How Internet Cookies Work

by Marshall Brain

Most Internet cookies are incredibly simple, but they are one of those things that have taken on a life of their own. Cookies started receiving tremendous media attention back in February 2000 because of Internet privacy concerns, and the debate still rages.

On the other hand, cookies provide capabilities that make the Web much easier to navigate. The designers of almost every major site use them because they provide a better user experience and make it much easier to gather accurate information about the site's visitors.

In this edition of **HowStuffWorks**, we will take a look at the basic technology behind cookies, as well as some of the features they enable. You will also have the opportunity to see a real-world example of what cookies can and cannot do using a sample page that we developed here at HowStuffWorks.

Cookie Basics

In April of 2000 I read an in-depth article on Internet privacy in a large, respected newspaper, and that article contained a definition of cookies. Paraphrasing, the definition went like this:

> Cookies are programs that Web sites put on your hard disk. They sit on your computer gathering information about you and everything you do on the Internet, and whenever the Web site wants to it can download all of the information the cookie has collected. **[wrong]**

Definitions like that are fairly common in the press. The problem is, none of that information is correct. Cookies are not programs, and they cannot run like programs do. Therefore, they cannot gather any information on their own. Nor can they collect any personal information about you from your machine.

Here is a valid definition of a cookie: A cookie is a piece of text that a Web server can store on a user's hard disk. Cookies allow a Web site to store information on a user's machine and later retrieve it. The pieces of information are stored as **name-value pairs**.

For example, a Web site might generate a unique ID number for each visitor and store the ID number on each user's machine using a cookie file.

If you use Microsoft's Internet Explorer to browse the Web, you can see all of the cookies that are stored on your machine. The most common place for them to reside is in a directory called **c:\windows\cookies**. When I look in that directory on my machine, I find 165 files. Each file is a **text file** that contains name-value pairs, and there is one file for each Web site that has placed cookies on my machine.

You can see in the directory that each of these files is a simple, normal text file. You can see which Web site placed the file on your machine by looking at the file name (the information is also stored inside the file). You can open each file by clicking on it.

For example, I have visited **goto.com**, and the site has placed a cookie on my machine. The cookie file for goto.com contains the following information:

```
    UserID    A9A3BECE0563982D    www.goto.com/
```

Goto.com has stored on my machine a single name-value pair. The name of the pair is **UserID**, and the value is **A9A3BECE0563982D**. The first time I visited goto.com, the site assigned me a unique ID value and stored it on my machine.

(Note that there probably are several other values stored in the file after the three shown above.

That is housekeeping information for the browser.)

Amazon.com stores a bit more information on my machine. When I look at the cookie file Amazon has created on my machine, it contains the following:

```
 session-id-time  954242000  amazon.com/
 session-id  002-4135256-7625846  amazon.com/
 x-main  eKQIfwnxuF7qtmX52x6VWAXh@Ih6Uo5H  amazon.com/
 ubid-main  077-9263437-9645324  amazon.com/
```

It appears that Amazon stores a main user ID, an ID for each session, and the time the session started on my machine (as well as an x-main value, which could be anything).

The vast majority of sites store just one piece of information -- a **user ID** -- on your machine. But a site can store many name-value pairs if it wants to.

A name-value pair is simply a named piece of data. It is not a program, and it cannot "do" anything. A Web site can retrieve only the information that it has placed on your machine. It cannot retrieve information from other cookie files, nor any other information from your machine.

How Does Cookie Data Move?

As you saw in the previous section, cookie data is simply name-value pairs stored on your hard disk by a Web site. That is all cookie data is. The Web site stores the data, and later it receives it back. A Web site can only receive the data it has stored on your machine. It cannot look at any other cookie, nor anything else on your machine.

The data moves in the following manner:

- If you type the URL of a Web site into your browser, your browser sends a request to the Web site for the page (see How Web Servers Work for a discussion). For example, if you type the URL **http://www.amazon.com** into your browser, your browser will contact Amazon's server and request its home page.
- When the browser does this, it will look on your machine for a cookie file that Amazon has set. If it finds an Amazon cookie file, your browser will send all of the name-value pairs in the file to Amazon's server along with the URL. If it finds no cookie file, it will send no cookie data.
- Amazon's Web server receives the cookie data and the request for a page. If name-value pairs are received, Amazon can use them.
- If no name-value pairs are received, Amazon knows that you have not visited before. The server creates a new ID for you in Amazon's database and then sends name-value pairs to your machine in the header for the Web page it sends. Your machine stores the name-value pairs on your hard disk.
- The Web server can change name-value pairs or add new pairs whenever you visit the site and request a page.

There are other pieces of information that the server can send with the name-value pair. One of these is an **expiration date**. Another is a **path** (so that the site can associate different cookie values with different parts of the site).

You have control over this process. You can set an option in your browser so that the browser informs you every time a site sends name-value pairs to you. You can then accept or deny the values.

How Do Web Sites Use Cookies?

Cookies evolved because they solve a big problem for the people who implement Web sites. In the broadest sense, a cookie allows a site to store **state information** on your machine. This information lets a Web site remember what **state** your browser is in. An ID is one simple piece of state information -- if an ID exists on your machine, the site knows that you have visited before. The state is, "Your browser has visited the site at least one time," and the site knows your ID from that visit.

Web sites use cookies in many different ways. Here are some of the most common examples:

- Sites can **accurately determine how many people actually visit the site.** It turns out that because of proxy servers, caching, concentrators and so on, the only way for a site to accurately count visitors is to set a cookie with a unique ID for each visitor. Using cookies, sites can determine:
  - How many visitors arrive
  - How many are new vs. repeat visitors
  - How often a visitor has visited

  The way the site does this is by using a **database**. The first time a visitor arrives, the site creates a new ID in the database and sends the ID as a cookie. The next time the user comes back, the site can increment a counter associated with that ID in the database and know how many times that visitor returns.

- Sites can **store user preferences** so that the site can look different for each visitor (often referred to as **customization**). For example, if you visit **msn.com**, it offers you the ability to "change content/layout/color." It also allows you to enter your zip code and get customized weather information. When you enter your zip code, the following name-value pair gets added to MSN's cookie file:

-
  ```
     WEAT  CC=NC%5FRaleigh%2DDurham®ION=  www.msn.com/
  ```

  Since I live in Raleigh, NC, this makes sense.

  Most sites seem to store preferences like this in the site's database and store nothing but an ID as a cookie, but storing the actual values in name-value pairs is another way to do it (we'll discuss later why this approach has lost favor).

- E-commerce sites can implement things like **shopping carts** and **"quick checkout" options**. The cookie contains an ID and lets the site keep track of you as you add different things to your cart. Each item you add to your shopping cart is stored in the site's database along with your ID value. When you check out, the site knows what is in your cart by retrieving all of your selections from the database. It would be impossible to implement a convenient shopping mechanism without cookies or something like them.

In all of these examples, note that what the database is able to store is things you have selected from the site, pages you have viewed from the site, information you have given to the site in online forms, etc. All of the information is stored in the site's database, and in most cases, a cookie containing your unique ID is all that is stored on your computer.

An Example

To give you a simple example of what cookies and a database can do, we have created a simple history and statistics system for this article. This system runs on the **HowStuffWorks** servers and lets you view your activity on the HowStuffWorks site. Here's how it works:

- When you visit **HowStuffWorks** for the first time, the server creates a unique ID number for you and stores a cookie on your machine containing that ID. For example, on the machine I am using now, this is what I see in the HowStuffWorks cookie file:

-
  ```
      user    35005    www.howstuffworks.com/
  ```

  There is nothing magic about the number 35,005 -- it is simply an integer that we increment each time a new visitor arrives. I was user number 35,005 to come to the HowStuffWorks site since this cookie system was installed. We could make the ID value as elaborate as we desire -- many sites use IDs containing 20 digits or more.

- Now, whenever you visit any page on HowStuffWorks, your browser sends your cookie containing the ID value back to the server. The server then saves a record in the database that contains the time that you downloaded the page and the URL, along with your ID.
- To see the history of your activity on HowStuffWorks, you can go to this URL on the site:

  http://computer.howstuffworks.com/history.php

  Your browser sends your ID value from the cookie file to the server along with the URL. The **history.php** page runs a piece of code that queries the database and retrieves your history on the site. It also calculates a couple of interesting statistics. Then it creates a page and sends it to your browser.

Try the URL for the history page now:
http://computer.howstuffworks.com/history.php
Then go view a couple of other pages on HowStuffWorks and try it again. You will see that the statistics change and so does the list of files. (Also note that the HowStuffWorks Registration System allows you to reset your history list whenever you like.)

Problems with Cookies

Cookies are not a perfect state mechanism, but they certainly make a lot of things possible that would be impossible otherwise. Here are several of the things that make cookies imperfect.

- **People often share machines** - Any machine that is used in a public area, and many machines used in an office environment or at home, are shared by multiple people. Let's say that you use a public machine (in a library, for example) to purchase something from an on-line store. The store will leave a cookie on the machine, and someone could later try to purchase something from the store using your account. Stores usually post large warnings about this problem, and that is why. Even so, mistakes can happen. For example, I had once used my wife's machine to purchase something from Amazon. Later, she visited Amazon and clicked the "one-click" button, not realizing that it really does allow the purchase of a book in exactly one click.

  On something like a Windows NT machine or a UNIX machine that uses **accounts** properly, this is not a problem. The accounts separate all of the users' cookies. Accounts are much more relaxed in other operating systems, and it is a problem.

  If you try the example above on a public machine, and if other people using the machine have visited HowStuffWorks, then the history URL may show a very long list of files.

- **Cookies get erased** - If you have a problem with your browser and call tech support, probably the first thing that tech support will ask you to do is to erase all of the temporary Internet files on your machine. When you do that, you lose all of your cookie files. Now when you visit a site again, that site will think you are a new user and assign you a new cookie. This tends to skew the site's record of new versus return visitors, and it also can make it hard for you to recover previously stored preferences. This is why sites ask you to **register** in some cases -- if you register with a user name and a password, you can login, even if you lose your cookie file, and restore your preferences. If preference values are stored directly on the machine (as in the MSN weather example above), then recovery is impossible. That is why many sites now store all user information in a central database and store only an ID value on the user's machine.

If you erase your cookie file for **HowStuffWorks** and then revisit the history URL in the previous section, you will find that **HowStuffWorks** has no history for you. The site has to create a new ID and cookie file for you, and that new ID has no data stored against it in the database. (Also note that the HowStuffWorks Registration System allows you to reset your history list whenever you like.)

- **Multiple machines** - People often use more than one machine during the day. For example, I have a machine in the office, a machine at home and a laptop for the road. Unless the site is specifically engineered to solve the problem, I will have three unique cookie files on all three machines. Any site that I visit from all three machines will track me as three separate users. It can be annoying to set preferences three times. Again, a site that allows registration and stores preferences centrally may make it easy for me to have the same account on three machines, but the site developers must plan for this when designing the site.

  If you visit the history URL demonstrated in the previous section from one machine and then try it again from another, you will find that your history lists are different. This is because the server created two IDs for you, one on each machine.

There are probably not any easy solutions to these problems, except asking users to register and storing everything in a central database.

When you register with the HowStuffWorks registration system, the problem is solved in the following way: The site remembers your cookie value and stores it with your registration information. If you take the time to login from any other machine (or a machine that has lost its cookie files), then the server will modify the cookie file on that machine to contain the ID associated with your registration information. You can therefore have multiple machines with the same ID value.

Why the Fury Around Cookies?

If you have read the article to this point, you may be wondering why there has been such an uproar in the media about cookies and Internet privacy. You have seen in this article that cookies are benign text files, and you have also seen that they provide lots of useful capabilities on the Web.

There are two things that have caused the strong reaction around cookies:

- The first is something that has plagued consumers for decades. Let's say that you purchase something from a traditional mail order catalog. The catalog company has your name, address and phone number from your order, and it also knows what items you have purchased. It can **sell your information** to others who might want to sell similar products to you. That is the fuel that makes **telemarketing** and **junk mail** possible.

  On a Web site, the site can track not only your purchases, but also the pages that you read, the ads that you click on, etc. If you then purchase something and enter your name and address, the site potentially knows much more about you than a traditional mail order company does. This makes **targeting** much more precise, and that makes a lot of people uncomfortable.

  Different sites have different policies. HowStuffWorks has a strict privacy policy and does not sell or share any personal information about our readers with any third party except in cases where you specifically tell us to do so (for example, in an opt-in e-mail program). We do aggregate information together and distribute it. For example, if a reporter asks me how many visitors HowStuffWorks has or which page on the site is the most popular, we create those aggregate statistics from data in the database.

- The second is unique to the Internet. There are certain infrastructure providers that can actually create cookies that are visible on multiple sites. DoubleClick is the most famous example of this. Many companies use DoubleClick to serve ad banners on their sites. DoubleClick can place small (1x1 pixels) GIF files on the site that allow DoubleClick to load cookies on your machine. DoubleClick can then track your movements across multiple sites. It can potentially see the search strings that you type into search engines (due more to the way some search engines implement their systems, not because anything sinister is intended). Because it can gather so much information about you from multiple sites, DoubleClick can form very rich **profiles**. These are still anonymous, but they are rich.

  DoubleClick then went one step further. By acquiring a company, DoubleClick threatened to link these rich anonymous profiles back to name and address information -- it threatened to personalize them, and then sell the data. That began to look very much like spying to most people, and that is what caused the uproar.

  DoubleClick and companies like it are in a unique position to do this sort of thing, because they serve ads on so many sites. **Cross-site profiling** is not a capability available to individual sites, because cookies are site specific.