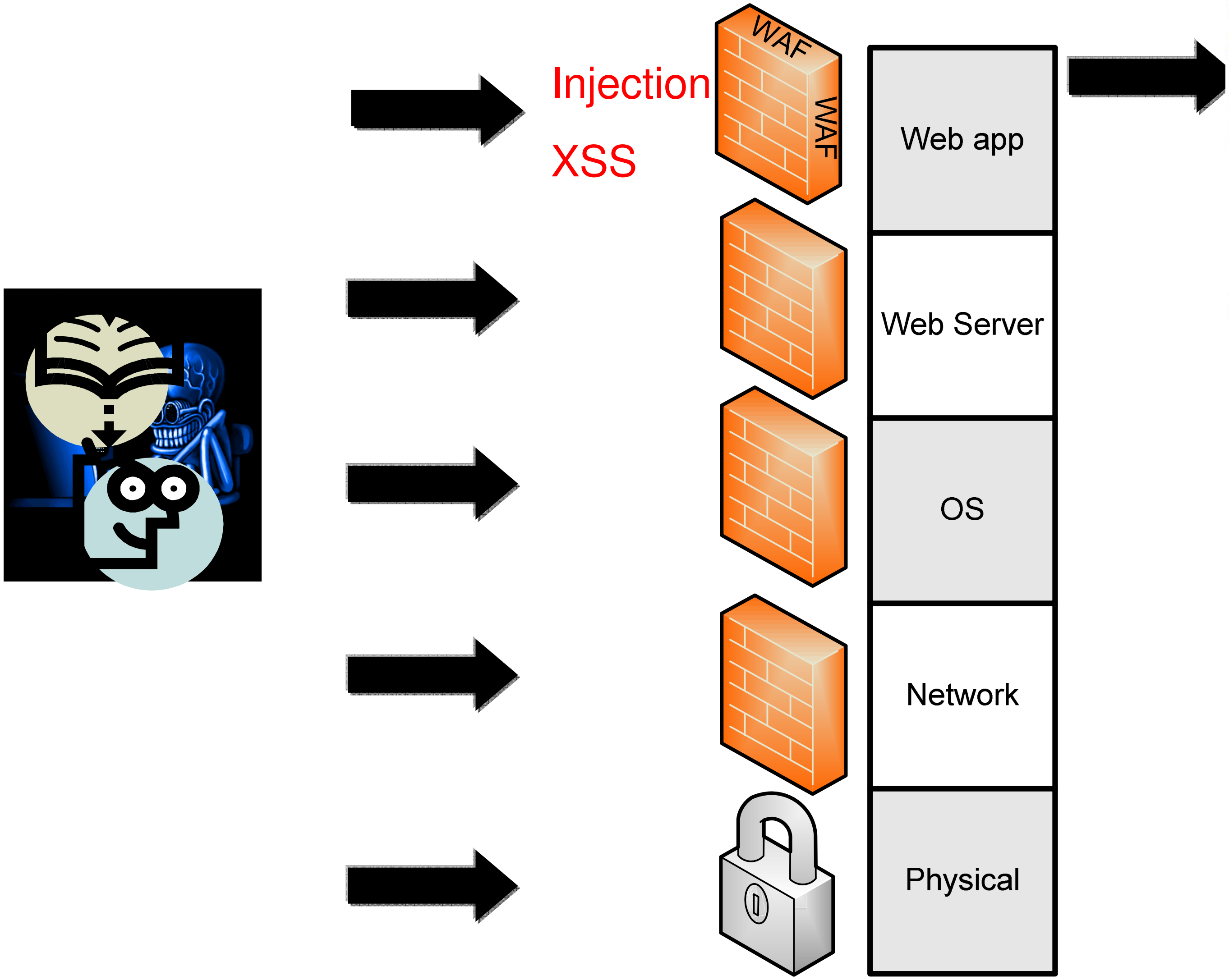
SQL Injection (CWE 89)

Vulnerable Statement

Traceback # 1 , # 2 , # 3 ,

```
88:
 ResultSet.CONCUR_READ_ONLY);
89:
 ResultSet results = statement.executeQuery(sqlStatement
90:
  .toString());
91:
```

Stop them with a Web Application Firewall



Injection

XSS

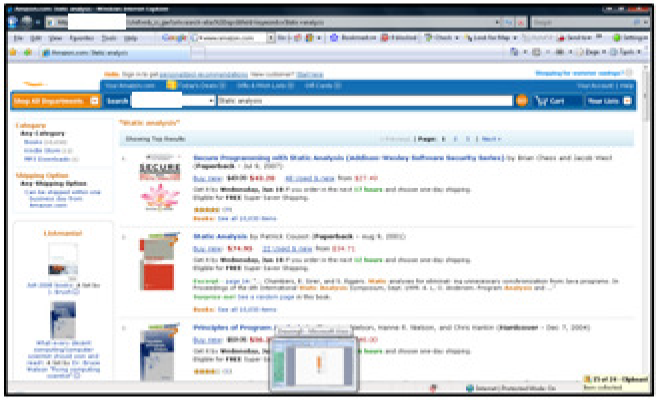
Web app

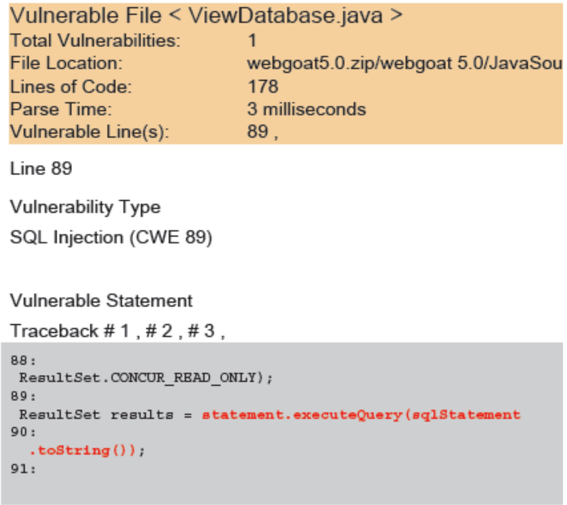
Web Server

OS

Network

Physical





Vulnerable File < ViewDatabase.java >

Total Vulnerabilities:	1
File Location:	webgoat5.0.zip/webgoat 5.0/JavaSou
Lines of Code:	178
Parse Time:	3 milliseconds
Vulnerable Line(s):	89 ,

Line 89

Vulnerability Type

SQL Injection (CWE 89)

Vulnerable Statement

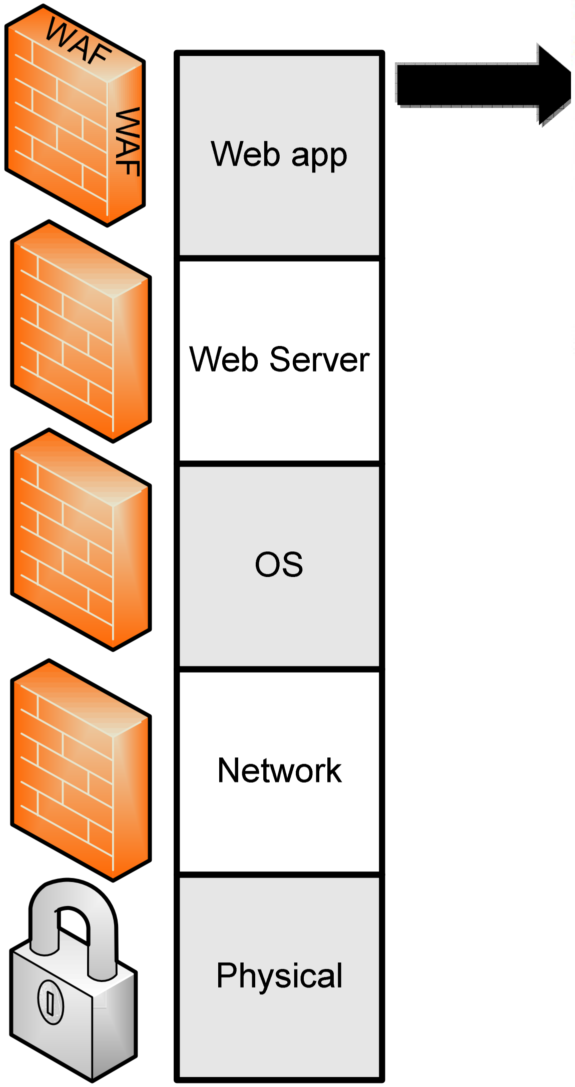
Traceback # 1 , # 2 , # 3 ,

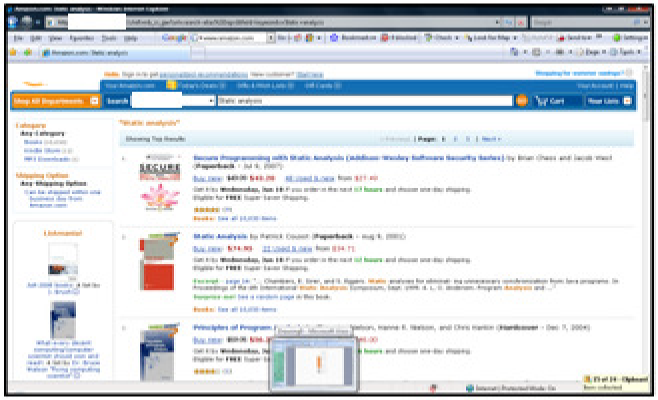
```
88:
 ResultSet.CONCUR_READ_ONLY);
89:
 ResultSet results = statement.executeQuery(sqlStatement
90:
   .toString());
91:
```

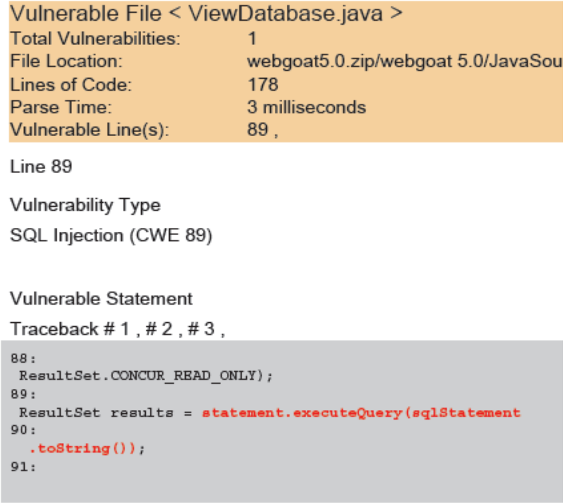
armorize technologies
Secure Your Web Code

Stop them with a Web Application Firewall

WAF Rules to prevent specific attacks against specific pages







Vulnerable File < ViewDatabase.java >	
Total Vulnerabilities:	1
File Location:	webgoat5.0.zip/webgoat 5.0/JavaSou
Lines of Code:	178
Parse Time:	3 milliseconds
Vulnerable Line(s):	89 ,

Line 89

Vulnerability Type

SQL Injection (CWE 89)

Vulnerable Statement

Traceback # 1 , # 2 , # 3 ,

```
88:
 ResultSet.CONCUR_READ_ONLY);
89:
 ResultSet results = statement.executeQuery(sqlStatement
90:
   .toString());
91:
```

Copyright. Armorize Technologies. 2007.

armorize technologies
Secure Your Web Code

Analyze the Code AND Scan the Application

Source Code Analysis (White box)

- **White box Testing**
- **Finds source code vulnerabilities**
- **Excellent way to address security early in development**
- **Bridges disconnect between security team and developers**
- **Once Vulnerabilities are identified**
  - Rewrite Code
  - Or install WAF

Analyze the Code AND Scan the Application

Penetration Testing (Black box)

- **Black box Testing**
- **Detects impact of unresolved source code vulnerabilities**
- **Vulnerabilities from configuration or architecture**
- **Disconnect**
  - No trace between entry point and vulnerable code
- **Once Vulnerabilities are identified**
  - Either rewrite application
  - Or install WAF

Analyze the Code AND Scan the Application

- **Complementary processes eased by automation**
- **When to use Source Code Analysis***
  - During Development
  - Security Team - Enterprise Level
  - Developers - IDE Integration
- **When to Pen Test***
  - After development
  - Once application is “ready”
  - Time intervals, upgrades, compliance regulations
  - New exploit or newly discovered hole
- **What about WAF**
  - Should be part of perimeter security
  - If rebuilding application is not an option
  - Then patch the holes with WAF

Can we reverse this?

Source Code Analysis After Deployment

- Bridges disconnect between Pen Tester and Developer
- Guide Penetration Test (gray box)
- Informed decision on rewrite or block

Pen Testing during Development

- Are whole applications or workable modules available?
- Do you have time to fix before deployment?
- Does not test mitigating perimeter controls

Summary

- **Stop looking for Silver Bullets**
- **People, Process and Technology**
  - Smart use of Technology
- **Source Code Analysis to build securely**
- **Source Code Analysis to bridge disconnect**
  - Security and Developers
- **Penetration Testing to test security**
- **Some scope for reversal**

Thank You!