A Novel Approach to Prevent Session Hijacking Attack

Darshan Tank
M.Tech Student
K J Somaiya College of Engineering
Vidyavihar East, Mumbai-400077

Ashwini Dalvi
Assistant Professor
K J Somaiya College of Engineering
Vidyavihar East, Mumbai-400077

ABSTRACT

Session hijacking is also called as cookie hijacking in which the attacker exploits a valid computer session sometimes also called a session key or session token to get an unauthorized access to user system or back-end server.so to prevent this type of attack we are creating a protocol that will prevent the attacker from gaining the access of encrypted cookie and back-end server. We are developing a Reverse proxy server (RPS) with a One Time Cookie (OTC) and generating a browser fingerprinting, IP address of system, session ID such that Reverse Proxy server handles a request using One Time Cookie (OTC) protocol to prevent adversary from capturing and injecting the session credentials also we are using Blowfish Algorithm for the encryption purpose. If any of this parameter alter than we can be easily identified the attacker.

Keywords

Session Hijacking, One Time Cookie, Reverse proxy server, Browser fingerprinting, session ID, IP address, Blowfish Algorithm, HTTP

1. INTRODUCTION

HTTP protocol is a Request-Response model. HTTP is an application protocol that runs on top of the TCP/IP suite of protocols (the foundation protocols for the Internet). first of all client sends a request (which asks for data) to the server and server sends back the response to its client since HTTP is a stateless protocol. This does not require the server to retain information or status about each user for the duration of multiple requests. However, when the application is built on the top of the HTTP protocol it needs to store every information of the state which is created during the session. Suppose one website (Shopping site) has a lot of visitors and it needs to keep track of every client and identify the client then it necessary to maintain the state, which can be done with the help of the cookies. Cookies are small files which are stored on a users computer. Generally, cookies are used to maintain the web session They are designed to hold a modest amount of data specific to a client and website and can be accessed either by the web server or the client computer. A Cookie is used for the authentication purpose HTTP cookie contain the small string files size is around 4kb.cookies data are less secure because it can transfer in the clear text form, so the adversary can capture the cookies and hijack the user session. Also, session hijacking attack is not the new, majority of the web application are still vulnerable to this type of attack.so to avoid this type of the attack most suggested method is to use the HTTPS protocol to secure the user session .HTTP protocol will be adding an additional layer of security to web application but use of the HTTPS protocol is the challenging task for the particular distributed system due to performance and financial concerns.so to avoid this problem Author has proposed a model in which they have created a reverse proxy server with the one time cookie approach where every request is made by the client is first to pass through the reverse proxy server they have use the blowfish as an encryption algorithm for the each request reverse proxy server collect the browser fingerprint, IP address and send this to the server after credential verification server response to reverse proxy server then it will set the OTC and session ID and send the request back to client. Client request with the OTC and RPS check the OTC and proceed the request. If anything changes then it will redirect the user to the login page.

2. LITERATURE REVIEWS

Web developers have periodically raised security concerns for session authentication cookies since their adoption. Burgers et al [1] It uses a server-side proxy that secures communication channel. It binds the application session with SSL/TLS session using HTTPS protocol and proxy server while disadvantage of the system is Hi-jacking is possible if both client and server are compromised and Doesn't support long living cookies for the highly distributed web application. [2] have demonstrated the various attacks possible with web authentication technique, including vulnerability to session hi-jacking attacks. As a result, security analysts have suggested ways to improve the robustness of cookies and session authentication. In addition, Juels et al. [4] have proposed the use of cache cookie; which is a different form of persistent state in the browser. But the problem with these mechanisms is that they still rely on static cookie token to authenticate requests. Thus, security analysts have also explored the use of dynamic cookies to prevent arbitrary reuse. Nikiforakis et al. At[6] It runs a software on the client computer and intercepts the Set-Cookie header before it is sent to the browser. Thus, cookies will never be in the browser, but This method relies on the client-side protection but doesn't protect the backend Server. Park et al. [3] suggested cryptographic cookie mechanisms that provide better confidentiality and integrity to web session. S.Jha and S.Ali in [5] they have use Tokens generation, session encryption and use of unique session ID but the disadvantage is Computation overhead. Spoofing and De authentication can still occur. The Author in [7] proposed a hybrid scheme that utilizes one-way hashing and sparse caching technique, but practically it is not possible to implement their research is focused on hashing, but they have not described how to prevent from the session hijacking attack.

3. PROPOSED SYSTEM OVERVIEW

Due to the problem with the mechanisms discussed in the literature survey that they still rely on static cookie token to authenticate requests. Thus, security analysts have explored the use of dynamic cookies to prevent arbitrary reuse. Hence, we proposed the prevention of session hijacking with OTC that makes use of RPS, OTC, IP address and Browser Fingerprinting to prevent session hijack. In this technique, the network layer and application layer are bound together in order to prevent the session credentials from hijacking. The technique uses a reverse proxy server that will generate session credentials. The session credentials include session

ID, browser ID, IP address and unique disposable one-time cookie (OTC). The session ID is generated by binding the application level and network level together to detect a change of machine for the current session. OTC and browser ID are again bounded with session ID so that it can detect the change of cookie or Browser for the current session. For each new session request, it will generate new disposable cookies.

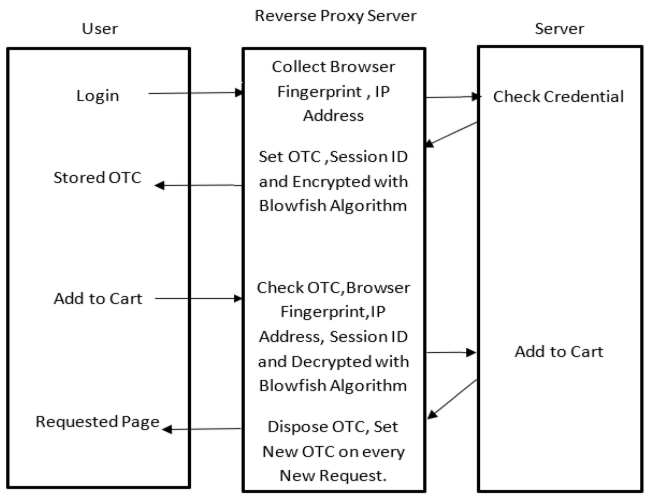
User
Reverse Proxy Server
Server
Login
Collect Browser Fingerprint , IP Address
Check Credential
Stored OTC
Set OTC ,Session ID and Encrypted with Blowfish Algorithm
Add to Cart
Check OTC,Browser Fingerprint,IP Address, Session ID and Decrypted with Blowfish Algorithm
Add to Cart
Requested Page
Dispose OTC, Set New OTC on every New Request.

Figure 1: Block Diagram of Proposed System

The components of the proposed system are as follows:

1) **User:** A user or client is the one who initiates a request. Suppose a user wants to purchase something, he will send request using which contains a user username and password, to the server this after successful authentication, the user will be given an OTC through which he/she will be authenticated for every request he/she makes. Each time a user sends a request, an OTC is sent along with the request this data is encrypted with the blowfish algorithm.

2) **RPS:** A proxy server is nothing but a computer that acts as an intermediary between endpoint devices. It is mainly used at the user or client side. However, instead of using a proxy server here at the client side, we use RPS on the server side. It is a type of proxy server that typically sits behind the firewall in private networks and directs client requests to the appropriate backend server. Thus, every request from the user has to pass through RPS. In this system, we use this server to combine browsers session ID and network IP address with a proxy server. The function of RPS is to obtain IP address, browser fingerprint, set OTC, and session ID, and for each incoming request. When a user requests, the proxy server will send the request to the user only if the user is validated by the system. When a request with the same application session ID of a genuine user is used with a different machine, the system will know that the session is stolen. Thus, it will remove the session cookie from the request, and the application session is invalidated.

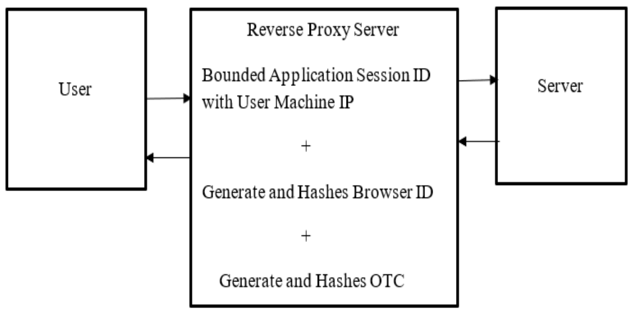
User
Reverse Proxy Server
Bounded Application Session ID with User Machine IP
+
Generate and Hashes Browser ID
+
Generate and Hashes OTC
Server

Figure 2: Flow Diagram of Reverse Proxy Server

3) **One Time Cookie:** One-Time Cookie is generated by the reverse proxy server for each request of the user. It provides more security than the alternative authentication cookies that do not require volatile state in the web browser. OTC is generated by the reverse proxy server for each request of the user and is disposed of once it is verified by the server. Thus, the legitimate user is verified without storing any volatile data. Browser fingerprint: A browser fingerprint is when, by visiting a website, that site can generate an ID (or fingerprint) that is unique to your computer. The fingerprint can then be sent to their server, and you can be tracked. No cookies required, no security holes required, no Do not track me setting can Make a difference. Device fingerprinting or browser fingerprinting is the systematic collection of information about a remote device, for identification purposes. Client-side scripting languages allow the development of procedures to collect very rich fingerprints: browser and operating system type and version, screen resolution, architecture type, lists of fonts, plugins, microphone, camera, etc.

4) **Server:** This is the actual server to which a request is sent by a client. The server checks credentials process all client's requests and responses to all clients.

The proposed system works as follows:

1) User enters his/her credentials.
2) Encrypted request is sent to RPS, which collects IP address and browser fingerprints from client and forwards the request to the server.
3) The server checks the credentials, processes the request, i.e., it fetches the requested page and sends it to the client, but before that, it passes through RPS.
4) RPS creates OTC, session ID and Encrypt the request using Blowfish Algorithm gives it to the client, and it also forwards the response.
5) User then stores OTC.
6) From now on for every request made by the user, the user sends OTC to RPS.
7) RPS checks it and again creates OTC for every new request made by the user.
8) RPS terminates the session if session ID, IP address, OTC, and browser fingerprint changes.

4. CONCLUSION AND FUTURE WORK

The risks associated with the use of cookies as session authentication tokens have been known for years. More robust alternatives have been proposed to replace authentication cookies; however, they have not been deployed because they fail to meet the requirements of highly distributed Web 2.0 applications. Specifically, most of the proposed alternatives require costly state synchronization across the web application, a serious concern for distributed systems. Thus, the proposed system is a new, updated general technique to detect and prevent session stealing that binds the session with network IP with the combination of Browser ID (fingerprinting) and One Time Cookie (OTC). A secured communication channel is used for authenticating the user on the server-side reverse proxy. Using a server-side proxy the secure communication channel is bound to the user authentication of the session. Also, browser fingerprinting enables identifying a given browser by constantly checking (e.g., with every request) if the browser is still behaving as it did when the session was started. Using this technique, the web application has effectively prevented session stealing attacks. This method is not dependent on the secure protocol to protect the network credentials; also, long-living cookies can also be used with our application. In this technique, OTC and browser ID, a secure alternative to authentication cookies. OTC is not only resistant to session hijacking, but also maintains the simplicity and improves the performance of the cookies. Moreover, browser ID will offer another security layer to web applications by reducing the threats associated with cookies. This technique has significantly improved the security of web sessions with minimum impact on scalability and performance.

Future Scope: In the presented work the system doesn't make use of HTTPS for transmitting the OTC. In the future, this technique can be improved by just incorporating the HTTPS protocol with the session credentials to handle the network connection.

5. ACKNOWLEDGEMENT

I am very thankful to my guide Prof. Ashwini Dalvi for her invaluable guidance and advice throughout this project.

6. REFERENCES

[1] Willem Burgers, Prevent Session Hijacking by Binding the Session to the Cryptographic Network Credentials, in Institute for Computing and Information Sciences, Radboud University Nijmegen, The Netherlands. 2013.

[2] C. Visaggio, Session Management Vulnerabilities in Todays Web, in IEEE Security and Privacy,48-56, 2010Tavel, P. 2007 Modeling and Simulation Design. AK Peters Ltd.

[3] J. S. Park and R. Sandhu, Secure Cookies on the Web, in IEEE Internet Computing, 36-44, 2000.

[4] A. Juels, M. Jakobsson, and T. Jagatic, Cache cookies for browser authentication (Extended Abstract), in IEEE Symposium on Security and Privacy, 2006.

[5] S. Jha and S. Ali , Mobile Agent Based Architecture to Prevent Session Hijacking Attacks in IEEE 802.11 WLAN, 5th Inter-national Conference on Computer and Communication Technology, 2014

[6] Nick Nikiforakis, Wannes Meert, Yves Younan, Martin Johns, and Wouter Joosen, SessionShield: Lightweight protection against session hijacking, in 3rd International Symposium Engineering Secure Software and Systems (ESSoS 2011), volume 6542 of Lecture Notes in Computer Science, pages 87-100. Springer-Verlag, 2011

[7] Alabrah, Amerah, and Mostafa Bassiouni. Preventing ses-sion hijacking in collaborative applications with hybrid cache-supported one-way hash chains. In Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), 2014 International Conference on, pp. 27-34. IEEE, 2014.

[8] Yassein, Muneer Bani, Shadi Aljawarneh, Ethar Qawasmeh, Wail Mardini, and Yaser Khamayseh. Comprehensive study of symmetric key and asymmetric key encryption algorithms. In Engineering and Technology (ICET), 2017 International Conference on, pp. 1-7. IEEE, 2017.

[9] http://cs.indstate.edu/ schinta/blowfish.pdf.

[10] https://www.greycampus.com/opencampus/ethical

hacking/session-hijacking-and-its-types.