

Top 10 Web 2.0 Attacks

Shreeraj Shah



Blueinfy

© Blueinfy Solutions



Who Am I?

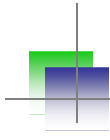
<http://shreeraj.blogspot.com>
shreeraj@blueinfy.com
<http://www.blueinfy.com>

- **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

Blueinfy Securityexposure
Strategic Security Solutions



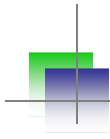
© Blueinfy Solutions



Attack Surface for Web 2.0

Blueinfy

© Blueinfy Solutions



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 ...

© Blueinfy Solutions



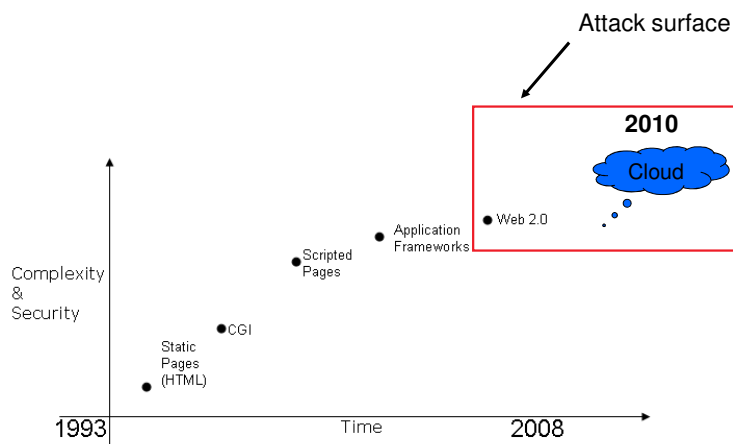
Technology Trends

- 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

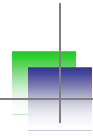
© Brainfy Solutions



Past, Present and Future



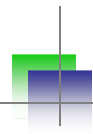
© Brainfy Solutions



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

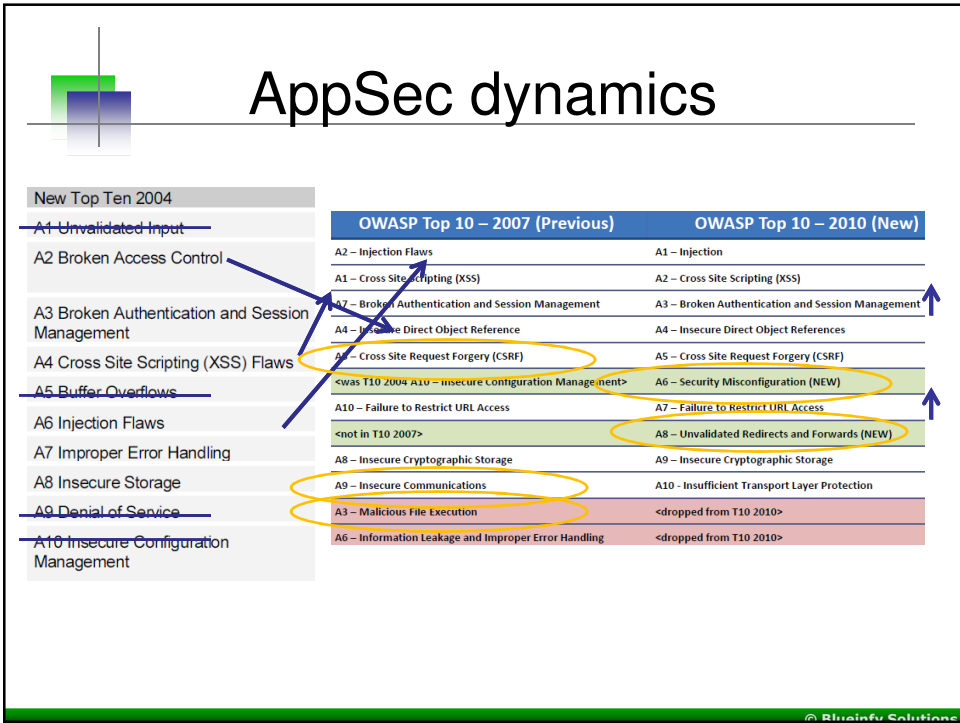
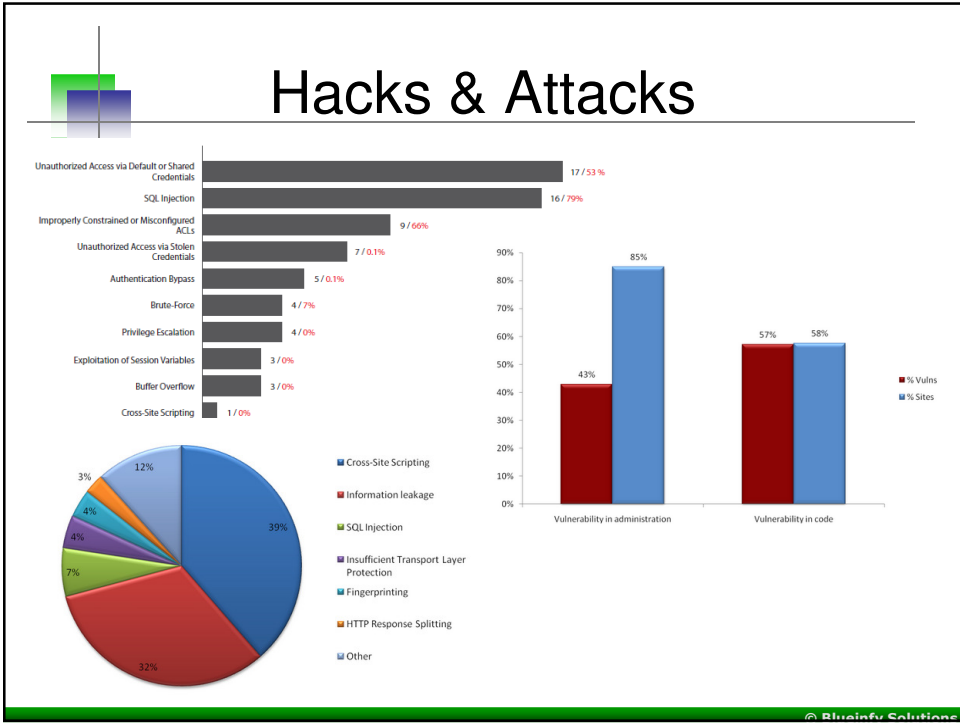
© Brainfy Solutions



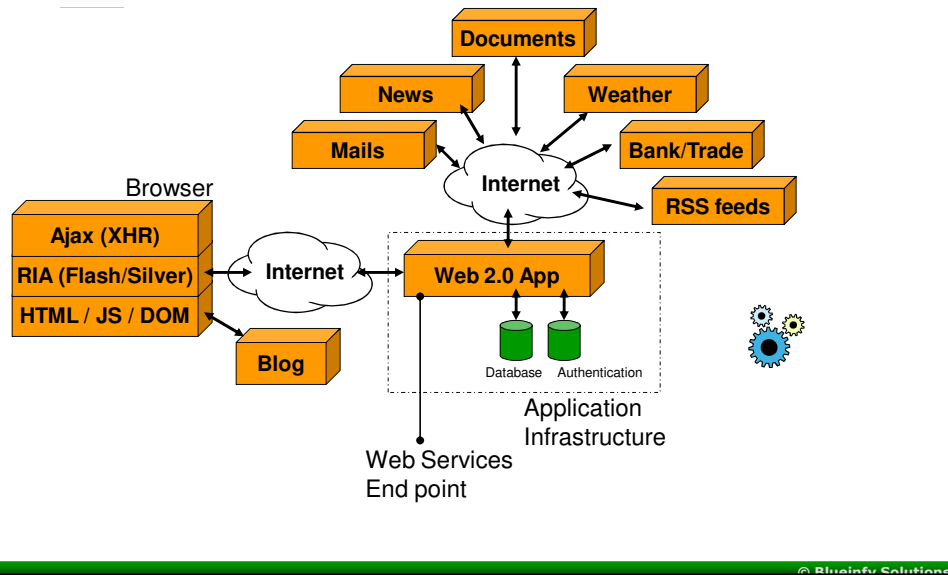
Web 2.0 Attacks

- **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

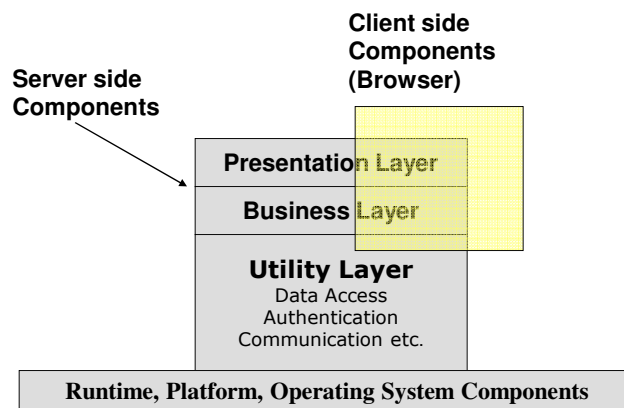
© Brainfy Solutions



Next Generation Architecture



Breaking traditional layers





Threat Model for 2.0

Changing dimension	Traditional	Web 2.0/RIA
Entry points	Structured	Scattered and multiple
Dependencies	Limited	<ul style="list-style-type: none">• Multiple technologies• Information sources• Protocols
Vulnerabilities	Server side [Typical injections]	<ul style="list-style-type: none">• Web services [Payloads]• Client side [XSS & XSRF]
Exploitation	Server side exploitation	Both server and client side exploitation

© Bluewinf Solutions



Top 10 Attacks

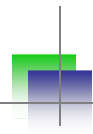
© Bluewinf Solutions



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

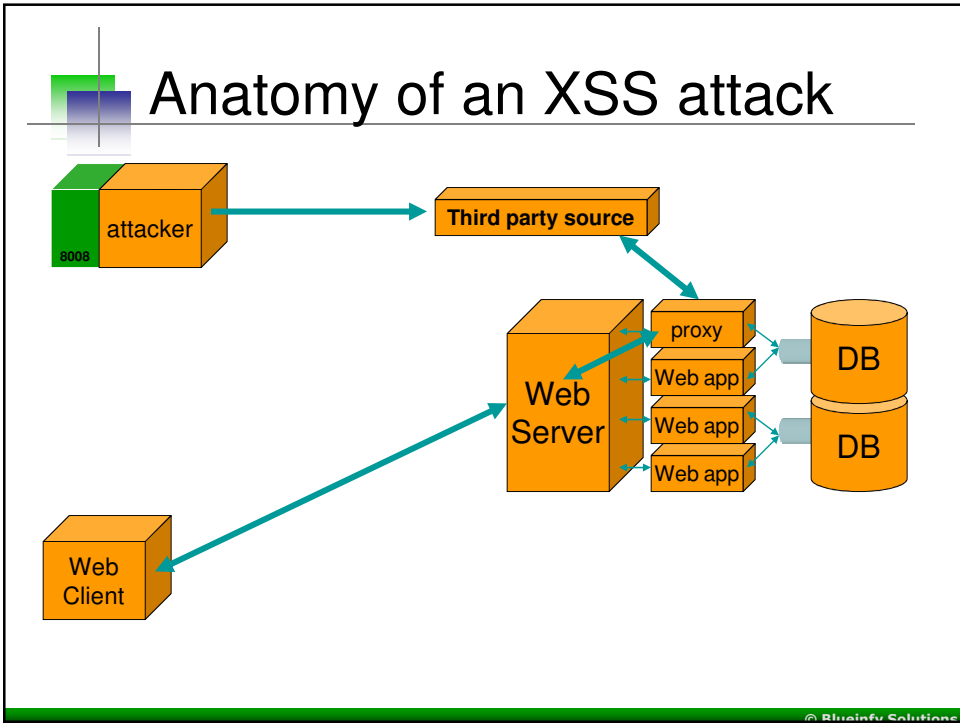
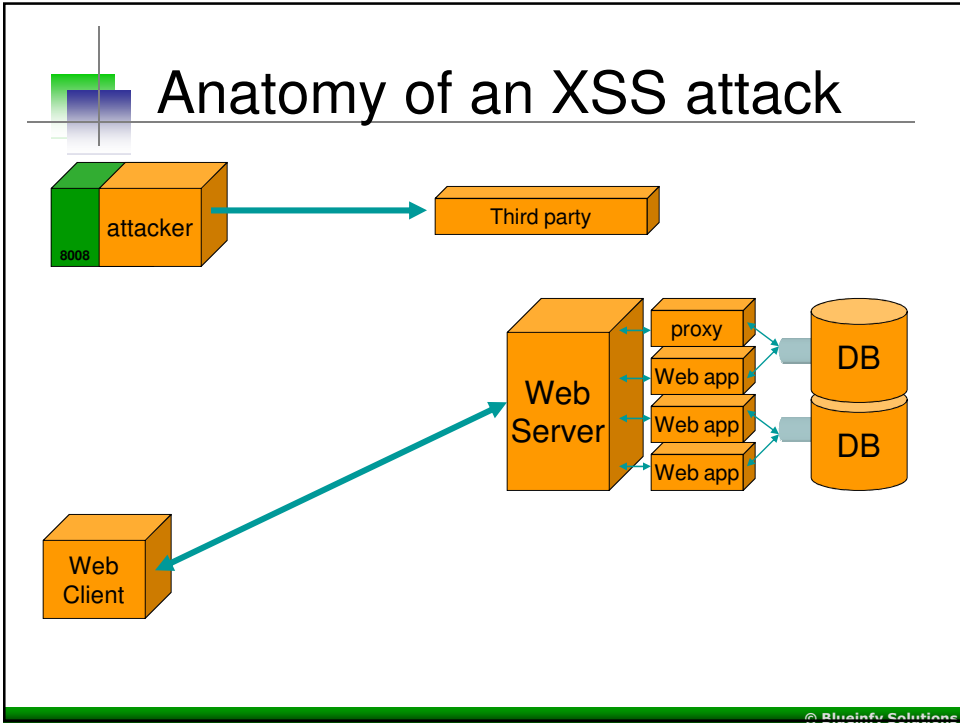
© Brainfy Solutions



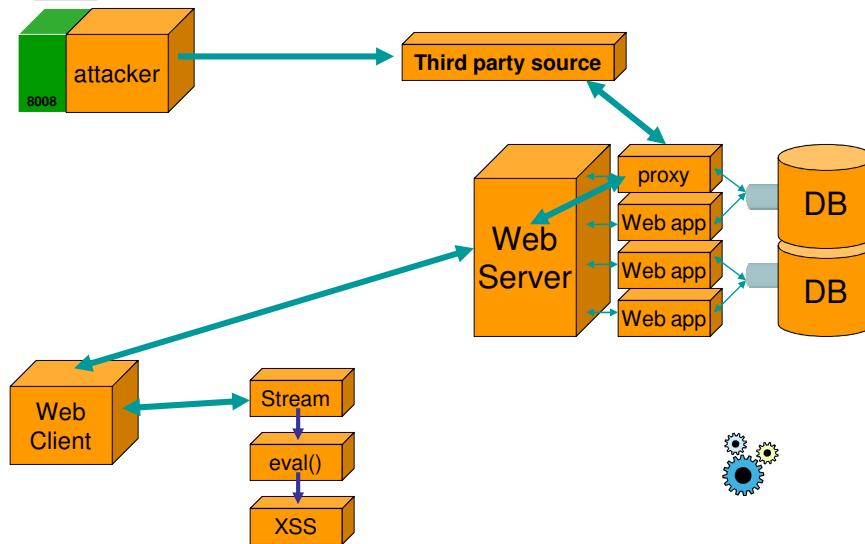
A1 – XSS with DOM

- 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

© Brainfy Solutions



Anatomy of an XSS attack

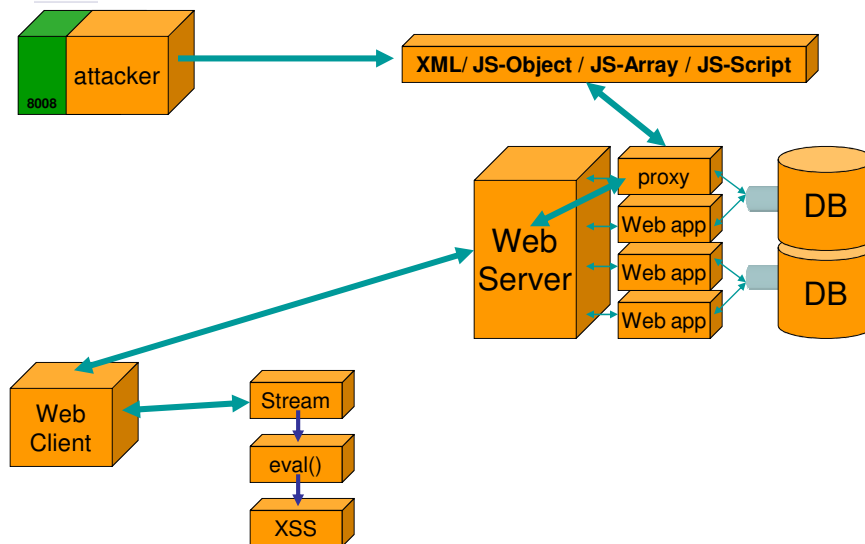


DOM based XSS

```
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



© Blueinfo Solutions

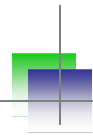


DOM based XSS



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



© Blueinfo 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 

© Brainfy Solutions



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 

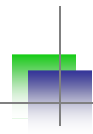
© Brainfy Solutions



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

© Brainfy Solutions



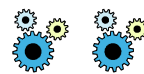
XPATH injection

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!



© Brainfy Solutions

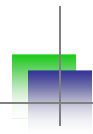


A5. Business Logic Bypass




- 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



© Brainfy Solutions





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

© Brainfy Solutions






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! 

© Brainfy Solutions



A8. XSS with Flash

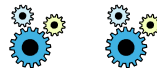
- XSS with flash 
- Usage of calls 
- Global access
- “asfunction” calls and methods
- Cross site flashing 
- Exploiting redirecting methods

© Brainfy Solutions



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



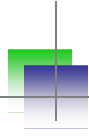
© Brainfy Solutions



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.

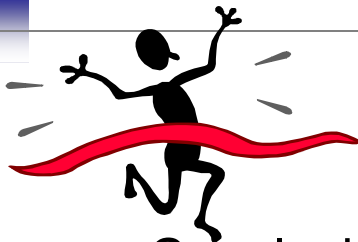
© Brainfy Solutions



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.

© Brainfy Solutions



Conclusion – Questions?

© Brainfy Solutions