Three Tier Verification Technique to Foil Session Sidejacking Attempts

Vinay Kumar
CSE Department, Manipal Institute of Technology
Manipal University
Manipal, Karnataka, India
waytowinay@gmail.com

***Abstract*— Session sidejacking is the term used to describe the theft of cookies used to authenticate the user to a web server. Session sidejacking attack is a form of session hijacking where the hacker uses packet sniffers to listen to the traffic between the client and server to steal the session cookie. Most of the websites use Hypertext Transfer Protocol Secure (HTTPS) only for Login purpose in order to protect the user name and password. However they revert back to Hypertext Transfer Protocol (HTTP) after this and all the cookies which are used to authenticate the user are sent to the server over an unsecure HTTP connection. Any hacker listening to this network using a packet sniffer can copy the cookies and use them to impersonate the victim and carry out activities on his behalf. Though the hacker won't know the password of the victim but he can still act on victim's behalf.**

A three tier session verification technique which is impervious to Session sidejacking is being proposed here. This technique allows the use of HTTP protocol and still protects the users from session sidejacking, however it assumes that the server uses a secure HTTPS connection for login purposes to avoid transmission of password in the clear. This technique uses a feature of Hyper Text Markup Language Version 5 (HTML5) called "local storage" to overcome the vulnerabilities of cookies and it foils any attempt to sidejack a session. This technique can be implemented using server side logic and client-side JavaScript.

Keywords - Session Hijacking; Session Sidejacking; Vulnerability of Cookies; eavesdropping; web session; local storage; Secret key;

I. INTRODUCTION

A. Why do we need cookies?

Cookies are used to maintain the state of a session between a server and a client. They can be used for authentication, storing site preferences, shopping cart contents or anything else that needs to be stored on the user's computer. A web server stores a cookie containing a unique session identifier plus a few other authentication details on the client's computer. With each subsequent request the web browser sends all the cookies along with the request. These cookies are used to identify the user.

B. What makes Wi-Fi networks vulnerable?

Wi-Fi networks are all around us. Most people have no idea that HTTP data is transmitted in form of clear text over a wireless network. This endangers the privacy of the users. This also makes Wireless network vulnerable as any hacker can use packet sniffers to capture user data and authentication information.

C. How is a web session hijacked?

The state of a session is maintained using cookies. Cookies are also used to identify the user. When a hacker listens to a Wireless network using packet sniffers, he can copy the cookies information transmitted along with the data packets. The hacker can simply save those cookies on his browser and effectively impersonate the victim and carry out activities on his behalf. The hacker has full access to the victim's account without knowing his username or password.

II. EXISTING SOLUTIONS

A. Use of HTTPS protocol

Websites can curb session sidejacking attacks by providing all their services through HTTPS. Websites like https://paypal.com offer their services on an encrypted connection because they cannot afford to lose any financial information to hackers. In such a situation a hacker cannot carry out any session sidejacking attack as all the data is being transmitted over an encrypted network and hence the packet sniffers are powerless.

B. Other Techniques

Some website store the session information in more than one cookie thereby making sure that the theft of a single magic cookie does not expose their website to risk. However the work around to this is trivial, the hacker can simply copy all the cookies and save them on his browser using a cookie editor browser Plug-in. Now when the hacker opens the website, all the stolen cookies that were associated with that website are sent to the server. No matter how the website distributes the session information among the cookies, when

978-1-4577-1088-9/11/$26.00 ©2011 IEEE

presented with all the cookies the server will allow access to the victim's account.

Also worth mentioning is the technique of associating a web session with the client's IP address. This technique is successful to some extent as it eliminates the possibility of anyone having a different IP address access the victim's account using his stolen cookies. But within a Wi-Fi network the hacker might be sharing the IP address with the victim. So this technique does not rule out the possibility of a session being sidejacked, but it sure does rule out rest of the IP addresses in the world getting a chance to misuse the stolen cookies. Also in certain cases people within a Wi-Fi network might share a small set of IP addresses thus not everyone on that Wi-Fi network can hack the victim's account.

III. INSPIRATION

Using HTTPS all the time is not a feasible solution as most of the websites don't really need to hide the data that is displayed to the users. However getting a user's session hijacked is still a concern as this cannot be permitted under any circumstance. Also implementation of HTTPS requires significant computational work to be done on the server side [1] [2]. Using the HTTPS protocol can be irritating as it provides a slow user experience. The problem gets aggravated for those with slow internet connection. Also HTTPS is not supported by websites which are hosted on Virtual hosts [1]. Virtual hosts are the web hosting providers which allow the web host to serve multiple websites from the same physical server.

So, the present conditions clearly demand the development of a technique which uses HTTP protocol for and is resistant to session sidejacking attacks.

IV. THE BASICS

The basic components of the Three Tier Verification Technique are explained below.

A. Secret Key

A secret key is a unique identifier which is generated by the server at the time of user login. The Secret Key is akin to the session key and the web application will require both the session key and the secret key to verify the authenticity of the user request

B. Storing the secret key on Client Browser

We need a way to store the Secret key on the client in such a way that the secret key persists across browser sessions, and is stored purely on the client side. Also it shouldn't be transmitted to the server along with each request just like it is done in the case of cookies. These facilities are provided by a new feature of HTML5 called Web Storage or Local Storage. Unlike cookies, which can be accessed by both the server and client side, Web storage falls exclusively under the purview of client-side scripting. Web storage data is not transmitted to the server along with every HTTP request [3]. Also web storage values cannot be modified by the user manually. The data written to local storage is scoped per origin, where *origin* is the combination of scheme, host, and port [3]. The page at http://example.com/localstorage.html can access local storage written by http://example.com/hello.html, but it can't access local storage written by http://www.microsoft.com/html5/sample.html. Nor can it access data written to local storage by https://example.com/secure.html.

Now we need a method to pass the secret key to the client so that it can be stored on the browser. We can achieve this in the following way.

When the user successfully logs into the website using the login page https://hello.com/login.html, then the web application stores a cookies called "key" on the client browser and its value field is the real key in encrypted form. Now the web server reverts back to HTTP and redirects the user to http://hello.com/home.html. Now the script on this page copies the key from the cookie named as "key" and stores it on the local storage and replaces the value in the cookies with a junk value like "xyz". This act of storing the value of the cookies "key" onto the local storage is to be performed only once when the user is redirected from login page to the homepage.

V. THREE TIER VERIFICATION

The three tier verification technique assumes that the website uses HTTPS connection for login process.

When the user successfully logs on to the website using the login page then the web application stores a cookie called "key" on the client browser and its value field is the real key in encrypted form. Now the web server reverts back to HTTP and redirects the user to the home page. Now the script on this page copies the key from the cookie named as "key" and stores it on the local storage and replaces the value in the cookies with a junk value like "xyz". This act of storing the value of the cookies "key" onto the local storage is performed only once when the user is redirected from login page to the homepage.

When the user visits a webpage, then the browser sends a user-agent string to the server hosting the website. This string contains information about the user's system. Typically user-agent information contains the name and version of the browser and version number of the operating system. Some additional information may also be a part of the user-agent string.

Also the IP address of the requesting client is sent along with each request.

Now the three tier verification technique makes use of the above mentioned three features to ensure that only the person

who actually logs into the website can view the user information.

Now the working of the three tier verification system is explained

A. Locking the session to Client IP address

When the user logs in, the server assigns a session variable to the user and stores a copy of the same on the client computer in form of a cookie.

Now as a part of our technique we want the server to lock the session variable to the client IP address. In the case of Wi-Fi many users surf the internet behind a Network Address Translator so that many users are effectively share the same IP address [1]. Hence from the server's point of view, there is no detectable difference between the victim and the attacker, and IP address binding is rendered useless [1].

However, in some cases (e.g. University Wi-Fi network) the Network address translation (NAT) system uses a small set of external IP addresses and maps to the set of internal IP addresses, so it is possible that some people in the same Wi-Fi network have a different IP address. So these people can be distinguished from the real user. Also it makes sure that no one else in the world using a different IP address can use the stolen cookies.

B. Locking the Session to UserAgent

We want the server to lock the session variable to the client IP address. This will make sure that a person using windows 7 or Firefox can be distinguished from a person who logged in using Windows XP or Internet Explorer. Also if the attacker is using the same Operating System and browser, but is using a slightly different version of the Browser, he can still be distinguished from the real user.

C. Locking the Session to Secret Key

The server generates a secret key at the time of login and associates it with the session variable. The secret key is unique for each user. This key is used to ensure absolute uniqueness for the person who actually logged in and created the session. A copy of the secret key is stored on the client side in the local storage.

D. Three Levels of Verification

1) Level 1 Verification

When a client request is received by the server, it compares the IP address of the requesting client to the IP address associated with the session cookie, if they match then the client is subjected to Level 2 verification.

2) Level 2 Verification

When the client is subjected to level-2 verification then the server compares the user agent of the requesting client with the user agent associated with the session. If the user agent configuration matches then it makes sure that the requesting client has the same IP address as well as same Personal Computer (PC) configuration as that of the real user. After this stage of verification is cleared, the client is subjected to level-3 verification.

3) Level 3 Verification

When the user is redirected to the home page or a request is sent to display the homepage then an empty page containing only java script code is loaded. The code extracts the value of secret key from the local storage and sends it to the server using the JavaScript function.

```
_doPostBack('serverFunc',secretkey)
```

Now the server will verify the secret key value and if the value of the secret key sent by the client is correct then the server displays the home page.

In this scenario cookies are necessary but not sufficient to validate the authenticity of a user request. An additional secret key is necessary to establish the authenticity of the requesting client. However cookies can easily be stolen and modified by the hacker. Now storing the secret value in the local storage makes sure that the hacker cannot steal the secret key by stealing the cookies. Also since there is no way to edit the local storage values, so its not possible to modify the secret key manually and hence the hacker cannot send a stolen secret key to the server. This ensures that the hacker cannot access the victims account.

VI. EFFECTS OF USER BEHAVIOR

A. Page Reload or refresh

Typical user might reload a page if the user does not get quick response from the server. This technique ensures that the rightful user still gets to view his page while an attacker gets redirected to login page.

B. Copying and Pasting the link in a new tab.

When a user pastes a link in the new tab, he still gets to see his page because the secret key is stored on the browser and the JavaScript code will still be able to extract it and send it to server for verification.

VII. LIMITATIONS

This technique suffers from two important limitations

A. JavaScript Requirement.

This technique will not work without JavaScript support, since JavaScript is required to handle the secret key. JavaScript is

code is used to extract and send the secret key from the client computer to the server.

B. Active Attacks.

This technique does not protect the user from active attacks in which the hacker can manufacture packets and send them to the server. This technique only addresses the "sidejacking" attack which is very easy to launch using packet sniffers.

REFERENCES

[1] Ben Adida. SessionLock: Securing web session against eavesdropping. WWW 2008 / Refereed Track: Security and Privacy - Web Client Security April 21-25, 2008 · Beijing, China

[2] http://webmonkey.com/2011/03/https-is-more-secure-why-isnt-the-web-using-it-today/

[3] Ian Hickson and David Hyatt. Html5. http://www.w3.org/html/wg/html5/

[4] Robert Graham. Sidejacking with Hamster, August 2007. http://erratasec.blogspot.com/2007/08/sidejacking-with-hamster_05.html.

[5] Whitfield Diffie and Martin E. Hellman. Newdirections in cryptography. IEEE Transactions on Information Theory, IT-22(6):644–654, 1976.

[6] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transport Protocol – HTTP/1.1, 1999.http://www.ietf.org/rfc/rfc2616.txt.

[7] Apache Software Foundation. SSL/TLS Strong Encryption FAQ – Apache HTTP Server.
http://httpd.apache.org/docs/2.0/ssl/ssl_faq.
html#vhosts2 last viewed on 1 November 2007.

[8] T. Berners-Lee, R. Fielding, and L. Masinter. Uniform Resource Identifier (URI): General Syntax, January 2005. http://ietf.org/rfc/rfc3986.txt

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

[10] Bill Venners. HTTP Authentication Woes, April 2006.
http://www.artima.com/weblogs/viewpost.jsp?thread=155252