Security Vulnerabilities in SOHO Routers

Craig Heffner, Derek Yap
www.sourcesec.com

Introduction

With embedded devices permeating today's home networks, they have begun to attract a higher level of scrutiny from the security community than in previous years. In particular, the members of GNUCitizen have been relentlessly testing routers and wireless access points. Their discovery of multiple vulnerabilities in the BT Home Hub router affected a wide range of home networks in the UK [1], and their Router Hacking Challenge prompted a flurry of vulnerability reports against a variety of popular home routers, including the venerable Linksys WRT54G [2]. Specific vulnerabilities in home routers range from traditional Web attacks, such as XSS and CSRF, to authentication bypass attacks and buffer overflows; it is assumed that the reader has at least a passing knowledge of the attacks described in this paper.

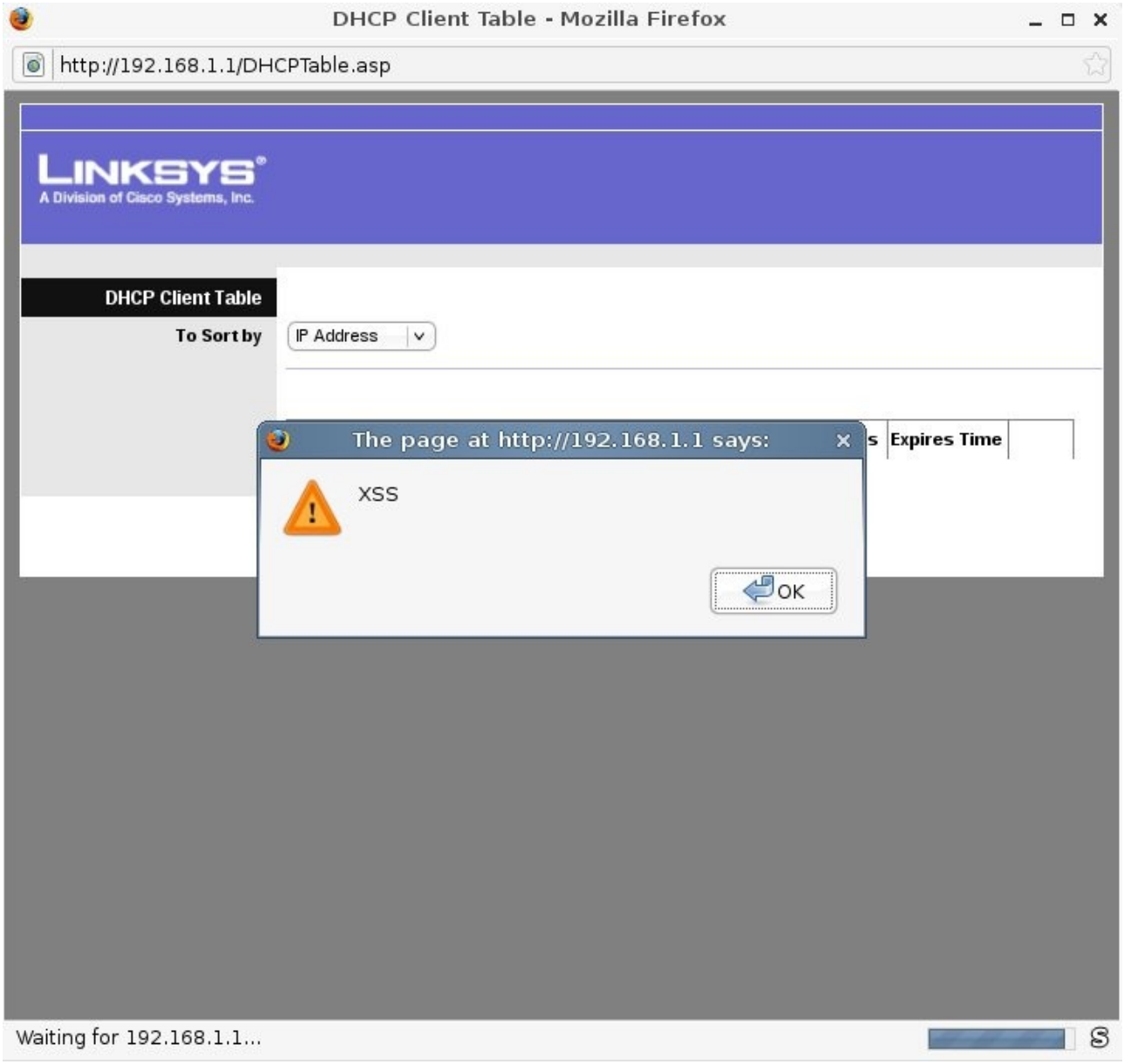
The purpose of this paper is to outline the security measures being taken by vendors to prevent such attacks in their home routing products, what those security measures accomplish, and where they fall short. We will use existing network tools to examine common vulnerabilities in a range of popular devices and demonstrate weaknesses in the security of those devices; additionally, we will examine common trends in security measures that have been duplicated across vendors, and examine how those trends help and hinder the security of their devices. In particular, we will examine the following home routers, which are some of the latest offerings from their respective vendors at the time of this writing:

- Linksys WRT160N
- D-Link DIR-615
- Belkin F5D8233-4v3
- ActionTec MI424-WR

Un-Authenticated Cross Site Scripting

Although XSS (Cross-Site Scripting) attacks are usually associated with online Web applications, nearly all home routers supply a Web-based administrative interface through which a user can easily manage the router. While viable XSS attack vectors against most routers may seem improbable given that users must authenticate to the router before accessing any interactive portion of the Web interface, one common method of injecting user-supplied data into the Web interface is injection via DHCP requests [3]. This is due to the fact that most routers will display a list of connected clients, along with their associated host names, in the administrative portion of the Web interface. The displayed host names are reported in the DHCP request packets that the client sends to the router (the router also acts as a DHCP server for the local network). Since requesting an IP address does not require authentication, any user who has access to the local network can perform such an attack by injecting JavaScript/HTML code into the administrative page via their host name.

While this is a known attack, some major vendors still ignore it. For example, one of Linksys newest offerings in the home router arena, the WRT160N, does not properly sanitize the host name value supplied by DHCP clients. This allows a malicious user on the network to inject arbitrary JavaScript into the DHCP Client Table page of the WRT160N's administrative Web interface. Scapy, a packet crafting tool, is well suited for demonstrating improper DHCP input validation [4]; to verify the existence of XSS in the WRT160N, Scapy can be used to send a DHCPREQUEST packet to the router, specifying a host name such as **');alert('XSS');//**. Not only will this allow JavaScript execution inside the administrator's browser, it also results in a failure of the Web page to properly display hosts in the list of connected clients. A similar situation exists in the D-Link DIR-615 router, where injecting invalid XML/HTML tags into a host name corrupts the XML-based clients database, causing the administrative interface to report that no hosts are connected to the network even when they are. A demonstration of the WRT160N attack can be seen below:


DHCP Client Table - Mozilla Firefox
http://192.168.1.1/DHCPTable.asp
LINKSYS
A Division of Cisco Systems, Inc.
DHCP Client Table
To Sort by
IP Address
The page at http://192.168.1.1 says:
XSS
OK
Expires Time
Waiting for 192.168.1.1...

Domain Name Hijacking

Another host-name related attack vector, again involving DHCP, is domain name hijacking [5]. This attack occurs when a router resolves internal host names to their respective IP addresses; as in the DHCP XSS attack, the internal client's host name is specified inside a DHCPREQUEST packet. This in itself is not a particular concern, but if an attacker can register themselves on the network with a host name of WPAD then they can carry out any number of man-in-the-middle attacks against other clients on the network [6]. WPAD attacks primarily affect Windows users, and Internet Explorer users in particular, as various Windows applications (including IE) will look for a WPAD server by default.

This problem is further complicated on home networks where no domain name is configured. Normally, host names will be registered as sub-domains of the network domain; i.e., if the domain name is “home”, then a host named “laptop” will be registered as “laptop.home”. However, small networks rarely have a domain name configured, so the host would simply be registered on the LAN as “laptop”. Thus, performing a DNS lookup for “laptop” would return the IP address of the internal client who registered the host name of “laptop”. But what if a host claims that its host name is “www.google.com”? Logic would suggest that a router would know better than to resolve requests for www.google.com to an internal IP address, but unfortunately that is exactly what some routers do; this allows an internal attacker to perform a single-packet DNS poison that will persist until the attacker either un-registers his host name, or leaves the network. One example is the ActionTec MI424-WR, a popular router that is currently being distributed to Verizon FiOS customers. A scripted Scapy command can easily demonstrate such an attack by attempting to register a client machine with the host name of www.google.com:

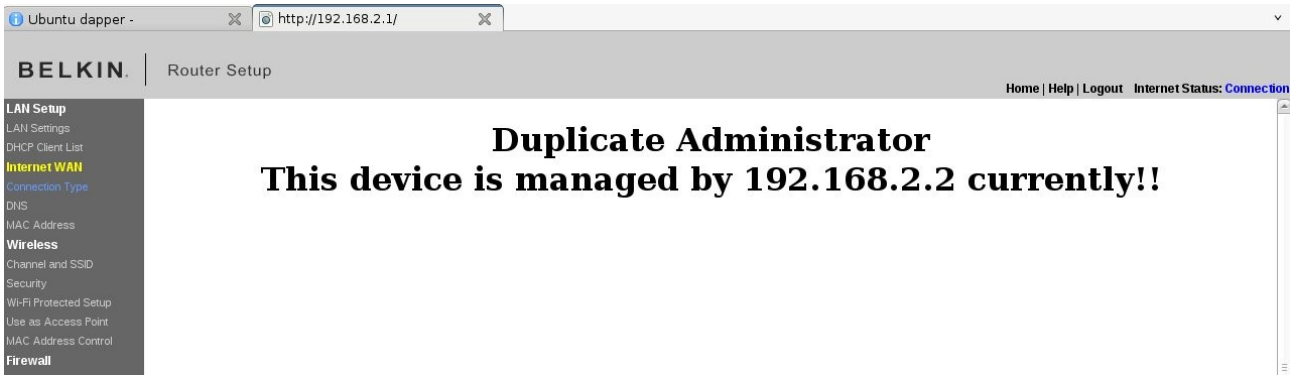
```
heff@lappy386 ~/Tools/DHCPXSS $ dig www.google.com

; <<>> DiG 9.4.2 <<>> www.google.com
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 31818
;; flags: qr rd ra; QUERY: 1, ANSWER: 5, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;www.google.com.                        IN      A

;; ANSWER SECTION:
www.google.com.         576781  IN      CNAME   www.l.google.com.
www.l.google.com.       298     IN      A       64.233.169.99
www.l.google.com.       298     IN      A       64.233.169.147
www.l.google.com.       298     IN      A       64.233.169.103
www.l.google.com.       298     IN      A       64.233.169.104

;; Query time: 13 msec
;; SERVER: 192.168.1.1#53(192.168.1.1)
;; WHEN: Sat Oct 25 12:17:06 2008
;; MSG SIZE  rcvd: 116

heff@lappy386 ~/Tools/DHCPXSS $ sudo ./dss.py -d 192.168.1.1 -h www.google.com 1>/dev/null
heff@lappy386 ~/Tools/DHCPXSS $ dig www.google.com

; <<>> DiG 9.4.2 <<>> www.google.com
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 27583
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;www.google.com.                        IN      A

;; ANSWER SECTION:
www.google.com.         3600    IN      A       192.168.1.6

;; Query time: 2524 msec
;; SERVER: 192.168.1.1#53(192.168.1.1)
;; WHEN: Sat Oct 25 12:17:18 2008
;; MSG SIZE  rcvd: 48
```

Session Hijacking

Due to the prolific use of the HTTP authentication mechanism employed by older routers, hijacking an administrative session was not difficult to an attacker who had access to LAN. If the administrative session was active on the wireless connection, the credentials (transmitted in plain text) could simply be sniffed passively from the network; if not, then an ARP poison attack would likely be required to capture the administrative traffic, but ARP poisoning is a trivial matter. Because HTTP authentication requires that the credentials be sent with every request, this method of capturing the login credentials was effective, provided there was an active administrative session in place.

A trend that we have seen in almost all new home routers is a move away from the HTTP authentication mechanism; instead, new routers are more likely to employ a Web-based login. Further, IP address authorization is used in place of the more traditional session cookie-based authorization schemes employed by most Web authentication mechanisms. While these IP-based sessions usually have a reasonable timeout period (about 10 minutes in most cases), and their use of Web based authentication means that the administrative credentials are sent only once, IP-based authorization is a poor scheme at best.

If an attacker suspects that an administrative session has been initiated, he can simply perform an ARP scan of the network to discover all of the active IP addresses on the network, and manually change his IP address to each of those until he is allowed access to the router. Some routers, such as the Belkin F5D8233-4v3, are even kind enough to tell you which IP address is currently managing the router:



Additionally, of the routers we examined that used IP-based authorization, none of them made a distinction between the wired and wireless segments of the network. This means that if a valid user logs in to the administrative interface from the wired segment of the network, an attacker on the wireless segment can change his IP address to match that of the valid user's machine and the router will allow both users to access administrative Web content.

The downside to this type of attack is that the attacker's access is limited; the session will remain open only until the valid user logs out, or until the session expires. However, some routers expose the administrative credentials to authenticated administrators, since, they believe, it is the administrator who is viewing the page. However, in a session-hijacking attack, this allows the attacker to retrieve the administrator's credentials, providing him with persistent access to the router. The Belkin F5D8233-4v3 is a good example of such a device; not only is it possible to perform session hijacking against this device, but the administrative password can be retrieved by viewing the source of the /system.stm page:

```
document.tF.timezone_daylightsaving.value=1;
document.tF.remote_mgmt_port.value=8080;
document.tF.remote_mgmt_ip_start.value='0.0.0.0';
document.tF.nat_enabled.value=1;
document.tF.upnp_enabled.value=1;
document.tF.login_timeout.value=10;
document.tF.http_passwd.value='adminpass';
document.tF.lang_code.value=0;
document.tF.country_codevalue.value='0';
document.tF.allow_remote_ip.value=0;
document.tF.remote_mgmt_enabled.value=0;
</SCRIPT>
<!--timezone-->
```

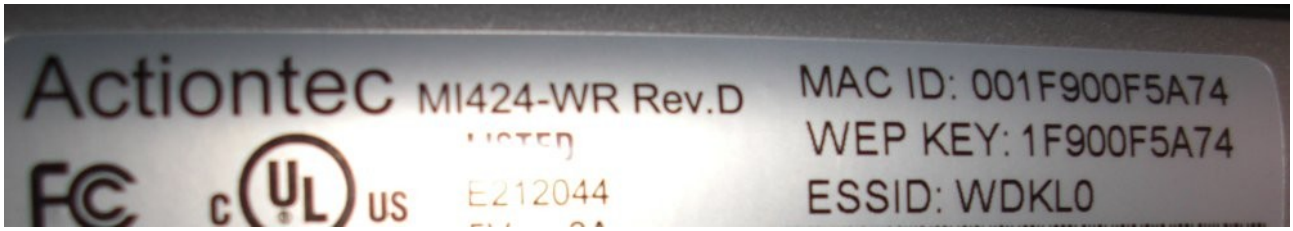
Likewise, the D-Link DIR-615 also reveals the plain text administrative password in its /Tools/Admin.shtml page:

```
 */
var local_debug = ("" === "") ? false : true;
mf = document.forms.mainform;
if (local_debug) {
        hide_all_ssi_tr();
        web_server_allow_wan_http_selector(false);
        wan_web_ingress_filter_name_selector(true);
        return;
}

/*
 * Get the current admin password (empty if logged in as user).
 */
var admin_pwd = "adminpass";
if ("true" === "true") {
        mf.password.value = admin_pwd;
        mf.password_verify.value = admin_pwd;
}

is_router_mode = "0" == "0";
if (is_router_mode) {
        web_server_allow_wan_http_selector(mf.web_server_allow_wa
```

Default WEP Keys

Default configurations are normally not considered “vulnerabilities” in and of themselves, however, any type of default setting becomes an issue when applied to cryptography. WEP and WPA keys are of particular interest with home routers, since few routers come without wireless capabilities these days. You will notice that all of the described attacks have so far required access to the LAN; wireless provides an attacker with access to the LAN, but still affords him the ability to remain reasonably removed from the LAN's physical location. In an effort to help protect users from wireless attacks, some vendors have begun shipping their products with wireless encryption enabled by default; unfortunately, the encryption method normally chosen is WEP (well known to be broken [15]), and as in the case of the BT Home Hub router, the proprietary algorithm used for generating the default WEP key can be reverse engineered and used by an attacker to gain access to such encrypted networks [8].

Many newer home routers still come with no encryption enabled, however, one notable exception is the ActionTec MI424-WR. This particular router is commonly distributed by Verizon, and invariably a plethora of them can be found in areas where Verizon FiOS is available. Unlike the BT Home Hub, the ActionTec routers do not attempt to obscure the method used to generate their default 40 bit WEP key:



As you can see, the WEP key is the low 40 bits of the router's MAC address. Although some have claimed that the MAC address used to generate the key is the wired MAC address and not the WiFi MAC address [9], the two are in fact one and the same [13]. Because WEP does not encrypt source/destination MAC addresses, any data packets to or from the ActionTec router will instantly reveal the WEP key. Also note that no active clients need be on the network in order for data packets to be generated, as the ActionTec routers are prone to periodically broadcasting un-solicited Spanning-Tree packets.

While a 40 bit WEP key can be cracked in a matter of minutes using tools such as Aircrack [15], setting up a default WEP key is a poor practice for two reasons:

1. It provides the user with a false sense of security
2. It allows an attacker to instantly, and completely passively, obtain the network's WEP key

To demonstrate the effectiveness of gleaning default WEP keys vs active WEP attacks, a Scapy script was run for fifteen seconds using an internal Intel wireless card from inside a suburban-area building. In that period, six WEP-enabled ActionTec access points were discovered; note that their SSIDs match the default SSID patterns set by Verizon, indicating that their default settings have likely not been modified:

```
$ sudo ./acattack.py -i wlan0
Placing wlan0 into monitor mode, please wait...
Capturing packets. Ctl+C to stop...

     BSSID              Router MAC         WEPKey      SSID
00:18:01:f2:c5:1d  00:18:01:56:53:a6  18015653A6  D32D9
00:18:01:f2:59:5c  00:18:01:58:35:81  1801583581  F3LJW
00:18:01:f2:c2:d1  00:18:01:56:80:74  1801568074  D3O7I
00:18:01:f3:9d:ed  00:18:01:5b:e1:98  18015BE198  L3O82
00:18:01:eb:e3:78  00:18:01:30:48:26  1801304826  V1EIT
00:18:01:f3:6c:70  00:18:01:5a:d5:91  18015AD591  J3PAL

Found 6 networks
Restoring wlan0 to managed mode, please wait...
```

Cross-Site Request Forgery Attacks

Cross-Site Request Forgery (CSRF) attacks against home routers is nothing new [7]. However, because most home routers have traditionally used basic HTTP authentication, the return vs risk ratio was considerably high. If a user happened to have authenticated to their router during that same browsing session, then a CSRF attack would be executed without the user's knowledge. However, this scenario is very unlikely considering that most users rarely (if ever) log in to their router's administration area, and the user would instead get a pop-up HTTP authentication box from their router.

To combat this, CSRF attacks started embedding the default logins for popular routers into their requests; if the supplied user name and password combination were correct, then the attack would succeed, else, the user would be prompted with a pop-up HTTP authentication box from their router. Since many people do not change their default passwords, this attack was fairly effective. However, browser vendors began protecting against this type of attack. Firefox, for example, displays the following message when attempting to exploit a CSRF attack in the described manner:

A trend that we have seen in almost all new routers is a move away from the HTTP authentication mechanism; instead, new routers are more likely to employ a Web-based login. Further, IP address authorization is used in place of the more traditional session cookie-based authorization schemes employed by most Web authentication mechanisms. While these IP-based sessions usually have a reasonable timeout period (about 10 minutes in most cases), this shifts the return vs risk ratio greatly in the attacker's favor because there will be no HTTP authentication message boxes: if a victim user happens to be authenticated to their router, then the attacker's CSRF attack will work; if not, then the attack will silently fail, but in either case the user is completely unaware. Further, the attacker can still attempt to authenticate to the router via forged HTTP GET/POST requests using known default credentials, thus authenticating the client's IP address before carrying out the remainder of the CSRF attack.

Authentication Bypass

Perhaps the worst type of vulnerability for any networked device is an authentication bypass vulnerability. These have been found in numerous home routers, including the popular Linksys WRT54G [16]. In home routers, these vulnerabilities are often due to the fact that only the user-interface pages of the Web interface check for authentication, while the scripts that actually process configuration changes do not. In the case of the Belkin F5D8233-4v3, the administrative scripts (located in /cgi-bin/) do check the login status of the user, and if the user has not authenticated redirects the user back to the login page. However, the scripts process the user's request *before* redirecting them to the login page; this allows any client on the network to make administrative changes without any prior authentication or authorization. Additionally, these scripts accept arguments via either GET or POST parameters. Coupling this vulnerability with a CSRF attack can prove to be devastating, as the CSRF requests can simply use image tags to modify the router's settings, eliminating the need for JavaScript to be enabled in the victim's browser. Some of the possible attack vectors for this particular vulnerability are shown in the table below:

Configure the router's primary DNS server	/cgi-bin/setup_dns.exe? dns1_1=192&dns1_2=168&dns1_3=2&dns1_4= 2
Enable remote management on port 8080	/cgi-bin/system_setting.exe? remote_mgmt_enabled=1&remote_mgmt_port= 8080
Restore the router's default factory settings	/cgi-bin/restore.exe
Reboot the router	/cgi-bin/restart.exe
Log in with the default password	/cgi-bin/login.exe?pws=

Authenticated Cross Site Scripting

We have already discussed how XSS attacks against most routers seem improbable given that users must authenticate to the router before accessing any interactive portion of the Web interface. For this reason, improper input validation in the interactive portions of a router's administrative interface may appear innocuous, and often times data supplied by an administrator is trusted more than it should be. However, as we have demonstrated, there are several methods in which an attacker can hijack or spoof an administrative session with the router. In doing so, the attacker can inject malicious HTML or JavaScript code into the administrative interface via these supposed harmless XSS bugs. When coupled with other attacks, such as CSRF attacks, these bugs can lead to an elevated risk of exploitation, or even allow the attacker to place a persistent back door inside the administrative interface.

One such example is that of the D-Link DIR-615. As we have already discussed, the DIR-615 makes the administrative password available in plain text to authenticated administrative users. However, using a combination of CSRF and XSS, a remote attacker can obtain this user name if the authenticated user browses to a Web page controlled by the attacker. There is a non-persistent XSS bug in the 'Schedule Name' parameter of the /Tools/Schedules.shtml page. This can be exploited by an attacker by loading an IFRAME that posts a schedule named </script><script>alert(1)</script><script> to the router; if the client who is viewing the attacker's page has recently authenticated to the D-Link router, then the attacker's code will run in the context of the administrative page, and can retrieve the contents of the /Tools/Admin.shtml page, which includes the administrative password.

UPNP Exploitation

UPNP attacks are nothing new [10], but started receiving more attention after GNUCitizen demonstrated that UPNP attacks could be carried out remotely when coupled with flash-based CSRF attacks [11]. Because UPNP is an unauthenticated protocol that, by definition, provides control over a router's configuration, insecure UPNP stacks can result in a plethora of exploitation possibilities, including command execution and re-configuration of DNS settings. While most new routers protect against these attacks, there is another UPNP action that we can use to our advantage.

The previously mentioned session hijacking attacks (and some of the CSRF attacks) require an administrator to already be authenticated with the target router. But waiting around for the average user to log into their router makes these attacks unlikely to succeed. Instead, an attacker can use UPNP to terminate a router's WAN connection, interrupting the user's Internet connection. Eventually, they are likely to:

1. Reset their router
2. Log into the router to diagnose the problem
3. Call their ISP, who will ask them to log into their router to diagnose the problem

The WAN connection can be terminated using the UPNP ForceTermination action, which was available in all of the routers that we examined. Using Miranda [14], a UPNP administration utility, we can easily send UPNP commands to a router, forcing it to terminate it's WAN connection:

```
upnp> host send 0 WANConnectionDevice WANIPConnection GetExternalIPAddress

NewExternalIPAddress : 71.127.154.130

upnp> host send 0 WANConnectionDevice WANIPConnection ForceTermination


upnp> host send 0 WANConnectionDevice WANIPConnection GetExternalIPAddress

NewExternalIPAddress : 0.0.0.0

upnp>
```

In the case of a remote CSRF attack, an attacker could use flash to attempt to terminate a user's connection and wait for them to log into the router before running the CSRF attack, although in practice this is unlikely to succeed.

One of the most common uses for UPNP is port forwarding. UPNP allows client applications, such as P2P programs and games, to open ports on the router in order to facilitate necessary communications with other peers or services. While these port forwarding rules are meant to forward traffic from external clients to internal clients, an attacker can make use of these rules to expose the router's administrative interface to the WAN by forwarding traffic to port 80 of the router's IP address. Configuring the router as the attacker's personal proxy is also possible, by telling the router to forward traffic not to an internal IP, but an external IP [12]. While most new routers prevent these types of attacks by checking the specified IP addresses, some, like the ActionTec MI424-WR, still allow users to forward incoming connections on external ports to port 80 of the router itself, effectively enabling remote administration on the device.

Router Host Names

Perhaps the one major hurdle in remote CSRF and UPNP attacks is knowledge of the router's internal IP address. If the user has modified the router's IP address, or the address range of the LAN, then the attacker is out of luck. However, one of the trends noted during examination of these home routers is that they all resolve themselves to a unique host name. For example, the ActionTec MI424-WR router's host name is “Wireless_Broadband_Router.home”. This allows an attacker to reference the router by its host name rather than IP address during any attack, mitigating the possibility of failure due to an invalid IP.

Conclusion

Router manufacturers are increasing the security of their devices, however, home router security still has a long road ahead of it. Below is a table listing each of the devices and their associated, reasonably exploitable, vulnerabilities mentioned in this paper; these types of vulnerabilities must be considered by all vendors, and should be investigated by any consumer before purchasing a router.

Vulnerability	ActionTec MI424-WR	Linksys WRT160N	D-Link DIR-615	Belkin F5D8233-4v3
Unauthenticated XSS	No	Yes	No	No
Authenticated XSS	No	No	Yes	No
DNS Hijacking	Yes	No	No	No
Session Hijacking	No	No	Yes	Yes
Default WEP	Yes	No	No	No
“Silent” CSRF	No	No	Yes	Yes
Authentication Bypass	No	No	No	Yes
Local UPNP	Yes	Yes	Yes	Yes
CSRF UPNP	Yes	No	No	No

References

[1] **BT Home Flub: Pwnin the BT Home Hub**, Adrian Pastor,
<http://www.gnucitizen.org/blog/bt-home-flub-pwnin-the-bt-home-hub/>

[2] **Router Hacking Challenge**, Petko D. Petkov,
<http://www.gnucitizen.org/blog/router-hacking-challenge/>

[3] **DHCP/mDNS Injection Issues**, Petko D. Petkov,
<http://www.gnucitizen.org/blog/dhcpmdns-injection-issues/>

[4] **Scapy**, Philippe Biondi, <http://www.secdev.org/projects/scapy/>

[5] **DHCP Name Poisoning Attacks**, Petko D. Petkov,
<http://www.gnucitizen.org/blog/r00ting-public-wifi-networks-dhcp-name-poisoning-attacks/>

[6] **WPAD: Internet Explorers Worst Feature**, Grant Bugher,
<http://perimetergrid.com/wp/2008/01/11/wpad-internet-explorers-worst-feature/>

[7] **CSRF Pharming**, Joe Walker,
<http://directwebremoting.org/blog/joe/2007/02/08/csrf_pharming.html>

[8] **Default Key Algorithms in Thompson and BT Home Hub Routers**, Adrian Pastor,
<http://www.gnucitizen.org/blog/default-key-algorithm-in-thomson-and-bt-home-hub-routers/>

[9] **Re: Verizon FiOS Default WEP Key Highly Insecure, ENIQomios**,
<http://www.dslreports.com/forum/r21179302-Verizon-FiOS-default-WEP-key-HIGHLY-insecure>

[10] **UPnP Hacks**, Armijn Hemel, <http://www.upnp-hacks.org/suspect.html>

[11] **Flash UPNP Attack FAQ**, Petko D. Petkov,
<http://www.gnucitizen.org/blog/flash-upnp-attack-faq/>

[12] **UPnP Hacks: Common Errors**, Armijn Hemel, <http://www.upnp-hacks.org/igd.html>

[13] **Verizon FIOS (and DSL?) wireless access point insecure default WEP key**, paul14075,
<http://seclists.org/bugtraq/2008/Sep/0311.html>

[14] **Miranda** , SourceSec.com,
<http://www.sourcesec.com/2008/11/07/miranda-upnp-administration-tool/>

[15] **Aircrack-ng**, Aircrack-ng.org, <http://www.aircrack-ng.org>

[16] **Linksys WRT54G Authentication Bypass**, Ginsu Rabbit,
<http://www.securityfocus.com/archive/1/442452/30/0/threaded>