Web 2.0 Hacking
Defending Ajax & Web Services

Shreeraj Shah

Dubai, HITB 2007
5th April 2007

Who am I?

• Founder & Director
  - Net Square (Brief)

• Past experience
  - Chase, IBM & Foundstone

• Interest
  - Web security research

• Published research
  - Articles / Papers – Securityfocus, O’erilly, DevX, InformIT etc.
  - Tools – wsChess, MSNPawn, Ajaxfinger, Scanajax
  - Advisories - Net, Java servers etc.

• Books (Author)
  - Hacking Web Services (Thomson 2006)
  - Web Hacking (AWL 2003)

Agenda

• Web 2.0
  - Industry
  - Technologies
  - Security
  - Impact of Web 2.0

• Ajax
  - Basics
  - Attacks
  - Methodology
  - Fingerprinting
  - Enumeration
  - Crawling
  - Scanning
  - Vulnerabilities
  - Securing Ajax

• Web Services
  - Basics
  - Methodologies
  - Security
  - Assessment
  - Discovery
  - Enumeration
  - Attack vectors
  - Defense
Agenda

- Web 2.0
  - Industry
  - Technologies
  - Security
- Impact of Web 2.0
- Ajax
  - Basics
  - Attacks
  - Methodology
  - Fingerprinting
  - Enumeration
  - Scanning
  - Vulnerabilities
  - Securing Ajax

- Web Services
  - Basics
  - Methodologies
  - Security
  - Assessment
  - Footprinting
  - Discovery
  - Enumeration
  - Attack vectors
  - Defense

Industry - Web 2.0

Agenda

- Web 2.0
  - Industry
  - Technologies
  - Security
- Impact of Web 2.0
- Ajax
  - Basics
  - Attacks
  - Methodology
  - Fingerprinting
  - Enumeration
  - Scanning
  - Vulnerabilities
  - Securing Ajax

- Web Services
  - Basics
  - Methodologies
  - Security
  - Assessment
  - Footprinting
  - Discovery
  - Enumeration
  - Attack vectors
  - Defense
Industry

- Web Services is forming back end and accessible on XML protocols
- AJAX – empowering browsers
- XML based services
- Rich Internet Applications are consuming back end web services
- Search engines and mechanisms for web services publishing are getting momentum

---

Industry

- **2007.** Web services would rocket from $1.6 billion in 2004 to $34 billion. [IDC]
- **2008.** Web Services or Service-Oriented Architecture (SOA) would surge ahead. [Gartner]
- Web 2.0 and Enterprise 2.0 are on its way to redefine application layer

---

Agenda

- Web 2.0
  - Industry
  - Technologies
  - Security
  - Impact of Web 2.0
- Ajax
  - Basics
  - Attacks
  - Methodology
  - Fingerprinting
  - Enumeration
  - Crawling
  - Scanning
  - Vulnerabilities
  - Securing Ajax

- Web Services
  - Basics
  - Methodologies
  - Security
  - Assessment
    - Footprinting
    - Discovery
    - Enumeration
    - Attack vectors
  - Defense
Web 2.0 Architecture

Web 2.0 Components

Technologies
Agenda

- Web 2.0
  - Industry
  - Technologies
  - Security
- Impact of Web 2.0
- Ajax
  - Basics
  - Attacks
  - Methodology
  - Fingerprinting
  - Enumeration
  - Crawling
  - Scanning
  - Vulnerabilities
  - Securing Ajax
- Web Services
  - Basics
  - Methodologies
  - Security
- Assessment
  - Footprinting
  - Discovery
  - Enumeration
  - Attack vectors
- Defense

Web 2.0 Security

- Complex architecture and confusion with technologies
- Web 2.0 worms and viruses – Sammy, Yammaner & Spaceflash
- Ajax and JavaScripts – Client side attacks are on the rise
- Web Services attacks and exploitation
- Flash clients are running with risks

Web 2.0 Security

- Mashup and un-trusted sources
- RSS feeds manipulation and its integration
- Single Sign On and information convergence at one point
- Widgets and third-party components are bringing security concerns
- Old attacks with new carriers
Stats ’06: Vulnerabilities

• 0.4% critical
  - could be used to form a prolific automated worm
• 16.6% high
  - could be exploited to gain control of the host
• 63% medium
  - could be used to access files/escalate privileges
• 20% low
  - vulnerabilities that leak information
  - allow a denial-of-service attack

Source: Network World

Stats ’06: Vulnerabilities

• cross-site scripting (14.5%)
• SQL injection (10.9%)
• buffer overflows (10.8%)
• web directory path traversal (3%)

Source: Network World

Web App Layer Attacks

• 95% companies hacked from web apps
  - FBI / CSI
• Most popular attacks against Web servers
  - incidents.org
• 3 out of 4 web sites vulnerable to attack
  - Gartner
Causes!

- Increase in toolkits and exploits
- Too many protocols causing confusion
- Race for deployment – poor implementation
- New technologies mean new attack points in application frameworks

Agenda

- Web 2.0
  - Industry
  - Technologies
  - Security
  - Impact of Web 2.0
- Ajax
  - Basics
  - Attacks
  - Methodology
  - Fingerprinting
  - Enumeration
  - Crawling
  - Scanning
  - Vulnerabilities
  - Securing Ajax
- Web Services
  - Basics
  - Methodologies
  - Security
  - Assessment
  - Footprinting
  - Discovery
  - Enumeration
  - Attack vectors
- Defense

Impact of Web 2.0

- Impact of Web 2.0 is on 4 dimensions
  - Application Infrastructure
  - Security threats
  - Methodology
  - Countermeasure

CSI Security Survey: Vulnerability Distribution

64% programming errors
66% misconfiguration, other problems
## Impact of Web 2.0

### Application Infrastructure

<table>
<thead>
<tr>
<th>Changing dimension</th>
<th>Web 1.0</th>
<th>Web 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>(AI1) Protocols</td>
<td>HTTP &amp; HTTPS</td>
<td>SAML, WS-FED, SAML over HTTP &amp; HTTPS</td>
</tr>
<tr>
<td>(AI2) Information structures</td>
<td>HTML transfer</td>
<td>XML, JSON, JS Objects etc.</td>
</tr>
<tr>
<td>(AI3) Communication methods</td>
<td>Anonymous, Pushback, Session and Redirect</td>
<td>Asynchronous &amp; Cross-domain (proxy)</td>
</tr>
<tr>
<td>(AI4) Information sharing</td>
<td>Single place information</td>
<td>Multiple sources (Urge for integrated information platform)</td>
</tr>
</tbody>
</table>

### Security Threats

<table>
<thead>
<tr>
<th>Changing dimension</th>
<th>Web 1.0</th>
<th>Web 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>(T1) Entry points</td>
<td>Structured</td>
<td>Scattered and multiple</td>
</tr>
<tr>
<td>(T2) Dependencies</td>
<td>Limited</td>
<td>Multiple technologies, Information sources, Protocols</td>
</tr>
<tr>
<td>(T3) Vulnerabilities</td>
<td>Server-side (Typical injection)</td>
<td>Web services (Payloads), Client-side (XSS &amp; XSRF)</td>
</tr>
<tr>
<td>(T4) Exploitation</td>
<td>Server-side exploitation</td>
<td>Both server and client-side exploitation</td>
</tr>
</tbody>
</table>

### Methodology

<table>
<thead>
<tr>
<th>Changing dimension</th>
<th>Web 1.0</th>
<th>Web 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Footprinting</td>
<td>Typical with &quot;Host&quot; and DNS</td>
<td>Empowered with search</td>
</tr>
<tr>
<td>Discovery</td>
<td>Simple</td>
<td>Difficult with hidden calls</td>
</tr>
<tr>
<td>Enumeration</td>
<td>Structured</td>
<td>Several streams</td>
</tr>
<tr>
<td>Scanning</td>
<td>Structured and simple</td>
<td>Difficult with extensive Ajax</td>
</tr>
<tr>
<td>Automated attacks</td>
<td>Easy after discovery</td>
<td>Difficult with Ajax and web services</td>
</tr>
<tr>
<td>Reverse engineering</td>
<td>On the server side (Default)</td>
<td>Client-side with Ajax &amp; Flash</td>
</tr>
<tr>
<td>Code reviews</td>
<td>Focus on server-side only</td>
<td>Client-side analysis needed</td>
</tr>
</tbody>
</table>
Impact of Web 2.0

- Countermeasure

<table>
<thead>
<tr>
<th>Changing dimension</th>
<th>Web 1.0</th>
<th>Web 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Owner of information</td>
<td>Single place</td>
<td>Multiple places (Mashups &amp; RSS)</td>
</tr>
<tr>
<td>Browser security</td>
<td>Simple DOM usage</td>
<td>Complex DOM usage</td>
</tr>
<tr>
<td>Validations</td>
<td>Server side</td>
<td>Client side (incoming content)</td>
</tr>
<tr>
<td>Logic shift</td>
<td>Only on server</td>
<td>Client side shift</td>
</tr>
<tr>
<td>Secure coding</td>
<td>Structured and single place</td>
<td>Multiple places and scattered</td>
</tr>
</tbody>
</table>

Agenda

- Web 2.0
  - Industry
  - Technologies
  - Security
  - Impact of Web 2.0
  - Ajax
    - Basics
    - Attacks
    - Methodology
      - Fingerprinting
      - Enumeration
      - Scanning
    - Vulnerabilities
    - Securing Ajax
- Web Services
  - Basics
  - Methodologies
  - Security
- Assessment
  - Footprinting
  - Discovery
  - Enumeration
  - Attack vectors
- Defense

Ajax basics

- Asynchronous JavaScript and XML
Ajax - Sample

```javascript
function loadhtml()
{
    var http;
    if(window.XMLHttpRequest)
    {
        http = new XMLHttpRequest();
    }
    else if (window.ActiveXObject)
    {
        http=new ActiveXObject("Msxml2.XMLHTTP");
    }
    http.open("GET", "main.html", true);
    http.onreadystatechange = function()
    {
        if (http.readyState == 4) {
            var response = http.responseText;
            document.getElementById('main').innerHTML = response;
        }
    }
    http.send(null);
}

Agenda

- Web 2.0
  - Industry
  - Technologies
  - Security
- Impact of Web 2.0
- Ajax
  - Basics
  - Attacks
  - Methodology
    - Fingerprinting
    - Enumeration
    - Scanning
  - Vulnerabilities
  - Securing Ajax
- Web Services
  - Basics
  - Methodologies
  - Security
  - Assessment
    - Footprinting
    - Discovery
    - Enumeration
    - Attack vectors
  - Defense

Ajax attack points

- Ajax components & Widgets
- Cross domain vulnerable browsers and callback implementations
- DOM manipulation calls and points
- Insecure eval()
- HTML tags
- Intranet nodes and internal resources
Ajax attack vectors

- Entry point scanning and enumeration
- Cross site scripting (XSS) attacks
- Cross site Request Forgery (CSRF) issues
- Client side code reverse engineering
- Security control and validation bypassing
- Local privacy information enumeration
- Ajax framework exploitation – known bugs

Agenda

- Web 2.0
  - Industry
  - Technologies
  - Security
- Impact of Web 2.0
- Ajax
  - Basics
  - Attacks
  - Methodology
    - Fingerprinting
    - Enumeration
    - Scanning
  - Vulnerabilities
  - Securing Ajax
- Web Services
  - Basics
  - Methodologies
  - Security
  - Assessment
  - Footprinting
  - Discovery
  - Enumeration
  - Attack vectors
  - Defense

Ajax fingerprinting

- Determining Ajax calls
- Framework fingerprinting
- Running with what?
  - Atlas
  - GWT
  - Etc.
- Ajaxfinger a tool to achieve this
- Can help in assessment process
Ajax enumeration

- Identifying XHR calls
- Decoding the back end calls
- Information enumeration on structures
  - JSON
  - XML
  - JS-Objects etc.
- Tools to determine Ajax calls
- Valuable information – Crawlers can’t get it because hidden in JavaScript

Ajax Crawling

- Crawling Ajax driven app – a challenge
- Resources are hidden in JavaScript
- Simple scanner will fail
- Crawling with actual DOM context
- Automated crawling with browser is required
- How?

Ajax Scanning

- Scanning Ajax components
- Retrieving all JS include files
  - Part of `<SCRIPT SRC=.....>`
- Identifying XHR calls
- Grabbing function
- Mapping function to DOM event
- Scanning code for XSS – look for `eval()` and `document.write()`
## Agenda

- **Web 2.0**
  - Industry
  - Technologies
  - Security
- **Impact of Web 2.0**
- **Ajax**
  - Basics
  - Attacks
  - Methodology
  - Fingerprinting
  - Enumeration
  - Scanning
  - Vulnerabilities
  - Securing Ajax
- **Web Services**
  - Basics
  - Methodologies
  - Security
  - Assessment
  - Footprinting
  - Discovery
  - Enumeration
  - Attack vectors
  - Defense

## Ajax serialization issues

- Ajax processing various information coming from server and third party sources. – XSS opportunities

```javascript
message = {
  from : "john@example.com",
  to : "jerry@victim.com",
  subject : "I am fine",
  body : "Long message here",
  showsubject : function(){document.write(this.subject)}
};
```

### XSS

## Ajax serialization issues

- JSON issues

```javascript
{"bookmarks": [{"link": "http://www.example.com"}, {"link": "Interesting link!"]}
```

- JS – Array manipulation

```javascript
new Array("Laptop", "Thinkpad", "T60", "Negro", "T500", "It is great and I have used it for 2 years")
```
Ajax and JS manipulation

- JavaScript exploitation – XSS
- Identifying DOM points like `document.write()`
- `Eval()` – another interesting point
- Attack APIs and tools for exploitation
- Lot can be done by an attacker from session hijacking to key loggers

Ajax and RSS injection

- RSS feeds are another entry point to the browser
- Injecting script to the RSS feeds and Ajax call may execute it.
- One click – Malformed linked injected into it and can lead to exploit “javascript:”
- Leveraging events – onClick, onMouse etc.

Cross-domain calls

- Browser security doesn’t support cross domain calls
- But cross domain callback with JavaScript is possible
- This can be lethal attack since cross domain information get executed on the current DOM context.
Agenda

- Web 2.0
  - Industry
  - Technologies
  - Security
- Impact of Web 2.0
- Ajax
  - Basics
  - Attacks
  - Methodology
  - Fingerprinting
  - Enumeration
  - Scanning
- Vulnerabilities
  - Securing Ajax

- Web Services
  - Basics
  - Methodologies
  - Security
  - Assessment
  - Footprinting
  - Discovery
  - Enumeration
  - Attack vectors
  - Defense

Defending Ajax

- No business logic information on client side.
- Do not trust third party source – filter it out
- No direct cross domain call back
- Filtering at browser level before processing information
- Avoiding client side validation

Defending Ajax

- No secret in Ajax calls
- Proper data structure selection and frameworks
- Avoid client side validation
- Securing client side calls like eval() and document.write()
- HTML tags filtering before serving to end client
Agenda

- Web 2.0
  - Industry
  - Technologies
  - Security
- Impact of Web 2.0
- Ajax
  - Basics
  - Attacks
  - Methodology
    - Fingerprinting
    - Enumeration
    - Crawling
    - Scanning
  - Vulnerabilities
  - Securing Ajax

- Web Services
  - Basics
  - Methodologies
  - Security
  - Assessment
    - Footprinting
    - Discovery
  - Enumeration
  - Attack vectors
  - Defense

Web services stack

- Presentation Stack
  - XML
- Security Stack
  - WS-Security
- Discovery Stack
  - UDDI, DISCO
- Access Stack
  - WSDL, SOAP
- Transport Stack
  - HTTP, HTTPS

Security!

- End Client
- In Transit
- Vendor Controlled
- User Controlled

Web Services Client
- HTTP POST
- SOAP Envelope

Web Services Engine
- Web Services Code & Components

Web Services Binsaries
- Deployment
- Shale
Assessment strategies

Risk - In transit

- In transit Sniffing or Spoofing
- WS-Routing security concern
- Replay attacks

Risk - Web services Engine

- Buffer overflow
- XML parsing attacks
- Spoiling Schema
- Complex or Recursive structure as payload
- Denial of services
- Large payload
Web services Deployment - Risk

- Fault code leaks
- Permissions & Access issues
- Poor policies
- Customized error leakage
- Authentication and Certification

Web services User code - Risk

- Parameter tampering
- WSDL probing
- SQL/LDAP/XPATH/OS command injection
- Virus/Spyware/Malware injection
- Brute force
- Data type mismatch
- Content spoofing
- Session tampering
- Format string
- Information leakage
- Authorization

Agenda

- Web 2.0
  - Industry
  - Technologies
  - Security
  - Impact of Web 2.0
- Ajax
  - Basics
  - Attacks
  - Methodology
  - Fingerprinting
  - Enumeration
  - Crawling
  - Scanning
  - Vulnerabilities
  - Securing Ajax
- Web Services
  - Basics
  - Methodologies
  - Security
  - Assessment
  - Footprinting
  - Discovery
  - Enumeration
  - Attack vectors
  - Defense
Footprinting

- Objectives
  - Place for web services...
  - We may know the company name in this case?
  - Do we have any whois for web services?
  - If we answer above questions then we can have enough information on what to assess?
UDDI

- *Universal Description, Discovery, and Integration (UDDI)*
- It acts as White/Yellow/Green pages
- Xmethods etc…
- Information can be published and retrieved from
- Gets replicated across networks over internet

UDDI

- It includes
  - businessEntity
  - businessService
  - bindingTemplate
  - tModel

UDDI

- Find UDDI APIs
  - businessEntity Structure
  - businessService Structure
  - bindingTemplate Structure
  - tModel Structure
Web Service Discovery
- After footprinting web services next step is to perform discovery.
- On the basis of services found one can do so.
- Finding access point for web services will point to its discovery.
- Discovery is the key to the kingdom.
- Once again over UDDI.

Agenda
- Web 2.0
  - Industry
  - Technologies
  - Security
  - Impact of Web 2.0
- Ajax
  - Basics
  - Attacks
  - Methodology
  - Fingerprinting
  - Enumeration
  - Scanning
  - Vulnerabilities
  - Securing Ajax
- Web Services
  - Basics
  - Methodologies
  - Security
  - Assessment
  - Footprinting
  - Discovery
  - Enumeration
  - Attack vectors
  - Defense

Web Service Discovery
- From various keys – Service and Business one can dig access point from UBN.
- This is a part of protocol and identified from XML block itself.
Web Service Search

- Search in public domain
- Use – Search Engines
- Google & MSN – An excellent tool
- Look for wsdl,asmx,jws etc.
- Filetype and allinurl are best friends
- Leveraging Web APIs

Agenda

- Web 2.0
  - Industry
  - Technologies
  - Security
  - Impact of Web 2.0
- Ajax
  - Basics
  - Attacks
  - Methodology
  - Fingerprinting
  - Enumeration
  - Crawling
  - Scanning
  - Vulnerabilities
  - Securing Ajax

- Web Services
  - Basics
  - Methodologies
  - Security
  - Assessment
  - Footprinting
  - Discovery
  - Enumeration
  - Attack vectors
  - Defense

Technology Identification

- Running on which platform?
- Configuration and Structures
- File extensions
- Path discovery
- This is very useful information
HITB 2007 © Shreeraj Shit

**Demo Application**

![Demo Application Image]

Web Services Location of WSDL

**Technology Identification**

- Location can be obtained from UDDI as well if already published.
- WSDL location [Access Point]
  
  \[ \text{http://192.168.11.2/ws/dvds4less.asmx?wsdl} \]

  .asmx – indicates .Net server from MS

**Technology Identification**

- Similarly .jws – for Java web services
- /ws/ - in the path indicates web services
- MS-SOAPToolkit can be identified as well

```plaintext
C:\>nc 192.168.11.2 80
HEAD / HTTP/1.0
HTTP/1.1 200 OK
Server: Microsoft-IIS/5.0
Date: Tue, 28 Sep 2004 18:48:20 GMT
X-Powered-By: ASP.NET
Connection: Keep-Alive
Content-Length: 7565
Content-Type: text/html
Set-Cookie: ASPSESSIONIDSSRQDRC=LMMPKHNAAOFDFMHD0OJHC0; path=/
Cache-control: private
```
**Technology Identification**

- Resource header spits some information as well

```
C:\>nc 192.168.11.2 80
HEAD /ws/dvds4less.asmx HTTP/1.0
HTTP/1.1 500 Internal Server Error
Server: Microsoft-IIS/5.0
Date: Tue, 28 Sep 2004 18:50:09 GMT
X-Powered-By: ASP.NET
X-AspNet-Version: 1.1.4322
Cache-Control: private
Content-Type: text/html; charset=utf-8
Content-Length: 3026
```

**WSDL Scanning/Enumeration**

- What is WSDL?
- What information one can enumerate from WSDL?
- WSDL exposure is threat or not?

**WSDL**

- WSDL is web services definition language
- It is similar to old IDL for remote calls used in CORBA or other remote invoke methods.
- It contains detail of methods
- Types of I/O
- Parameters of methods
- It is XML document with standards.
Nodes of WSDL

- Data types
- Message Types
- Operations

**WSDL <Service>**

```xml
<wSDL>
<service name="dvds4less">
  <port name="dvds4lessSoap" binding="s0:dvds4lessSoap">
    <soap:address location="http://192.168.11.2/ws/dvds4less.asmx"/>
  </port>
</service>
```

Where the call is going to hit?
It is where service is listening.

**WSDL <portType>**

```xml
<portType name="dvds4lessSoap">
  <operation name="Intro">
    <input message="s0:IntroSoapIn"/>
    <output message="s0:IntroSoapOut"/>
  </operation>
  <operation name="getProductInfo">
    <input message="s0:getProductInfoSoapIn"/>
    <output message="s0:getProductInfoSoapOut"/>
  </operation>
  <operation name="getRebatesInfo">
    <input message="s0:getRebatesInfoSoapIn"/>
    <output message="s0:getRebatesInfoSoapOut"/>
  </operation>
</portType>
```

Methods one can call.
WSDL <Message>

```xml
<portType name="dvds4lessSoap">
    <operation name="getProductInfo">
        <input message="s0:getProductInfoSoapIn"/>
        <output message="s0:getProductInfoSoapOut"/>
    </operation>
</portType>

<message name="getProductInfoSoapIn">
    <part name="parameters" element="s0:getProductInfo"/>
</message>

<message name="getProductInfoSoapOut">
    <part name="parameters" element="s0:getProductInfoResponse"/>
</message>
```

WSDL <Types>

```xml
<s:element name="getProductInfo">
    <s:complexType>
        <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="id" type="s:string"/>
        </s:sequence>
    </s:complexType>
</s:element>

<s:element name="getProductInfoResponse">
    <s:complexType>
        <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="getProductInfoResult" type="s:string"/>
        </s:sequence>
    </s:complexType>
</s:element>
```

WSDL Profile after Scan

<table>
<thead>
<tr>
<th>Methods</th>
<th>INPUT</th>
<th>OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intro</td>
<td>-No-</td>
<td>String</td>
</tr>
<tr>
<td>getProductInfo</td>
<td>String</td>
<td>String</td>
</tr>
<tr>
<td>getRebatesInfo</td>
<td>String</td>
<td>String</td>
</tr>
</tbody>
</table>
How it looks?

WSDL
- PortType
- Service
- Message
- Types

Web Services
Code
OR
Class

Remote Invokes

Intro

getProductInfo

getProductRebateInfo

Agenda

- Web 2.0
  - Industry
  - Technologies
  - Security
- Impact of Web 2.0
- Ajax
  - Basics
  - Attacks
  - Methodology
    - Fingerprinting
    - Enumeration
    - Scanning
- Vulnerabilities
- Securing Ajax

- Web Services
  - Basics
  - Methodologies
  - Security
  - Assessment
    - Footprinting
    - Discovery
    - Enumeration
    - Attack vectors
    - Defense

AV 1 - XML poisoning

- XML node manipulation
- Attack on parsing logic
  - SAX
  - DOM
- Can be lethal – DoS or breaking execution logic
XML poisoning

```xml
<CustomerRecord>
  <CustomerNumber>289001</CustomerNumber>
  <FirstName>John</FirstName>
  <LastName>Smith</LastName>
  <Address>Apt 31, 1st Street</Address>
  <Email>john@smith.com</Email>
  <PhoneNumber>3809922347</PhoneNumber>
</CustomerRecord>
```

XML poisoning

```xml
<CustomerRecord>
  <CustomerNumber>289001</CustomerNumber>
  <FirstName>John</FirstName><CustomerNumber>289001</CustomerNumber>
  <FirstName>John</FirstName><CustomerNumber>289001</CustomerNumber>
  <LastName>Smith</LastName>
  <Address>Apt 31, 1st Street</Address>
  <Email>john@smith.com</Email>
  <PhoneNumber>3809922347</PhoneNumber>
</CustomerRecord>
```

XML poisoning

```xml
<CustomerRecord>
  <CustomerNumber>289001</CustomerNumber>
  <FirstName>John</FirstName>
  <FirstName>John</FirstName>
  ... 100 time...
  <LastName>Smith</LastName>
  <Address>Apt 31, 1st Street</Address>
  <Email>john@smith.com</Email>
  <PhoneNumber>3809922347</PhoneNumber>
</CustomerRecord>
```
**AV 2 - Parameter tampering & Fault code leakage**

- Fault code of web services spit lot of information about internal workings.
- This attack can fetch internal paths, database interfaces etc.
- Fault code is part of SOAP envelope and this helps an attacker to make logical deduction about assets.

**SOAP request**

```xml
<?xml version="1.0" encoding="utf-16"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<soap:Body>
<getRebatesInfo xmlns="http://tempuri.org/">
<fileinfo>abx.xyz</fileinfo>
</getRebatesInfo>
</soap:Body>
</soap:Envelope>
```

**SOAP response**

```xml
<?xml version="1.0" encoding="utf-16"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<soap:Body>
<soap:Fault>
<faultcode>soap:Server</faultcode>
<faultstring>Server was unable to process request. --&gt; Could not find file
"c:\inetpub\wwwroot\rebates\abx.xyz";</faultstring>
<detail/>
</soap:Fault>
</soap:Body>
</soap:Envelope>
```
AV 3 - SQL injection

- SQL injection can be done using SOAP traffic.
- It is an innovative way of identifying database interface points.
- One can leverage `xp_cmdshell` via SOAP.
- Back end database can be compromised using this attack.

SOAP request

```xml
<?xml version="1.0" encoding="utf-16"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <soap:Body>
        <getProductInfo xmlns="http://tempuri.org/">
            <id>1</id>
        </getProductInfo>
    </soap:Body>
</soap:Envelope>
```
<?xml version="1.0" encoding="utf-16"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<soap:Body>
<soap:Fault>
<faultcode>soap:Server</faultcode>
<faultstring>Server was unable to process request. --&gt; Cannot use empty object or column names. Use a single space if necessary. &lt;/faultstring&gt;
<detail>
</detail>
</soap:Fault>
</soap:Body>
</soap:Envelope>

<?xml version="1.0" encoding="utf-16"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<soap:Body>
<getProductInfo xmlns="http://tempuri.org/">
<id>1 or 1=1</id>
</getProductInfo>
</soap:Body>
</soap:Envelope>

<?xml version="1.0" encoding="utf-16"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<soap:Body>
<getProductInfoResponse xmlns="http://tempuri.org/">
<getProductInfoResult>
/(1)Finding Nemo($14.99)/
/(2)Bend it like Beckham($12.99)/
/(3)Doctor Zhivago($10.99)/
/(4)A Bug's Life($13.99)/
/(5)Lagaan($12.99)/
/(6)Monsoon Wedding($10.99)/
/(7)Lawrence of Arabia($14.99)/
</getProductInfoResult>
</getProductInfoResponse>
</soap:Body>
</soap:Envelope>

<?xml version="1.0" encoding="utf-16"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<soap:Body>
<getProductInfoResponse xmlns="http://tempuri.org/">
<getProductInfoResult>Entire Table is out</getProductInfoResult>
</getProductInfoResponse>
</soap:Body>
</soap:Envelope>
SOAP response

Exploiting this Vulnerability

Exploit code

SOAP request

Works!

Looks Normal response

SOAP request

But … Code got executed

Got Admin via cmdshell
AV 4 – XPATH injection

- XPATH is a new way of querying XML documents.
- This attack works nicely on web services since they use XML extensively.
- Developer’s loophole can be leveraged with an exploit.
- XPATH query crafting is next generation attack methods.

XPATH Injection - Basics

- XPATH is a language defined to find information from XML document.
- As XPATH name suggests it indeed uses path to traverse through nodes of XML document and look for specific information from the document.
- XPATH provides expressions like slash (/), double slash (//), dot (.), double dot (..), @, =, <, > etc. It helps in traversing through XML document.

XPATH – Vulnerable Code

```csharp
string fulltext = "";
string coString = "Provider=SQLOLEDB;Server=(local);database=order;User ID=sa;Password=mypass";
SqXmlCommand co = new SqXmlCommand(coString);
co.RootTag="Credential";
co.CommandType = SqXmlCommandType.Sql;
co.CommandText = "SELECT * FROM users for xml Auto";
XmlReader xr = co.ExecuteXmlReader();
xr.MoveToContent();
fulltext = xr.ReadOuterXml();
XmlNodeList xmln = doc.SelectNodes(credential);
string temp;
if(xmln.Count > 0)
{
  //True
}
else //false
```
Attacking XPATH point

• \(//\text{users[@username}='\text{'+user+'\text{'} and @password}='\text{'+pass+'\text{'}]}\);
• 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.
• \(//\text{users[@username}='\text{’ or 1=1 or ‘=}’ and @password}='\text{any}’\)
Bingo!

AV 5 – LDAP injection

• LDAP authentication in place
• Possible to manipulate LDAP queries
• May leads to enumeration OR manipulation
• Interesting attack vector
• Fault code leaks LDAP interface

AV 6 – File System access

• Identifying file system points
• Directory traversing & Access
• Leads to file access and source code exposure
• Lethal if found!
AV 7 – SOAP brute forcing

- SOAP envelope takes user & pass accounts.
- It is possible to bruteforce SOAP envelope and look for specific responses.
- This is a possible attack which can get into the system.
- Analyzing SOAP response is key for this set of attack.

AV 8 – Parameter overflow

- Adding large buffers to XML nodes
- Depending on code controls – It may fail in handling
- Breaking the application
- May compromise as well
- Traditional buffer overflow type attacks
AV 9 – Operating System access
• Point to OS
• Remote command execution is possible
• Either by "|" or ";"  
• Attack is very much possible
• Leads to admin/root on the box…

AV 10 – Session hijacking
• Web services can maintain sessions  
  - [WebMethod(EnableSession=true)]
• Possible to reverse engineer session
• Cookie tempering is reality…
• Can be compared to traditional web application session.

Other attacks
• External referencing – XML schema
• XSS attack
• In transit attacks – replay and spoofing
Agenda

- Web 2.0
  - Industry
  - Technologies
  - Security
- Impact of Web 2.0
- Ajax
  - Basics
  - Attacks
  - Methodology
  - Fingerprinting
  - Enumeration
  - Crawling
  - Scanning
  - Vulnerabilities
  - Securing Ajax

- Web Services
  - Basics
  - Methodologies
  - Security
  - Assessment
  - Footprinting
  - Discovery
  - Enumeration
  - Attack vectors
  - Defense

Defense 1 SOAP filtering

- Regular firewall will not work
- Content filtering on HTTP will not work either since it is SOAP over HTTP/HTTPS
- SOAP level filtering and monitoring would require
- ISAPI level filtering is essential
- SOAP content filtering – products or in-house

Content filtering
Defense 2 WSDL hardening

- WSDL is major source of information
- Should not have any leakage
- Only provide necessary methods
- Invokes over SSL only
- Thorough WSDL hardening

Defense 3 Authentication & Authorization

- WSDL access control
- Credentials – WS-Security
- Certificate analysis
- SOAP and XML filtering before access
Defense 4 Secure Coding

• Fault code management and Exception control
• Input validation
• SQL integration
• Levels of coding - using different components

Defense 5 XML parsing

• Good XML parsing should be used
• .Net/J2EE – may have issues with XML parsing
• Buffer overflows using schema poisoning

Thanks!

Email - shreeraj@net-square.com
Blog - http://shreeraj.blogspot.com