Hal Berghel

Hijacking the Web

Cookies revisited: Continuing the dialogue on personal security and underlying privacy issues.

Based on the positive feedback I received regarding my column, "Caustic Cookies" (May 2001), I conclude there is a genuine interest in the technical aspects underlying privacy and security issues relating to Internet use.

As I said in my May column, the Web is all about a pair of killer Internet protocols—HTML and HTTP—that define, enable, and constrain Web applications. HTTP is the application layer protocol that sits on top of the Transmission Control Protocol (TCP) which, in turn, sits atop the Internet Protocol (IP). The collection of utilities relating to these protocols that reside between the physical communications layer of the Internet, and the productivity tools that we use to get work done (for instance, Web browsers and email clients) is called the TCI/IP protocol suite (a.k.a protocol stack).

The HTTP part of this protocol suite is "stateless." Under the typical scenario, this means that once an initial communication exchange between a client and a server is completed, the connection between them is dropped. This communication exchange is all built around what is commonly called the "TCP three-way handshake," the motivation for which results from the fact that IP is "lossy"—that is, if a packet gets lost in an IP transmission, it's gone forever. TCP overcomes this deficiency by keeping track of each leg of the communication exchange. At its most simple level, the three-way handshake works something like this:

Stage 1—the SYN phase. A client sends an initial synchronizing (SYN) packet to some server with an initial sequence number (ISN) = x.
Stage 2—the SYN + ACK phase. The server accepts the SYN packet from the client, and sends an acknowledgment back with an ISN = y and alerts the client that it awaits another packet from the client with an ISN of x + 1.
Stage 3—the ACK phase. The client accepts the SYN + ACK packet and sends an ACK packet back to the server with an ISN of x + 1.

Now both client and server are connected and can begin tri-phase dialogues.

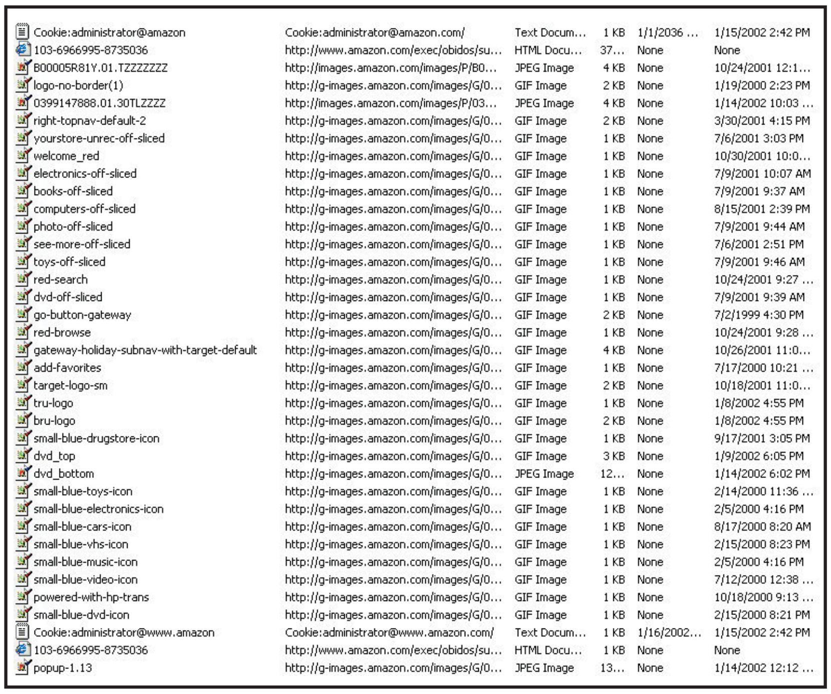
Note that in this exchange data persistence ends at Stage 3, so if we're to build upon previous dialogues, we'll have to do it by echoing the contents of previous communications. That's why the Web community began thinking about state management mechanisms in the mid-1990s. The primary state management mechanism they came up with was the "cookie" (see sidebar).

The Causticity of Cookies

Cookies are used by Web applications as a surrogate for the current state of communication. They have the following properties:

• They are binary data stored on

JASON SCHNEIDER



Cookie:administrator@amazon	Cookie:administrator@amazon.com/	Text Docum…	1 KB	1/1/2036 …	1/15/2002 2:42 PM
103-6966995-8735036	http://www.amazon.com/exec/obidos/su…	HTML Docu…	37…	None	None
B00005R81Y.01.TZZZZZZZ	http://images.amazon.com/images/P/B0…	JPEG Image	4 KB	None	10/24/2001 12:1…
logo-no-border(1)	http://g-images.amazon.com/images/G/0…	GIF Image	2 KB	None	1/19/2000 2:23 PM
0399147888.01.30TLZZZZ	http://images.amazon.com/images/P/03…	JPEG Image	4 KB	None	1/14/2002 10:03 …
right-topnav-default-2	http://g-images.amazon.com/images/G/0…	GIF Image	2 KB	None	3/30/2001 4:15 PM
yourstore-unrec-off-sliced	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	7/6/2001 3:03 PM
welcome_red	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	10/30/2001 10:0…
electronics-off-sliced	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	7/9/2001 10:07 AM
books-off-sliced	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	7/9/2001 9:37 AM
computers-off-sliced	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	8/15/2001 2:39 PM
photo-off-sliced	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	7/9/2001 9:44 AM
see-more-off-sliced	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	7/6/2001 2:51 PM
toys-off-sliced	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	7/9/2001 9:46 AM
red-search	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	10/24/2001 9:27 …
dvd-off-sliced	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	7/9/2001 9:39 AM
go-button-gateway	http://g-images.amazon.com/images/G/0…	GIF Image	2 KB	None	7/2/1999 4:30 PM
red-browse	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	10/24/2001 9:28 …
gateway-holiday-subnav-with-target-default	http://g-images.amazon.com/images/G/0…	GIF Image	4 KB	None	10/26/2001 11:0…
add-favorites	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	7/17/2000 10:21 …
target-logo-sm	http://g-images.amazon.com/images/G/0…	GIF Image	2 KB	None	10/18/2001 11:0…
tru-logo	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	1/8/2002 4:55 PM
bru-logo	http://g-images.amazon.com/images/G/0…	GIF Image	2 KB	None	1/8/2002 4:55 PM
small-blue-drugstore-icon	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	9/17/2001 3:05 PM
dvd_top	http://g-images.amazon.com/images/G/0…	GIF Image	3 KB	None	1/9/2002 6:05 PM
dvd_bottom	http://g-images.amazon.com/images/G/0…	JPEG Image	12…	None	1/14/2002 6:02 PM
small-blue-toys-icon	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	2/14/2000 11:36 …
small-blue-electronics-icon	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	2/5/2000 4:16 PM
small-blue-cars-icon	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	8/17/2000 8:20 AM
small-blue-vhs-icon	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	2/15/2000 8:23 PM
small-blue-music-icon	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	2/5/2000 4:16 PM
small-blue-video-icon	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	7/12/2000 12:38 …
powered-with-hp-trans	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	10/18/2000 9:13 …
small-blue-dvd-icon	http://g-images.amazon.com/images/G/0…	GIF Image	1 KB	None	2/15/2000 8:21 PM
Cookie:administrator@www.amazon	Cookie:administrator@www.amazon.com/	Text Docum…	1 KB	1/16/2002…	1/15/2002 2:42 PM
103-6966995-8735036	http://www.amazon.com/exec/obidos/su…	HTML Docu…	1 KB	None	None
popup-1.13	http://g-images.amazon.com/images/G/0…	JPEG Image	13…	None	1/14/2002 12:12 …

Figure 1. The temporary Internet files resulting from a single access to www.amazon.com.

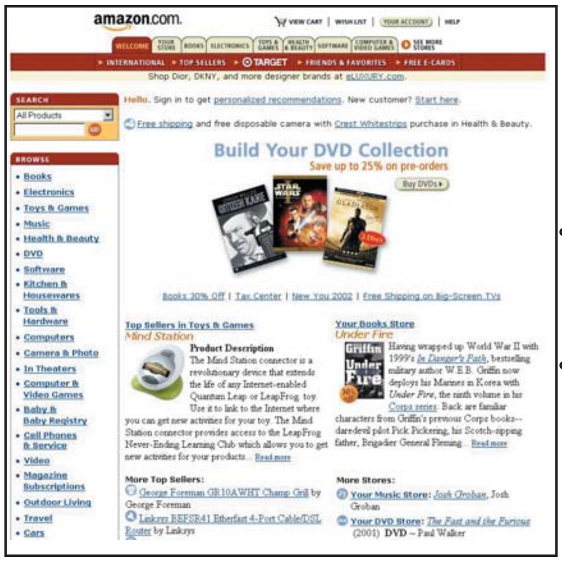


Figure 2. This downloaded page from Amazon.com contains a bounty of hidden data when viewed as HTML source.

the user's computer hard drive(s).

- The data will likely be either encoded or encrypted (the casual observer will not be able to understand the contents).
- The content of each cookie is determined exclusively by the server.
- Cookies can consist of up to 20 strings of 4,096 bytes in length (at least in the original Netscape proposal was initially adopted by the Internet Engineering Task Force).
- Cookies can and do record clickstream information about the user's Web browsing habits.
- Cookies can be shared by third-party Web hosts with neither the user's knowledge nor permission. (Not surprisingly, these are called third-party cookies.)
- They may be invoked by malicious or poorly designed servers to produce denial-of-service attacks (cookie storms) that can disrupt applications and normal network traffic.
- The user may not exercise control over the ultimate destination of cookies (cookie leakage).
- There is no control over the content of a cookie. Specifically, there is no way to prevent a cookie from containing sensitive, classified, or otherwise secure or private information. In fact, cookies routinely store session IDs, user IDs, and passwords.
- Web browsers may store cookies from unvisited sites without the user's knowledge or permission.
- Browsers do not allow you to cancel cookies accepted prior to disabling cookie acceptance in your browser. This has to be done manually (in the parlance of netizens this is called "tossing your cookies").
- Cookies are stored in a variety of places on your hard drive—in cookie directories under user folders within "documents and settings" directories, in bookmark folders like "Internet_www _related," to name two.
- Netscape and Microsoft handle cookies in different ways, so it is not possible to use a

one-size-fits-all strategy to manually manage them in both browser environments.

These properties cause some of us to call into question the desirability of using cookies in the first place. In my view, to borrow a phrase from Dijkstra, cookie technology is a mistake carried through to perfection—not because the concept of transaction management is misguided, after all, there is nothing wrong with the concept of an e-commerce shopping cart, but because this paradigm was set up with inadequate protections and safeguards for end users. For these sorts of reasons, the Clinton administration banned cookies from federal Web sites in the absence of "compelling need" to the contrary in June, 2000.

So one may imagine the members of the IETF Working Group on cookies trying to balance the need for Web transaction management with the vulnerabilities associated with the properties previously mentioned. Alternatives to cookies such as URL encoding, the use of HTML hidden fields, or storage on the server side seemed fraught with difficulties, so perhaps they chose the path of least resistance with the hope that the Web world would behave with some concern for individual privacy and security. After all, hope springs eternal, even on the Internet.

Now for the Bad News

It's time for a reality check. First, every Web "handshake" between client and server is potentially invasive. To illustrate the point, on January 15, 2002, I connected to Amazon.com with the default settings of my Explorer browser just as it came from the developer. Note that entering www.amazon.com produced this extended URL with some additional information embedded: www.amazon.com/exec/obidos/subst/home/home.html/103-9484532-4674260. I hadn't even shopped for anything and already some user-auditing was taking place. That apparent plaintext number appended to the extended URL is curious, isn't it? Oh, well, isn't a URL just ephemeral data in a browser window?

Here the proverbial plot thickens. Unbeknownst to me, a cornucopia of temporary Internet files were dumped on my hard drive (see Figure 1). Assuming that there's nothing insidious (like Web bugs, an unwise assumption to make, by the way) embedded in the graphics, the only real penalties are the waste of 100KB of disk space and the fact that the address of the Amazon.com cookie administrator was permanently stored on my hard drive. I would prefer to retain the 100KB of disk space and remain anonymous to cookie administrators, but we'll write this off to experience, too.

But this is just the beginning. In addition to the mysterious string added to the URL and the 100KB of Web guano deposited on my hard drive, two cookies were deposited on my hard drive as well:

Cookie 1:

```
seenpop
1
www.amazon.com/
1600
1778814848
29466237
3595971744
29466136
*
```

Cookie 2:

```
session-id
103-9484532-4674260
amazon.com/
1536
3392438272
29467418
3569411744
29466136
*
session-id-time
1011686400
amazon.com/
1536
3392438272
29467418
3569571744
29466136
*
ubid-main
430-0087201-7219178
amazon.com/
1536
2916341376
31961269
3571131744
29466136
*
x-main
hQFiIxHUFj8mCscT@Yb5Z7xsV
sOFQjBf
amazon.com/
1536
2916341376
31961269
3571131744
29466136
*
```

To borrow a phrase from Dijkstra, cookie technology is a mistake carried through to perfection.

Let's see if we can figure out what some of this means. First, observe that both cookies are linked by some common field values. This suggests some sort of candidate key for a database. Hmmmm. I wonder if there isn't a transaction database in the background. Second, note that there's a field identified as a session ID in the second cookie. Where have we seen that before? Yep, that was the mysterious number at the end of the extended URL. This session ID serves essentially the same role at the applications layer that the sequence number did within the TCP/IP protocol suite—it is part of the authentication process that the application goes through to keep track of the session activity, and link it to a user and an activity log. Now things are beginning to make sense. If we have a transaction database, and an authentication number issued to a user, we probably have identified at least part of a gateway into the database. What's more, when I view the source of the downloaded page from Amazon.com (see Figure 2), I find that same session ID number seems to be added to virtually every link on the page. Take a look at the following HTML fragment for one of the image maps.

```
<map name="right_top_nav_map">
<area shape="rect" href=/exec/obidos/
shopping-basket/ref=top_nav_sb_
gateway/103-9484532-4674260
coords="0,0,80,21">
<area shape="rect" href=/exec/
obidos/wishlist/ref=top_nav_wl_gate-
way/103-9484532-4674260
coords="85,0,151,21">
<area shape="rect"
href=/exec/obidos/account-access-
login/ref=top_nav_ya_gateway/103-
9484532-4674260
coords="155,0,256,21">
<area shape="rect" href=/exec/
obidos/tg/browse/-/508510/
ref=top_nav_hp_gateway/103-
9484532-4674260
coords="260,0,299,21">
</map>
```

Clearly the magic number 103-9484532-4674260 is a core ingredient of the transaction information is being used by Amazon.com about my current activities.

Hijacking the Web

We've reached the point of cascading absurdities. In my last column on cookies, I explained why cookies are well-intentioned mistakes. Now cookies are just part of the quagmire we get into when we engage in poorly thought-through TCP/IP state management. In the example in the previous section, I found that there are several pieces of information about our Web activities that can be spread around cyberspace.

Consider the session ID. This one piece of information appears in the extended URL, two different cookies, and the actual HTML contents of the Web page delivered. What is to prevent a hacker from changing these values and spoofing some other session ID? In a word, nothing. This is one of the primary ways hackers hijack Web sessions. This could be done by modifying the cookie with a text editor and then reconnecting to the Web site, changing the values in the extended URL, or even modifying the hard-coded information after saving the HTML page and then reloading it in the browser. A little trial and error can produce a real mess for innocent victims.

What could one achieve by doing this? For one, hijacking the session ID may reveal enough of the contents of records about users and their behavior to allow a dedicated evil-doer to circumvent the application-level authentication and pretend to be someone else. A truckload of stolen plasma monitors here, a 7-digit withdrawl there—it all adds up. Your imagination can complete the story.

This is a disaster in the making, I hear you cry. Not surprisingly, I've saved the worst for last. If we weren't vulnerable enough, there is an automated environment that takes all of the busywork out of session hijacking—specialized proxy servers.

Proxy servers are conduits between the client (browser application) and the server. The proxy server maintains a complete communication stream at both ends. But because it's an intermediary between client and server, it has the capability to intercept and alter the information as it's passed back and forth within the communication stream. Such valued morsels as session IDs, cookie contents, transaction information, prices, amounts, account numbers, passwords, and so forth are all fair game. Any session credential exchanged in the communications that is in plaintext can be easily altered. If the meaning of the information isn't clearly identified, experimentation is called for. Further, secure sockets layer (SSL) and other encryption environments are of no use because the hijacking takes place at the applications layer above SSL. Hackers have access to several different types of proxy servers that have the built-in capability of editing and transmitting session information in real time.

Actually, there's more to the story, but I'll have to deal with topics like account harvesting and database invasions in a later column.

I'll leave you with this thought: in the world of online banking and e-commerce, the price to be paid for personal security is eternal vigilance. C

Hal Bergel (www.acm.org/hlb) is professor and chair of Computer Science at the University of Nevada at Las Vegas, and a frequent contributor to the literature on cyberspace.

The Origin of the Term "Cookie"

I admitted my ignorance in my May 2001 column on the origins of the term "cookie." Fortunately, David Kristol filled me in. Kristol reminded me that the term has had a long history in computer science, and pointed me to version 4.2.9 (Jan. 2000) of Eric Raymond's online "Jargon File" (eps.mcgill.ca/jargon/jargon.html). Here are two entries from that source relevant to our discussion:

magic cookie n.
[Unix; common] (1) Something passed between routines or programs that enables the receiver to perform some operation; a capability ticket or opaque identifier. Especially used for small data objects that contain data encoded in a strange or intrinsically machine-dependent way. For example, on non-Unix OSs with a non-bytestream model of files, the result of ftell(3) may be a magic cookie rather than a byte offset; it can be passed to fseek(3), but not operated on in any meaningful way. The phrase "it hands you a magic cookie" means it returns a result whose contents are not defined but which can be passed back to the same or some other program later. (2) An in-band code for changing graphic rendition (inverse video or underlining) or performing other control functions (see also cookie). Some older terminals would leave a blank on the screen corresponding to mode-changing magic cookies; this was also called a glitch (or occasionally a "turd"; compare mouse droppings). See also cookie.

cookie n.
A handle, transaction ID, or other token of agreement between cooperating programs. "I give him a packet, he gives me back a cookie." The claim check you get from a dry-cleaning shop is a perfect mundane example of a cookie; the only thing it's useful for is to relate a later transaction to this one (so you get back the same laundry). Compare magic cookie; see also fortune cookie. Now mainstream in the specific sense of Web-browser cookies.

Netscape simply ported the term over to the world of Webdom in the mid-1990s to designate their variety of persistent identifiers.

As an aside, Kristol has written what may be the definitive history of cookies, "HTTP Cookies: Standards, Privacy and Politics," (Vol. 1, No. 2, ACM *Transactions on Internet Technology*). The abstract reveals the scope of the article: "How did we get from a world where cookies were something you ate and where non-techies were unaware of Netscape cookies to a world where cookies are a hot-button privacy issue for many computer users? This article describes how HTTP "cookies" work and how Netscape's original specification evolved into an IETF Proposed Standard. I also offer a personal perspective on how what began as a straightforward technical specification turned into a political flashpoint when it tried to address nontechnical issues such as privacy." Kristol's is an excellent article on cookies from all perspectives. C