Web Maniac
Hacking Trust

Aditya K Sood [adi ks [at] secniche.org]
SecNiche Security
Web Maniac - Hacking Trust

Pentesting web applications in a hacker’s way. Attack surface varies from application to application. How to think below the surface? That’s the aim!

All contents of this presentation represent my own beliefs and views and do not, unless explicitly stated otherwise, represent the beliefs of my current, or any of my previous in that effect, employers.

Screenshots have been shared from various resources. This is done to show the comparative model of various methodologies.
About Me

- Founder, SECNICHE Security Labs.  
  [http://www.secniche.org](http://www.secniche.org)

- PhD Candidate at Michigan State University

- Worked previously for Armorize as Senior Security Practitioner, COSEINC as Senior Security Researcher and Security Consultant for KPMG

- Written content Author for HITB E-Zine, Hakin9, ELSEVIER, USENIX Journals.
- Like to do Bug Hunting and Malware dissection.
- Released Advisories to Forefront Companies.
- Active Speaker at Security Conferences including RSA etc.

- Blog: [http://secniche.blogspot.com](http://secniche.blogspot.com) | [http://zeroknock.blogspot.com](http://zeroknock.blogspot.com)
Agenda

✓ Web Application Security Standards
✓ Web Application Security - A view of Reality
✓ Web Application – Testing and Development Methodologies
✓ Facets of Web Application Pen Testing (WAPT)
✓ Demonstrations – Live Targets
## Web Application Security Standards – Really

<table>
<thead>
<tr>
<th>WASC Threat Classification v2</th>
<th>OWASP Top Ten 2010 RC1</th>
</tr>
</thead>
<tbody>
<tr>
<td>WASC-19 SQL Injection</td>
<td>A1 - Injection</td>
</tr>
<tr>
<td>WASC-23 XML Injection</td>
<td>A1 - Cross Site Scripting (XSS)</td>
</tr>
<tr>
<td>WASC-28 Null Byte Injection</td>
<td>A7 - Broken Authentication and Session Management</td>
</tr>
<tr>
<td>WASC-29 LDAP Injection</td>
<td>A4 - Insecure Direct Object Reference</td>
</tr>
<tr>
<td>WASC-30 Mail Command Injection</td>
<td>A5 - Cross Site Request Forgery (CSRF)</td>
</tr>
<tr>
<td>WASC-31 OS Commanding</td>
<td>A6 - Insecure Configuration Management</td>
</tr>
<tr>
<td>WASC-39 XPath Injection</td>
<td>A8 - Insecure Cryptographic Storage</td>
</tr>
<tr>
<td>WASC-46 XQuery Injection</td>
<td>A9 - Insecure Cryptographic Storage</td>
</tr>
<tr>
<td>WASC-08 Cross-Site Scripting</td>
<td>A3 - Broken Authentication and Session Management</td>
</tr>
<tr>
<td>WASC-01 Insufficient Authentication</td>
<td>A4 - Insecure Direct Object References</td>
</tr>
<tr>
<td>WASC-18 Credential/Session Prediction</td>
<td>A7 - Failure to Restrict URL Access</td>
</tr>
<tr>
<td>WASC-37 Session Fixation</td>
<td>A10 - Failure to Restrict URL Access</td>
</tr>
<tr>
<td>WASC-47 Insufficient Session Expiration</td>
<td>A8 - Unvalidated Redirects and Forwards (NEW)</td>
</tr>
<tr>
<td>WASC-01 Insufficient Authentication</td>
<td>A3 - Malicious File Execution</td>
</tr>
<tr>
<td>WASC-02 Insufficient Authorization</td>
<td>A9 - Insecure Communications</td>
</tr>
<tr>
<td>WASC-33 Path Traversal</td>
<td>A10 - Insufficient Transport Layer Protection</td>
</tr>
<tr>
<td>WASC-09 Cross-site Request Forgery</td>
<td>A5 - Cross-Site Request Forgery</td>
</tr>
<tr>
<td>WASC-14 Server Misconfiguration</td>
<td>A6 - Security Misconfiguration</td>
</tr>
<tr>
<td>WASC-15 Application Misconfiguration</td>
<td>A7 - Failure to Restrict URL Access</td>
</tr>
<tr>
<td>WASC-10 Denial of Service</td>
<td>A10 - Insufficient Transport Layer Protection</td>
</tr>
<tr>
<td>WASC-11 Brute Force</td>
<td>A7 - Failure to Restrict URL Access</td>
</tr>
<tr>
<td>WASC-21 Insufficient Anti-automation</td>
<td>A8 - Unvalidated Redirects and Forwards</td>
</tr>
<tr>
<td>WASC-34 Predictable Resource Location</td>
<td>A9 - Insecure Cryptographic Storage</td>
</tr>
<tr>
<td>WASC-38 URL Redirector Abuse</td>
<td>A8 - Unvalidated Redirects and Forwards</td>
</tr>
<tr>
<td>WASC-50 Insufficient Data Protection</td>
<td>A9 - Insecure Cryptographic Storage</td>
</tr>
<tr>
<td>WASC-04 Insufficient Transport Layer Protection</td>
<td>A10 - Insufficient Transport Layer Protection</td>
</tr>
</tbody>
</table>

### OWASP Top 10 – 2007 (Previous) vs OWASP Top 10 – 2010 (New)

- **A2 – Injection**
- **A1 – Cross Site Scripting (XSS)**
- **A7 – Broken Authentication and Session Management**
- **A4 – Insecure Direct Object Reference**
- **A5 – Cross Site Request Forgery (CSRF)**
- **A6 – Security Misconfiguration (NEW)**
- **A10 – Failure to Restrict URL Access**
- **A7 – Failure to Restrict URL Access**
- **A8 – Unvalidated Redirects and Forwards (NEW)**
- **A9 – Insecure Cryptographic Storage**

At higher levels in ASVS, the use of tools is encouraged. But to be effective, the tools must be heavily tailored and configured to the application and framework in use.
• Standards provide specific classification of vulnerabilities
• Do they comprise of all types of vulnerabilities?
• Are all types of web attacks predefined in them?
• Do you think the design of web application matters? [to what extent]
• A view of web application and a website under testing.
• Do platforms and web servers matter while web application assessment?
• Do you think penetration testing of web applications is beyond these standards?
Web Application State and Risks

Threat Agents

Attack Vectors

Security Weaknesses

Security Controls

Technical Impacts

Business Impacts

Attack

Weakness

Control

Asset

Function

Impact

Static Data

Active Data

Sandboxed Application

Application

© OWASP
Web Application Architecture - Development
Web Application Testing - Methodologies

- Discovery
  - Collect network infrastructure information, develop web application footprint

- Automated Analysis
  - Automated scan for common technical vulnerabilities (e.g., SQL Injection, XSS etc)

- Manual Analysis
  - Elimination of false positives, manual checks for false negatives, manual checks for logical vulnerabilities.

- Exploit
  - Exploit technical and logical vulnerabilities, provide proof-of-concept

Application Penetration Testing
- Application Spidering
- Authentication Testing
- Session Management Testing
- Data Validation Testing

- Web Service Testing
- Ajax Testing
- Business Logic Testing
- Risk Management
- Reporting

Penetration Testing
- Vulnerability Remediation
- Architectural Analysis

- Code Review
Why Security Testing?

Defacement Statistics

© Zone H
Web Application Security! Reality - Broken
Is that Ethical?
Existence and Reality – Web Penetration Test

- Is this all about compliance (PCI)?
- Is this all about reporting generic issues and using reports for cert’s?
- Do you think organizational teams patch all the reported issues?
- White box or Black box – Changed definitions.

- Security Assessment ≠ Penetration Testing [Mismatch]

- Time dependency – A big factor in determining the effectiveness

- Penetration Tests – Does not provide security / That’s the Truth
  - Applied security comes out of the actions taken to remove those vulnerabilities which are exploited during the course of penetration testing.
  - Vulnerability assessment provides a glimpse of security to some degree
  - Penetration tests emulate real world attacks to exploit the network and web infrastructure
  - Effective penetration tests provide a degree to which systems can be exploited. It can be more.
Pentesting Stringency in Real World
Is that True?

Pen-testing is overrated

Then what about Human Ignorance?

A critical component in every sphere. Hard to beat it.
Thinking in the Wild – Web Penetration Testing.

Is it all about shooting what we see?

Do we need to take care of the hidden or shadowed?
Web Penetration Test – The Refined Art

• Turning the Black Box Testing into White box Testing

• Expertise – Hacking in a controlled manner

• Meeting the expectations
• Pen Tester – The Word of Advice

“Everything that goes wrong on the target host, network, or on the Internet from two weeks before you plug in to two weeks after you submit the report will be your fault.”
Demonstration

- Attacking Web Apps through Content Rendering – 4:15 M
- SQLXSSI – XSS through SQL Injections : Yahoo – 5:30 M
- Persistent Redirection Attacks and Malware - 4:00 M
- Content Delivery Networks – Infection Behavior - 4:09 M
- Widget Redirection Attacks – Outbrain – 3:20 M
Demo 1: Document Content Rendering Attacks

Demonstrations

Document Hyperlinking XSS Payload Rendering Attacks

Vulnerability in Web applications and Enterprise Solutions

This issue is a potential part of web application functionality
Disclaimer - This is only for demonstration purposes. All on your mobility.

Proof of Concept

Step 1 - Finding a upload module in the web application and understanding the design
Step 2 - Setting XSS payload in MSWord Hyperlinks
Step 3 - Uploading the file on the web application and allow it to render
Step 4 - Rendering the MS Word document while translation results in XSS
Step 5 - All at your own risk.

(C) SecNiche Security (http://www.secniche.org)
Demo 2: SQLXSSI – Using SQLI to conduct XSS

Demo 2: SQLXSSI – Using SQLI to conduct XSS

Demonstrations

SQLXSSI – XSS Payloads in SQL Variables

Database – MYSQL / MSSQL / ORACLE

Vulnerability in Web applications and Enterprise Solutions

It has been transferred into a malware demo for understanding and leveraging knowledge out of the issue.
Disclaimer - This is only for demonstration purposes. All on your mobility.

Proof of Concept

Step 1 – Finding a vulnerable website throwing reflective errors
Step 2 – Exploiting through UNION SQL poisoning with XSS payloads
Step 3 – Reflective error readers XSS payload in context of browser
Step 4 – Possibility of all sorts of attacks.

(C) SecNiche Security (http://www.secniche.org)
Demo 3: Persistent Logout Redirection Attacks

Demonstrations

Persistent Redirection - Logout Module - Malware Infection - Attacks

Vulnerability in Web applications and Enterprise Solutions

This issue is a potential redirection attack which is persistent in logout module of an enterprise application. It has been transferred into a malware demo for understanding and leveraging knowledge out of the issue.

Disclaimer: This is only for demonstration purposes. All on your mobility.

Proof of Concept

Step 1 - Analysing the URL and inherent application parameters. Debug it to get the parameter used for redirection on logout module.
Step 2 - URL is designed with malicious link as a logout redirection parameter value
Step 3 - User is allowed to log into the application
Step 4 - User logged in and started doing some work and logged out.
Step 5 - During logout redirection occurs to malicious website and starts downloading files into system.

(C) SecNiche Security [ http://www.secniche.org ]
Demo 4: Third Party Content Delivery Infections

Demonstrations

Content Delivery Networks – Website Hacking and Malware Infection

Third Party Content Rendering

Vulnerability in Web applications and Enterprise Solutions

It has been transferred into a malware demo for understanding and leveraging knowledge out of the issue.

Disclaimer - This is only for demonstration purposes. All on your mobility.

Proof of Concept

(C) SecNiche Security (http://www.secniche.org)
Demo 5: Widget Redirection Attacks

Demonstrations

Widget Redirection Attacks – Malware Infection

Third Party Content Rendering

Vulnerability in Web applications and Enterprise Solutions

It has been transferred into a malware demo for understanding and leveraging knowledge out of the issue.

Disclaimer - This is only for demonstration purposes. All on your mobility.

Proof of Concept

(C) SecNiche Security (http://www.secniche.org)
Questions and Queries
Thanks

SecNiche Security: http://www.secniche.org

Hacker Halted – http://www.hackerhalted.com