Empowering Users Against SideJacking Attacks

Ryan D. Riley
Department of Computer Science and Engineering
Qatar University
Doha, Qatar
ryan.riley@qu.edu.qa

Nada Mohaamed Ali
Department of Computer Science and Engineering
Qatar University
Doha, Qatar

Kholoud Saleh Al-Senaidi
Department of Computer Science and Engineering
Qatar University
Doha, Qatar

Aisha Lahdan Al-Kuwari
Department of Computer Science and Engineering
Qatar University
Doha, Qatar

ABSTRACT

SideJacking occurs when an attacker intercepts a session cookie and uses it to impersonate a user and gain unauthorized access to a web-based service. To prevent SideJacking, a server should enable HTTPS and configure all session cookies to only be transmitted over a secure link. Many websites do not do this, however, and the user may be unaware. In this work we present a Firefox extension that will allow users to quickly and easily determine whether the server they are visiting is susceptible to SideJacking attacks.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection (e.g., firewalls)*; K.4.4 [**Computers and Society**]: Electronic Commerce—*Security*

General Terms

Security

1. THE PROBLEM

A SideJacking [1] attack occurs when an attacker intercepts a session cookie and uses it to impersonate a different user of a web based service.

Typically, when a user logins in to a website on the Internet they enter their username and password into a web form and transmit it to the server. After the password is verified, the server generates a large random number and sends it to the user's web browser as a session cookie: a credential that the browser will then send back to the server in order to authenticate itself when making any future requests. In an ideal world, all communication between the client and the server would occur over an encrypted connection in order to protect it from the view of an attacker. In the real world, however, encrypted connections require significantly more processor power than normal connections, and many websites transmit only the password over an encrypted connection and use an unencrypted connection for everything else. In this case the session cookie is used by the client to authenticate itself to the website over the unencrypted connection. In this scenario the attacker would not be able to sniff the password, but he would be able to sniff the session cookie. Even if the connection is secure, an active attacker may be able to force an insecure connection that will cause the session cookie to be sent unencrypted. Once the attacker has the session cookie they can use it to access the website as if they were the legitimate user. This is called a SideJacking [1] or cookie hijacking [2] attack.

As an example, consider a user accessing their Hotmail account while an attacker is sniffing their connection at a coffee shop that provides free Wi-Fi. When the user connects to the Hotmail server in their web browser, Hotmail presents them with a login page that transmits their username and password over an HTTPS connection. When the user logs in, the attacker is unable to learn their username or password. After the login succeeds, the Hotmail server generates a random session cookie and sends it to the user's web browser over the secure connection. Next, the server instructs the browser to use an unencrypted connection to access the inbox. The browser initiates the unencrypted connection, sending the session cookie in order to prove it has logged in, and is given access to the inbox. Because the connection is unencrypted, the attacker is able to see the session cookie and save a copy of it. The attacker then initiates a connection to the Hotmail server, sends the session cookie it stole from the user, and is given access to the user's inbox despite not knowing the user's password.

A major problem with SideJacking attacks is that if a website does not ensure that the session cookies are only sent over encrypted connection, a user's only real recourse for preventing the attack is to simply not use the service. The burden of responsibility is on the website to provide protection against this sort of attack. Despite the fact that SideJacking attacks have been widely known since 2007, many major websites (and many more small ones) are still susceptible to the attack. Still worse, most users are completely ignorant of the fact that the website they are using may be susceptible.

2. OUR WORK

In this work, our goal is to allow users to quickly and easily ascertain if websites they use are susceptible to SideJacking

SIGCOMM'10, August 30–September 3, 2010, New Delhi, India.
ACM 978-1-4503-0201-2/10/08.

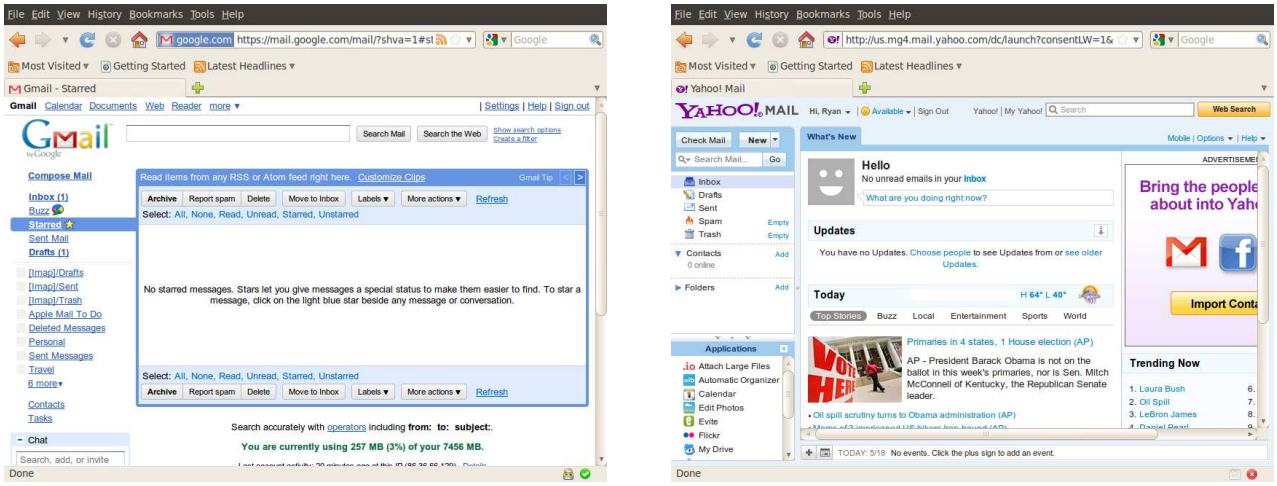

Figure 1: Screenshots of the extension is use. Notice the green checkmark and the red x in the lower right.

attacks. To accomplish this we will first develop a set of heuristics for determining if a given website is susceptible to SideJacking. Next we will develop an extension for the Firefox web browser that uses our heuristics to rate every website visited by a user and provide a graphical indication of whether or not the website is susceptible. With such an extension available users will be able to make informed decisions about the security of web services they choose to use. In addition, websites that are known to be insecure may be incentivized to invest in their infrastructure to provide encrypted connections and increase the security level for all of their users.

2.1 Heuristics

At first glance, the heuristic to determine SideJacking-ability seems trivial: If the connection is encrypted than the connection cannot be SideJacked (the session cookie would be encrypted) and if the connection is not encrypted than SideJacking is possible. In reality, however, things are more complicated.

The first complication is that some websites will permit both encrypted and unencrypted connections, meaning that an active attacker may be able to force an encrypted connection to become unencrypted (even only temporarily) in order to steal the session cookie. In order to determine if this sort of attack can occur, one option is to check the secure flag[1] of the session cookie to determine whether or not the cookie can be sent unencrypted. If it can, then we can assume the connection can be SideJacked.

The second complication is that a single website may have many cookies associated with it, and determining which one (or ones) is the crucial session cookie is not obvious. For example, when using Google's Gmail there are 10 cookies that are sent, and 8 of them appear to be randomly generated values. (Meaning they could potentially be session cookies.) Some of these cookies have the secure flag set, others do not. How do you decide whether or not the session cookie can be stolen if you aren't sure which one is the session cookie?

In our current implementation, we use a simple heuristic: A connection cannot be SideJacked if the connection is encrypted and at least one of the cookies sent has the secure flag set. This heuristic is far from optimal. While it will not provide any false positives (saying that a site can be SideJacked when it can't), it may provide false negatives by declaring a site safe even though its session cookie might not be set secure. Studying and refining the heuristics is our current work.

[1]The secure flag is set on the cookie by the server and instructs the client to only send the cookie over a secure connection such as HTTPS.

2.2 Firefox Extension

A preliminary Firefox extension has been built that uses the heuristic stated above to check the status of every website visited and report it to the user using simple icons. See Figure 1 for screenshots. The source code will be released soon.

3. REFERENCES

[1] Robert Graham. Errata Security: SideJacking with Hamster. `http://erratasec.blogspot.com/2007/08/sidejacking-with-hamster_05.html`.

[2] Mike Perry. CookieMonster: Cookie Hijacking. `http://fscked.org/projects/cookiemonster`.