Why Johnny Can’t Pentest: An Analysis of Black-box Web Vulnerability Scanners

Adam Doupé, Marco Cova and Giovanni Vigna
University of California, Santa Barbara
DIMVA 2010 – 7/8/10
Outline

- Introduction to black box web vulnerability scanners
- Design of custom vulnerable website – WackoPicko
- Results
- Analysis
Contributions

- Describe the design of a testing web application
- Identify a number of challenges that scanners need to overcome when testing modern web applications
- Test the performance of eleven real-world scanners and identify areas that need further work
Web Application Vulnerability Scanners

Diagram: Server, Crawler, Attack, Analysis
Vulnerable Web Application – WackoPicko: Design

- Authentication
- Upload Pictures
- Comment on Pictures
- “Purchase” Pictures
- Tag Search
- Guestbook
- Admin Area
Vulnerable Web Application – WackoPicko: Publicly Accessible

- XSS
  - Reflected, Stored, and Reflected behind JavaScript
- Session ID
- Weak Password
- Reflected SQL Injection
- Command Line Injection
- File Inclusion
- File Exposure
- Parameter Manipulation
Vulnerable Web Application – WackoPicko: Authentication

- Reflected XSS behind Flash
- Stored SQL Injection
- Directory Traversal
- Multi-step Stored XSS
- Forceful Browsing
- Logic Flaw
Crawling Challenges

- HTML Parsing
- Multi-Step Process / State
- Infinite Website
- Authentication
- Client-side Code
  - Web Input Vector Extractor Teaser (WIVET)
# Scanners

<table>
<thead>
<tr>
<th>Name</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acunetix</td>
<td>$4,995 – $6,350</td>
</tr>
<tr>
<td>AppScan</td>
<td>$12,550 – $32,500</td>
</tr>
<tr>
<td>Burp</td>
<td>£125 ($190.82)</td>
</tr>
<tr>
<td>Grendel-Scan</td>
<td>Open source</td>
</tr>
<tr>
<td>Hailstorm</td>
<td>$10,000</td>
</tr>
<tr>
<td>Milescan</td>
<td>$495 – $1,495</td>
</tr>
<tr>
<td>N-Stalker</td>
<td>$899 – $6,299</td>
</tr>
<tr>
<td>NTOSpider</td>
<td>$10,000</td>
</tr>
<tr>
<td>Paros</td>
<td>Open source</td>
</tr>
<tr>
<td>w3af</td>
<td>Open source</td>
</tr>
<tr>
<td>Webinspect</td>
<td>$6,000 – $30,000</td>
</tr>
</tbody>
</table>
Experiment

- Each scanner run four times:
  - WackoPicko
    - Initial – No configuration (point and click)
    - Config – Given valid Username/Password
    - Manual – Used proxy to thoroughly browse site.
  - WIVET – Testing JavaScript capabilities

- Limitations
## Results

<table>
<thead>
<tr>
<th>Name</th>
<th>Reflected XSS</th>
<th>Stored XSS</th>
<th>SQL Injection</th>
<th>Command Line Injection</th>
<th>File Inclusion</th>
<th>File Exposure</th>
<th>XSS via JavaScript</th>
<th>XSS via Flash</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acunetix</td>
<td>Initial</td>
<td>Initial</td>
<td>Initial</td>
<td>Initial</td>
<td>Initial</td>
<td>Initial</td>
<td>Initial</td>
<td>Initial</td>
</tr>
<tr>
<td>AppScan</td>
<td>Initial</td>
<td>Initial</td>
<td>Initial</td>
<td>Initial</td>
<td>Initial</td>
<td>Initial</td>
<td>Initial</td>
<td>Initial</td>
</tr>
<tr>
<td>Burp</td>
<td>Initial</td>
<td>Manual</td>
<td>Initial</td>
<td>Initial</td>
<td>Initial</td>
<td>Initial</td>
<td></td>
<td>Manual</td>
</tr>
<tr>
<td>Grendel-Scan</td>
<td>Manual</td>
<td>Config</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hailstorm</td>
<td>Initial</td>
<td>Config</td>
<td>Config</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Manual</td>
</tr>
<tr>
<td>Milescan</td>
<td>Initial</td>
<td>Manual</td>
<td>Config</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NTOSpider</td>
<td>Initial</td>
<td>Initial</td>
<td>Initial</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Paros</td>
<td>Initial</td>
<td>Initial</td>
<td>Config</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Manual</td>
</tr>
<tr>
<td>w3af</td>
<td>Initial</td>
<td>Manual</td>
<td>Initial</td>
<td></td>
<td>Initial</td>
<td></td>
<td></td>
<td>Manual</td>
</tr>
<tr>
<td>Webinspect</td>
<td>Initial</td>
<td>Initial</td>
<td>Initial</td>
<td></td>
<td>Initial</td>
<td></td>
<td></td>
<td>Manual</td>
</tr>
</tbody>
</table>
Results

[Bar chart showing the results of different tools in various modes: False negatives, Detection in MANUAL mode, Detection in CONFIG mode, and Detection in INITIAL mode.]

- AcuNetix
- AppScan
- Burp
- Grendel-Scan
- Hailstorm
- MileScan
- N-Stalker
- NTOSpider
- Paros
- w3af
- Webspect
Missed Vulnerabilities

- Missed by all scanners
  - Session ID
  - Weak Password
  - Parameter Manipulation
  - Forceful Browsing
  - Logic Flaw

- Will discuss later
  - Stored SQL Injection
  - Directory Traversal
  - Stored XSS Behind Login
False Positives

- Ranged from 0 to 200+
  - Average was ~25
- Why?
  - Server Path Disclosure
- “Actual” False Positives
  - Hailstorm
    - XSS, 2 Code Injection
  - NTOSSpider
    - 3 XSS
  - w3af
    - PHP eval() Injection
Measuring and Comparing Detection Capabilities

- Strictly Dominates
Dominates Graph

More Dominant

Less Dominant
Attack and Analysis Capabilities

- Default values
- XSS attacks
- Command-line Injection
- SQL Injection
- File Exposure
- Remote Code Execution
Crawling Capabilities

- Number of Accesses
  - Range from ~50 per page to ~3,000 per page
  - Hailstorm accessed vulnerable pages that required an account on INITIAL scan!

- HTML
  - Burp and N-Stalker
    - `<TEXTAREA>`
  - Milescan and Grendel-Scan
    - POST
  - Hailstorm
    - No-Injection
  - w3af
    - No Default
Crawling Capabilities

- Uploading a Picture
  - 2 Scanners uploaded without help
  - 3 Scanners unable to upload one!
Crawling Capabilities – Client-side Code

- **WIVET**
  - 3 Scanners couldn’t complete
    - Paros and Burp – `<base>`
    - N-Stalker – Frame?
  - Dynamic JavaScript
    - Webinspect, Acunetix, NTOSpider, Hailstorm
  - JavaScript library
  - No Flash
Crawling Capabilities – Authentication

- Created an account successfully
  - 4 Scanners
    - Hailstorm
    - N-Stalker
    - NTOSpider
    - WebInspect
Lessons Learned: Want to make your own benchmark?

- Incorporate lots of logging in the application
- Two versions of the site
  - No vulnerabilities
  - All vulnerabilities
- Script running the tests
- Include:
  - File upload forms
  - AJAX
  - Several JavaScript UI Libraries
Conclusions

- Ability to crawl as important as detection
- Many vulnerabilities cannot be detected
- Cost not directly proportional to functionality
Questions?
Thanks!