Top 10 Web 2.0 Attacks

Shreeraj Shah

Who Am I?

- **Founder & Director**
  - Blueinfy Solutions Pvt. Ltd. (Brief)
  - SecurityExposure.com
- **Past experience**
  - Net Square, Chase, IBM & Foundstone
- **Interest**
  - Web security research
- **Published research**
  - Articles / Papers – Securityfocus, O’erilly, DevX, InformIT etc.
  - Tools – wsScanner, scanweb2.0, AppMap, AppCodeScan, AppPrint etc.
  - Advisories - .Net, Java servers etc.
- **Books (Author)**
  - Web 2.0 Security – Defending Ajax, RIA and SOA
  - Hacking Web Services
  - Web Hacking
Attack Surface for Web 2.0

Real Life Cases

• Applications reviewed – Banking, Telecom and Portals
• Vulnerabilities and Exploits
  – DOM based attacks
  – Blind Injections over JSON/XML
  – Authentication bypass (LDAP/XPATH)
  – Client side logic disclosure
  – CSRF over XML, AMF and JSON
  – And many more …
Technology Trends

- Web 2.0 – Ajax, Silverlight and Flex/Flash
- Web Services and SOA
- Cloud APIs and SaaS
- Browser empowering – HTML 5 and several other features
- Traditional stacks are evolving around frameworks

Past, Present and Future

- Attack surface
- 2010
- Cloud
- 1993
- Time
- 2008
- Complexity & Security
- CGI
- Application Frameworks
- Static Pages (HTML)
Web Attacks and Targets

- 80% Sites are having security issues
- Web Application Layer vulnerabilities are growing at higher rate in security space
- Client side hacking and vulnerabilities are on the rise – from 5% to 30% (IBM)
- Web browser vulnerabilities is growing at high rate
- End point exploitation shifting from OS to browser and its plugins

Web 2.0 Attacks

- **Cross Site Scripting (XSS)**
  - Web 2.0 systems such as social networks, blogs or wikis, making Web 2.0 applications especially vulnerable to XSS.
  - DOM based XSS
  - Ajax and Flash issues
- **Cross Site Request Forgery (CSRF) / Cross Gadget Request Forgery (CGRF)**
  - Phishing – DOM based redirects
  - Information Leakage – client side leaks and errors/exceptions
  - Injection Flaws – XPATH, LDAP, JavaScript & JSON
  - Information Integrity – mashups and un-trusted sources
  - Insufficient Anti-Automation – CSRF and Phishing
### Hacks & Attacks

- Unauthenticated Access via Default or Shared Credentials
- SQL Injection
- Transport Layer Security
- Cross Site Scripting
- Cross Site Request Forgery
- Insufficient Transport Layer Protection
- Session Fixation
- Insecure Deserialization
- Uncontrolled Resource Consumption
- Information Leakage
- SQL Injection
- Unrestricted Direct Object Reference
- Cross-Site Request Forging
- Security Misconfiguration
- Other

### AppSec dynamics

#### New Top Ten 2004
- A1 Unvalidated Input
- A2 Broken Access Control
- A3 Broken Authentication and Session Management
- A4 Cross Site Scripting (XSS) Flaws
- A5 Buffer Overflows
- A6 Injection Flaws
- A7 Improper Error Handling
- A8 Insecure Storage
- A9 Denial of Service
- A10 Insecure Configuration Management

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

<table>
<thead>
<tr>
<th>OWASP Top 10 – 2007 (Previous)</th>
<th>OWASP Top 10 – 2010 (New)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1 Injection</td>
<td>A1 Injection</td>
</tr>
<tr>
<td>A2 Cross Site Scripting (XSS)</td>
<td>A2 Cross Site Scripting (XSS)</td>
</tr>
<tr>
<td>A3 Broken Authentication and Session Management</td>
<td>A3 Broken Authentication and Session Management</td>
</tr>
<tr>
<td>A4 Cross Site Request Forgery (CSRF)</td>
<td>A4 Cross Site Request Forgery (CSRF)</td>
</tr>
<tr>
<td>A5 Insufficient Transport Layer Protection</td>
<td>A5 Insufficient Transport Layer Protection</td>
</tr>
<tr>
<td>A6 Insecure Deserialization</td>
<td>A6 Insecure Deserialization</td>
</tr>
<tr>
<td>A7 Misuse of Cryptographic Storage</td>
<td>A7 Misuse of Cryptographic Storage</td>
</tr>
<tr>
<td>A8 Insecure Credentials</td>
<td>A8 Insecure Credentials</td>
</tr>
<tr>
<td>A9 Denial of Service</td>
<td>A9 Denial of Service</td>
</tr>
<tr>
<td>A10 Information Leakage</td>
<td>A10 Information Leakage</td>
</tr>
</tbody>
</table>

- New in 2010:
  - A2, A3, A4, A5, A6, A7, A8, A9, A10
- Removed from 2010:
  - A1, A2, A3, A4, A5, A6, A7, A8, A9, A10
Next Generation Architecture

Breaking traditional layers

Server side Components

Client side Components (Browser)

Presentation Layer

Business Layer

Utility Layer
Data Access
Authentication
Communication etc.

Runtime, Platform, Operating System Components
Threat Model for 2.0

<table>
<thead>
<tr>
<th>Changing dimension</th>
<th>Traditional</th>
<th>Web 2.0/RIA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entry points</td>
<td>Structured</td>
<td>Scattered and multiple</td>
</tr>
<tr>
<td>Dependencies</td>
<td>Limited</td>
<td>• Multiple technologies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Information sources</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Protocols</td>
</tr>
<tr>
<td>Vulnerabilities</td>
<td>Server side [Typical injections]</td>
<td>• Web services [Payloads]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Client side [XSS &amp; XSRF]</td>
</tr>
<tr>
<td>Exploitation</td>
<td>Server side exploitation</td>
<td>Both server and client side</td>
</tr>
<tr>
<td></td>
<td></td>
<td>exploitation</td>
</tr>
</tbody>
</table>

Top 10 Attacks
Top Attacks

1. Dom based XSS – Ajax
2. SQL injection – SOAP & XML
3. Blind SQL over JSON
4. Auth Bypass- XPATH and LDAP
5. Business Logic Bypass
6. Decompilation Attack and Info Leakage
7. WSDL scanning and API exposure - Cloud
8. XSS with Flash
9. CSRF with XML
10. Widgets/Mashup Exploitation

A1 – XSS with DOM

• Ajax based XSS is relatively new way of attacking the client
• Code written on browser end can be vulnerable to this attacks
• Various different structures can have their own confusion
• Information processing from un-trusted sources can lead to XSS
Anatomy of an XSS attack

attacker

Web Client

Third party

Web Server

proxy

Web app

Web app

DB

DB

attacker

Web Client

Third party source

Web Server

proxy

Web app

Web app

DB

DB
Anatomy of an XSS attack

DOM based XSS

```javascript
if (http.readyState == 4) {
    var response = http.responseText;
    var p = eval("" + response + "");
document.open();
document.write(p.firstName +"<br>");
document.write(p.lastName +"<br>");
document.write(p.phoneNumbers[0]);
document.close();
```
Anatomy of an XSS attack

DOM based XSS

document.write(…)
document.writeln(…)
document.body.innerHTML=…
document.forms[0].action=…
document.attachEvent(…)
document.execCommand(…)
document.body. …
window.attachEvent(…)
document.location=…
document.location.hostname=…
document.location.replace(…)
document.location.assign(…)
document.URL=…
window.navigate(…)

© Blueinfy Solutions
A2. SQL injection – SOAP/XML

- XML streams and SOAP are common communication mechanism
- We are no longer using simple name value pairs over HTTP now
- Poisoning SOAP/XML stream
- Can access SOAP directly and inject variants
- Possible to own the system or back-end

A3. Blind SQL over JSON/AMF

- JSON/AMF calls are popular with frameworks
- Developers are using them very frequently
- JSON/AMF errors are hidden and may not come out
- It may be there as 200 OK
- Blind SQL over JSON is possible
- Needs to analyze behavior of the stream to identify issues - AMF
A4. Auth Bypass- XPATH and LDAP

- XPATH parsing standard error
- XPATH is method available for XML parsing
- MS SQL server provides interface and one can get table content in XML format.
- Once this is fetched one can run XPATH queries and obtain results.
- What if username/password parsing done on using XPATH – XPATH injection

XPATH injection

```java
string credential = "//users[@username=""+user+"" and @password=""+pass+""];
```

- XPATH parsing can be leveraged by passing following string ' or 1=1 or '='
- This will always true on the first node and user can get access as who ever is first user.
Bingo!
A5. Business Logic Bypass

- Business logic would be part of client side
- Looking into JavaScript functions
- Looking for calls
- Looking for sources and trust
- Debugging JavaScript
- Leads to potential hold in the logic

A6. Decompilation Attack

- SWF decompiler
- Analyzing action scripts and other files
- Tools can help in fetching information
- Identifying back-end calls
- AMF stream and fuzzing them
- Silverlight de-compilation on similar lines
- Lot of analysis and findings on these lines
A7. WSDL Scanning

- WSDL discovery – it is possible to discover WSDL file
- In some cases it may have some internal API calls
- These APIs can be abused and invoked without going through the process
- Application hacking and attacks over SOAP/WSDL – possible!

A8. XSS with Flash

- XSS with flash
- Usage of calls
- Global access
- “asfunction” calls and methods
- Cross site flashing
- Exploiting redirecting methods
A9. CSRF with XML

• CSRF with XML stream is possible
• If stream is not validated then it can cause this particular attack
• Libraries and implementations are not checking content-type in some cases
• It is possible to craft HTML page to originate XML streams
• Lethal attack and can cause damage to end-user

A10 – Widget/Mashup

• RSS feeds
• Mashups
• Widgets

• Possible to perform XSS or Clickjacking.
• Inter-widget spying is also reality
• Reverse engineering can help in identifying it.
Defense

• One needs to do Code Analysis and Whitebox testing to identify some of these vulnerabilities.
• DOM based calls and dynamic content injection – incoming content validations.
• WAF will not protect JSON/AMF/XML streams etc.
• Cross Domain Calls and Security.

Conclusion – Questions?