Prevention of Session Hijacking and Authentication Providingtothe Session Cookie

Prasuna Kotturu, SyamPrajwalKammula, Sai Surya Bunga, Praneeth Sai Atluri

Abstract:*The current world is running around the word "Privacy". Every individual's aim is to secure their data and transactions so that no one can access them without proper authentication. In this digital era, all the data stored in the internet protected by a password. The general opinion is that a password can protect the data from being acquired by an unauthorized user. The issue is about what happens subsequently with an authorized login. Once we login into our account, all our actions, state of browser and timestamps are recorded in a simple text file known as "Cookie". In this paper, we proposed a mechanism which is easy to implement and robust in providing authentication to the session cookie. This obstructs an unauthorized user from getting access to our private data. Our mechanism provides authentication by using the concept of hashing combined with a unique identifier.*

Keywords: *Privacy, Authentication, Authorization, Hijacking, Hashing, Salt, Cookies.*

I. INTRODUCTION

Cookie is an important feature of today's internet which came to existence because of the stateless nature of HTTP protocol [3]. Cookie was created by Lou Montulli who called it "magic cookie" as it acts as a token having data exchanged between client and server [1]. Cookie made all the things like online shopping, targeted advertising, and personalized content possible. Cookie does not guarantee the confidentiality and integrity of the data present in it. There are mainly four features on which the security lies on, which are, Integrity, Confidentiality, Authentication and Authorization.Privacy of cookie is a paradoxical statement because cookie attains privacy by being encrypted and at the same time, it compromises the privacy of the user by sharing his details with third parties. In this digital era, there are many types of cookies each one having its own piece of work to be done. The different types of cookies are,

- First party cookies
- Session cookies
- Third party cookies
- Flash cookies
- HTTP Only cookies
- Secure cookies
- Zombie cookies, etc.

From all the above cookies, session cookie is the one responsible for the unique identification of interactions between a client and a server. Session is the collection of connections and interactions made between a client and server in a particular time frame [1]. This cookie maintains the information about these sessions hence it is called as session cookie. There are many fields in this session cookie out of which session id is one. Every session will be having a unique identifier called as session id. Let's say there are two users, user A and user B accessing the same server(website). Both the users will be involved in conversation with the server, but their conversations will never intersect. This is because of the unique session ids of the user sessions. As we all know that with great technologies comes the greater threats, although the session cookies are a very helpful piece of technology, they are vulnerable. A session id can easily be stolen by capturing the cookies. Once your session id is out, the attacker can use your account without knowing your login credentials. He can send the requests to the server as if you are sending them. Attackers create a scene where the attacker virtually takes your place in communication with the server.

A. Cryptography

Cryptography is the technique involving encryption and decryption. The word cryptography can be split as crypt + graphy which means hidden + writing. Main purpose of cryptography is to make sure that the data is visible to the user to whom it is to be visible. This makes the data useless for the attacker even if he gains access to it illegally.

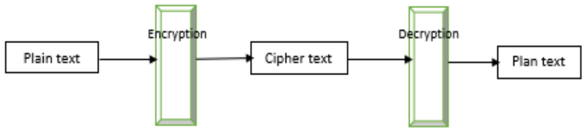
Manuscript published on November 30, 2019.
* Correspondence Author

PrasunaKotturu*, Assistant Professor, Department ofComputer Science and Engineering, KoneruLakshmaiah Educational Foundation, Guntur, A.P, India.

SyamPrajwalKammula, Student, Department of Computer Science and Engineering, KoneruLakshmaiah Educational Foundation, Guntur, A.P, India.

Sai Surya Bunga, Student, Department of Computer Science and Engineering, KoneruLakshmaiah Educational Foundation, Guntur, A.P, India.

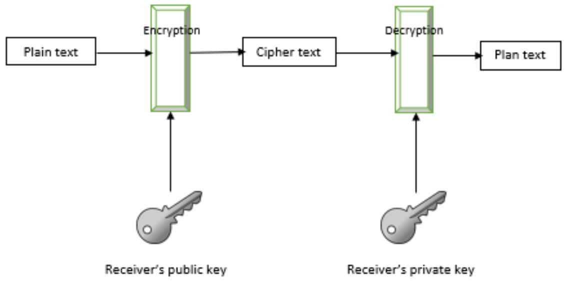
Praneeth Sai Atluri, Student, Department of Computer Science and Engineering, KoneruLakshmaiah Educational Foundation, Guntur, A.P, India.



The process of cryptography can be explained in detail by using the above diagram. The original message called as plain text is encrypted using a key to generate the cipher text which will be transmitted to the receiver. The receiver will decrypt the cipher text using the key according to the algorithm used for encryption to get the plain text.

B. Public Key Cryptography

Asymmetric cryptography or Public-key cryptography uses pairs of keys which are public keys which may be disseminated widely, and private keys which are known only to the owner. The production of such keys depends on cryptographic algorithms based on mathematical problems to produce one-way functions. Effective security solely needs keeping the personal key private; the general public key is often overtly distributed while not compromising security. Within these systems, anybody can encrypt a message using the receiver's public key, but that encrypted message can only be decrypted with the help of receiver's private key.



C. Hashing

Hashing is a cryptography method that converts any form of data into a unique string. Any data can be hashed, no matter its size or type. Generally hashing, regardless of the data's size, type, or length, the hash that any data produces are always the same length. It is a one-way function, you can put data into a hashing algorithm and get a unique string, but if we come upon a new hash, we cannot decipher the input data. Hashing is easy to perform, but it is extremely difficult to reverse it. (The difference between hashing and encryption is that encryption can be reversed, or decrypted, using a specific key.) The most widely used hashing functions are MD5, SHA1 and SHA-256. Few of the hashing processes are significantly harder to crack than others.

D. SHA-1 Hashing

Any Secure Hash Algorithm 1 is a hash function that will accept input of any size and produces an output of size 160-bits as hash value. The output value after hashing is referred to as a message digest. The message digest is then rendered as a hexadecimal number which is 40 digits long. National Security Agency (NSA) of the United States of America designed SHA1.

E. Session Hijacking

Session hijacking is also called as cookie hijacking. This is a process of gaining control over a valid computer session by an unauthorized user. In this session hijacking, the attacker will be stealing the cookie (i.e., session cookie) which is used authenticate a user to a web server. By using an intermediary system or with access to the saved cookies, the attacker can easily steal HTTP cookies which are used to maintain a session on many web sites. So, this has some connection with the web developers. If the session cookie gets into the hands of an attacker, he might use the "Pass the cookie" technique to perform session hijacking.

F. Wireshark

Wireshark tool is an open-source packet analyzer. Wireshark is used for network troubleshooting, analysis, software package and communication protocol development, and education. Originally named as Ethereal, it was renamed as Wireshark in 2006 due to trademark problems. Wireshark is a cross-platform tool and it uses the Qt widget toolkit in current releases to implement its user interface and using pcap to capture packets. It runs on Linux, macOS, BSD, other Unix-like operating systems, and Windows. Wireshark, and the other programs distributed with it, are free software, released under the terms of the General Public License.

G. Network Sniffer

Network sniffer takes snapshot copies of the data flowing over a network without redirecting it. Some sniffers work only with TCP/IP packets, but the more sophisticated tools work with many other network protocols and at lower levels, including Ethernet frames. In the past, sniffers were tools used exclusively by professional network engineers. Nowadays, however, with software available for free on the web, they are also popular with internet hackers and people who are curious about networking.

H. Salt

It is a random string that enhances security and it is an additional input to a function that hashes data or a password. They are used to protect the passwords in storage.Historically a password was stored in plaintext on a system, but over time additional safeguards developed to protect a user's password against being read from the system.

Users	Passwords	Hashed passwords with salt
User1	Password123	qwertyuiop123as
User2	Password123	asdfghjklop951h

II. LITERATURE REVIEW

LaCroix et al [1] gave a picture about cookie, who invented it, the way cookies work and the way a cookie can be stolen from a user to gain illegal access to accounts. He also explained about various mitigation techniques for end-users.

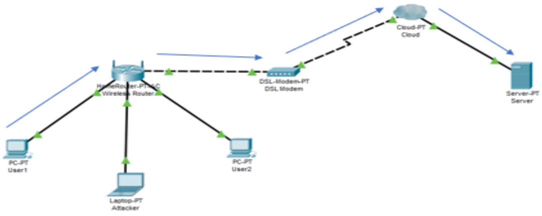
This paper demonstrated a situation having an attacker, a victim and a situation of cookie stealing. Li et al [2] analyzed the working methodology of Cookies, proposed the method of making cookies against the security threats. He also presented various options to meet the security constraints. Various security threats caused by the cookies are explained in this paper. Later on, concepts like confidentiality, integrity and at last the cookies security solutions like to strengthen the security of transmission between site cookies, to give strength to the security of information about cookies stored on the site and refuse them and resist the cookies monitoring user's behavior are discussed. Park et al [3] discussed about secure cookies on the web in his paper. Security concerns and security threats of cookies are explained. Providing user authentication, integrity and confidentiality using cookies is discussed. Applications of secure cookies like User Authentication, Electronic Transactions (e-Commerce), Pay-Per-Access and Attribute-Based Access Control are listed in this paper. Alex X. Liu1 et al [4] proposed a protocol which provides security to the cookies as all the security features like integrity, confidentiality, anti-replay and authentication are satisfied. The proposed protocol is very efficient and indeed deployed easily. This protocol is called efficient as it ensures all the security services. In Fu's Cookie Protocol, a server issues a secure cookie to a client having the following fields in the value field of the HTTP cookie. Server key is the only key involved in the Fu's protocol. Encryption of the data field present in every cookie is done by using this server key; but security is not achieved by this solution. This problem can be solved in a simple and efficient way. HMAC (username|expiration time, sk) is used as the key for encryption. Replay attacks can be done to the Fu's protocol. To solve this problem, author propose that we can add the SSL session key to the keyed-hash message authentication code of a cookie. In simple terms, HMAC (username|expirationname|data|session key, sk) is to be used as the authentication code of each cookie. As we use the same server key for all cookies in the Fu's protocol, it is easy to perform volume attacks. By using an encrypted key as proposed by the author, we can provide a scalable and efficient solution for the problem. Wei-Bin Lee et al [5] used a self-verification mechanism to propose an efficient, secure, and practical system of cookies. Cookie verification, simple key management, symmetric encryption and no sensitive verification table on the server are the features of the self-verification mechanism adopted by the author. Digital signature consists of a secret parameter produced by the server with the help of this self-verification mechanism. This parameter is checked by the web server and used to decide on the decryption key of a cookie. The web server does not have to maintain a sensitive verification table because the secret parameter used in this protocol is kept as secret by using the concept of digital signature. By using this proposed scheme, we can reduce the computational load on the server as this scheme provides high security and reduces cost at the web server side. Because of this reduced load, the author believe that this scheme provides more benefit when compared to the existing schemes when used for social networking sites, online shopping, and cloud services. Guy Pujolle et al [6] proposed a cookie mechanism which is secure as it is based on reverse proxy implementation. This mechanism makes sure that there is an end to end secure connection between client and servers in a network. The proposed mechanism provides integrity control, confidentiality, no-replay and source authentication to the cookies. This proposed secure cookie mechanism uses cryptography functions. All the cookie attributes to be protected are kept in a message M which is given as input for the HMAC function along with a secret key generated by reverse proxy and a counter value. sessionID is one of the attributes present in the message M which is used to uniquely identify a user session. To achieve source authentication, anti-replay cookies and integrity control, we integrate the counter value to the sessionID as per the secure cookie mechanism. User session management, end to end secure session and transparency are ensured by this design of the solution. Yitao Yao [7] in this paper described about systems and methods for using cookies to verify the origins of web related request. His proposal was that the systems and methods use a client identification value that may be sent from a client to a server. The server uses the client identification value to determine that the origin of the request matches the origin of previous requests so that personalized or other private data is not improperly sent to the wrong client. Praveen Gauravaram [8] in this paper focused mainly on the importance of protection of passwords. In this paper, the author recommended a hash function which is a combination of the salt and password to prevent the attacks such as birthday and dictionary attacks. The author presented the first security analysis of this salt||password hash function. In this paper author also made research on password||hash function and found a security gap between prefix-salt and suffix-salt methods of hashing passwords. In this paper, the author demonstrated the subtle property of this hashing application by performing a precomputed offline birthday attack and resulted in finding the possibility of building multiple passwords for an unknown password for same hash value and salt. Mohamad Badra et al [9] illustrated about the authentication provided by security protocols to prevent replay attacks and how shared secret keys and public key infrastructure are used to generate new keys for each session. In this paper, authors explained about how Challenge- Response authentication mechanism works and drawbacks of basic Challenge-Response authentication. The authors proposed a new semantic meaning for the challenge that helps in reducing identity assumption attacks. The extension proposed by the authors is completely compatible in backward manner and will continue the basic authentication process. Young-II Kim [10], in this paper proposed a MAC address-based communication restricting method and explained the mechanism of this method. In this method upon receiving a request for communication packet data is received which consists of a MAC destination address and a MAC source address. The access vectors of the MAC addresses are present in an address entry table is verified, and access if the security keys of the MAC destination and source addresses are not matched then the access is denied. Zheng Qi [11], in this paper provided an architecture for an authentication engine for SHA1 multi- loop or multi-round authentication algorithms to increase the speed at which data packets are transferred over computer network.

According to the author, this architecture can be used in combination with data encryption and protocols. The author also suggested that this application can also be used in combination with applications that does not perform data encryption or decryption. In this paper, the author also explained about the increase in the performance of processing of short data packets with this authentication engine. Swain et al [12] presented a technique which made use of steganography (which is the art of hiding a file or a message inside another file or a message) and cryptography to achieve secrecy in communication.The cryptographic algorithm used in this paper is a block cipher.

P. M. V. K. Sandeep at al [13], proposed a mechanism to reduce the chances of attack on Mobile as-hoc Networks. Through this paper, they stated some privacy requirements which are related to attackers in MANET. This mechanism adds the security feature of digital signature with Enhanced Adaptive Acknowledgement method to provide more trust with its performance. The experimental results of this mechanism are also found to be satisfactory. Gandharba Swain et al [14], through this paper proposed a classification of security techniques in network such as authentication, non-repudiation and secrecy. The secrecy techniques come under two categories: cryptography and steganography. Steganography also is a very suitable method to achieve secrecy in communication process. When both cryptography and steganography are utilized then the process becomes double secured. All the major techniques on image steganography proposed by different researchers are discussed in this paper very briefly. The techniques on audio steganography are also briefly discussed. A hybrid model is then shown using both cryptography and steganography techniques. N. Kusuma et al [15], in this paper proposed the methods to prevent attacks on digital signatures. Cryptography contains concepts about how to protect data in the digital form and to give security services. The primary task of a signature is to provide the means for any entity to bind its private identity information. The various attacks done on the digital signature were verified. RSA signature scheme was the method used and it remains till today as one of the most versatile techniques which are present. Fiat-Shamir signature schemes, DSA and related signature schemes are among the two other methods which were verified.

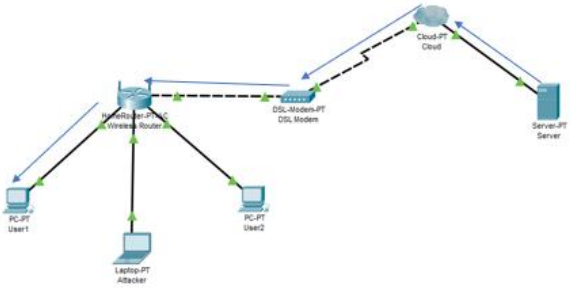
III. PROBLEM STATEMENT

Every session has a unique session id and all these session ids are very easy to be stolen., it is very important to safeguard these session ids from attackers in order to have a secure data exchange. These days, after the introduction of online video streaming websites like Amazon prime, Netflix, Hotstar, etc., it has become very common among the people to share a single account with many others. This can be done by sharing a single session id with others so that everyone can use a single session i.e., a single account even if the people using it are geographically located in different places. This is just an example of taking advantage of the vulnerability present in the current cookie management system. This vulnerability has become a common point of interest for many attackers in order to gain access to user accounts to which they are not authorized to enter. Banking, fund transfer, e commerce, etc. are being affected because of this vulnerability. The attacker can use any network sniffing tools like raw cap, Wireshark, etc. to capture the packet containing the session cookie in which the session id is present. The process of steeling the cookie is explained using the diagrams below.



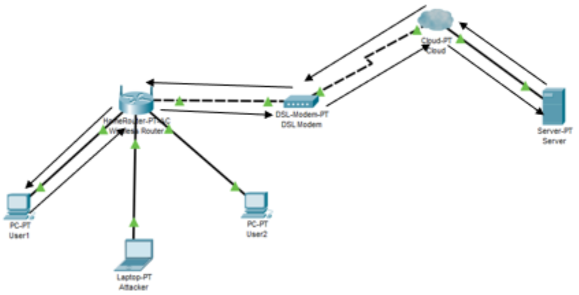
(a) User1 request for connection.

Initially, User requests the server to start a session in order to establish communication with it.



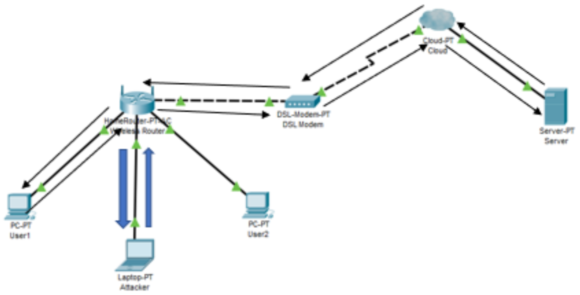
(b) Connection established and server sends a packet with cookie in it to user1.

The server responds with a packet as response to the request sent by the server. This packet consists of several cookies out of which session cookie is one. This session cookie has the unique session id which will keep track of all the communication done in that session.



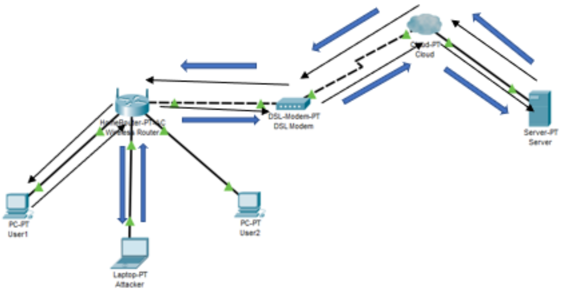
(c) Communication between server and client.

Now, the user and the server can interact using the session id till the cookie expires.



(d) Interference of attacker.

The attacker captures the session cookie using any of the network sniffing tools and gain the session id using which he will bypass the login process to access the account of user1. By doing this, server will be getting requests from both user1 and attacker with same session id hence it can't identify the difference between a legitimate user and an illegitimate user.

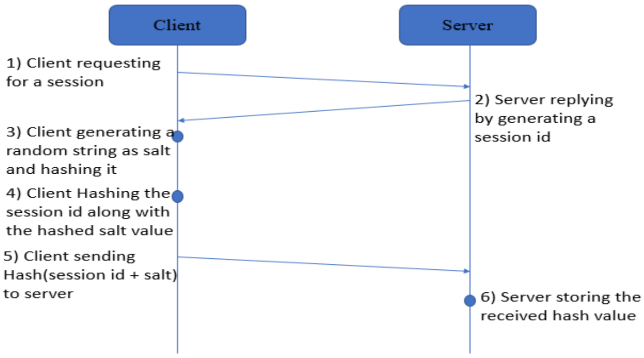


(e) **Attacker accessing the user1's account**.

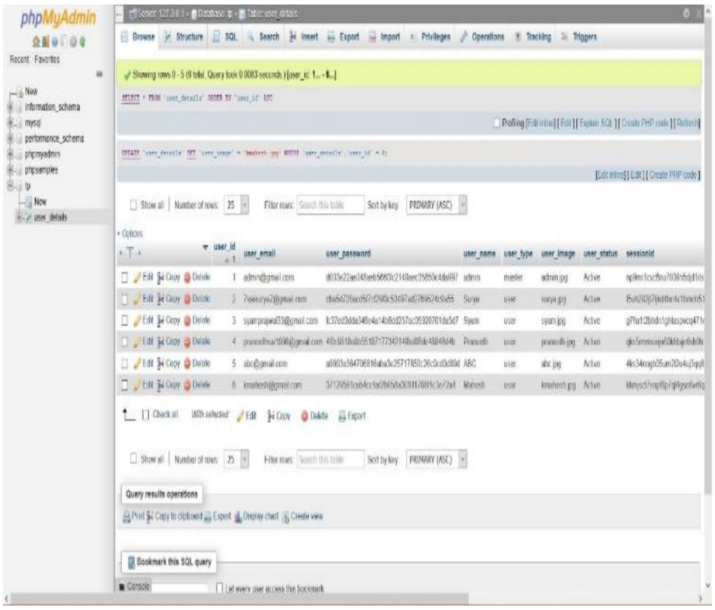
The attacker and user1 accessing the account (i.e., interacting with the server) with a same session id.

IV. PROPOSED METHODOLOGY

As we all know that salt is a random string that enhances security, we want to use the concept of Salt by concatenating it to the Session ID that has been generated during the Session establishment. The result of the concatenation is hashed using hash algorithms like SHA-256, MD-5 etc. We call this hashed value as randomized session hash. This should be only known to the client and the server. So, at the beginning of the session establishment the client generates a random string and shares it with the server in a secure way then the server generates a randomized hash with help of the random string that is sent by the client ,the client sends this random string in a secure way by encrypting it with algorithms like AES, DES, RSA etc. This message can be sent by using protocols like SSH (Secure Session) or Telnet. Initially the server establishes an SSH connection with the client with the help of IP address that can be identified when the client is connected to the server. So that this randomized hash is shared only between client and server. When multiple clients are connected to the server, multiple randomized hashes will be generated each of the randomized hashes will be unique and the length of the randomized hash depends on the hash algorithms that is being used. When a client establishes a secure connection with the server, client generates a random string and sends it to the server in a secure way. Then the server generates a session ID and produces the randomized hash, so when an attacker can obtain the session ID by capturing cookies with the help of tools like Wireshark, Raw cap. If it is a HTTP cookie there is no need of any decryption, the attacker can easily see the Session ID and HTML content that is exchanged between client and server. By using tools related to MIMT (Man in the middle attack) he can also tamper the cookie which causes a lot of damage to both client and server. If the request is HTTPS, then all the content and cookie will be in an encrypted format even though they can be easily decrypted by using Man in the middle attack during the key exchange. so, in order to provide a secure session, the captured cookie should be of no use to the attacker. The attacker will be needing the random string that is generated by the client in order to gain access to that session which is not possible and is difficult to obtain because it is generated during the establishment of the session and is transmitted through SSH The client verifies its authenticity for every major redirection so that the server can easily identify any malicious connection to that server. The random string may be a hash of a string that is generated randomly or a MAC address of that host or any unique value or it can also be the IP address of that host. But the disadvantage of using the IP address as a random string is that it can be unique only to the network if any other client in a different network can have the same IP address as a previous host. In that case both will have same value for random string, which is not feasible. This solution needs the browser code to be changed for the SSH connection and transmission of messages are needed, so we will be using Python, Java, PHP etc. for generating the random string and establishing a secure session. The use of MAC address as a random string is one feasible way but it does have its own disadvantages when attacker tries to ARP (Address Resolution Protocol) spoof. So, a randomly generated hash of a string gets the job done. The below diagram will give a clear picture of what must happen when a session starts between a client and a server.


Client
Server
1) Client requesting for a session
2) Server replying by generating a session id
3) Client generating a random string as salt and hashing it
4) Client Hashing the session id along with the hashed salt value
5) Client sending Hash(session id + salt) to server
6) Server storing the received hash value

V. RESULTS


phpMyAdmin
Recent Favorites
New
information_schema
mysql
performance_schema
phpmyadmin
phpsamples
tp
New
user_details
Server: 127.0.0.1 » Database: tp » Table: user_details
Browse Structure SQL Search Insert Export Import Privileges Operations Tracking Triggers

✓ Showing rows 0 - 5 (6 total, Query took 0.0083 seconds.) [user_id: 1... - 6...]

```
SELECT * FROM `user_details` ORDER BY `user_id` ASC
```

□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

```
UPDATE `user_details` SET `user_image` = 'kmahesh.jpg' WHERE `user_details`.`user_id` = 6;
```

[Edit inline] [Edit] [Create PHP code]

□ Show all | Number of rows: 25 Filter rows: Search this table Sort by key: PRIMARY (ASC)

+ Options

	user_id	user_email	user_password	user_name	user_type	user_image	user_status	sessionid
□ Edit Copy Delete	1	admin@gmail.com	[illegible]	admin	master	admin.jpg	Active	[illegible]
□ Edit Copy Delete	2	7saisurya7@gmail.com	[illegible]	Surya	user	surya.jpg	Active	[illegible]
□ Edit Copy Delete	3	syamprajwal33@gmail.com	[illegible]	Syam	user	syam.jpg	Active	[illegible]
□ Edit Copy Delete	4	praneethsai1998@gmail.com	[illegible]	Praneeth	user	praneeth.jpg	Active	[illegible]
□ Edit Copy Delete	5	abc@gmail.com	[illegible]	ABC	user	abc.jpg	Active	[illegible]
□ Edit Copy Delete	6	kmahesh@gmail.com	[illegible]	Mahesh	user	kmahesh.jpg	Active	[illegible]

□ Check all With selected: Edit Copy Delete Export

□ Show all | Number of rows: 25 Filter rows: Search this table Sort by key: PRIMARY (ASC)

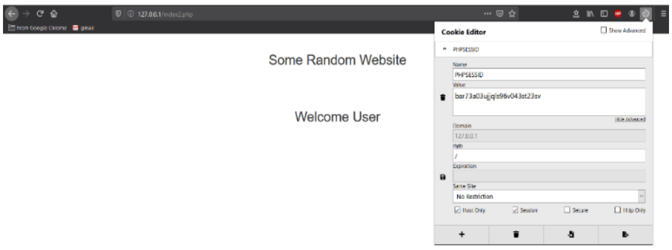
Query results operations

Print Copy to clipboard Export Display chart Create view

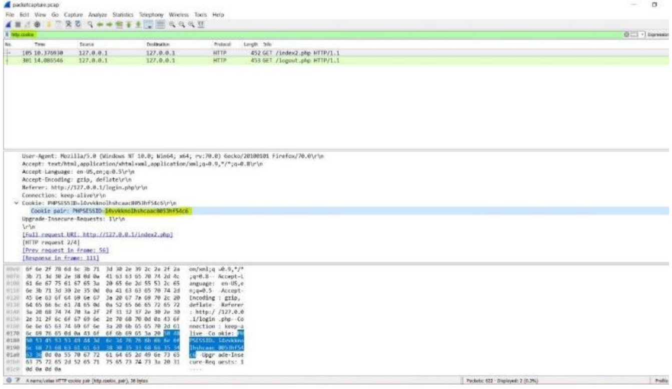
Bookmark this SQL query

Console □ Let every user access this bookmark

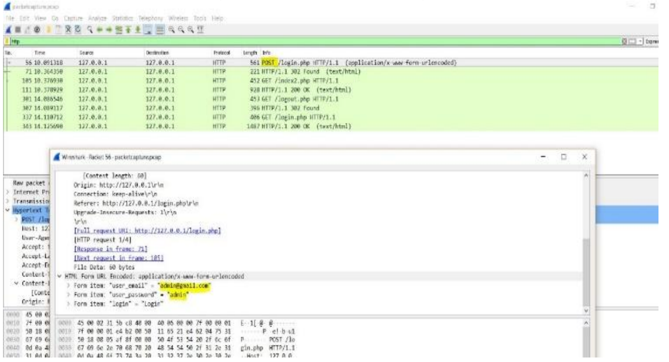
- **Database table containing login details of various users.**


```
127.0.0.1/index2.php

Some Random Website

Welcome User

Cookie Editor                          □ Show Advanced
PHPSESSID
Name
PHPSESSID
Value
bar73a03ujjqls96v043st23sv
Domain
127.0.0.1
Path
/
Expiration

Same Site
No Restriction
☑ Host Only   ☑ Session   □ Secure   □ Http Only
```

- Cookie generated when user logs in


```
packetcapture.pcap
http.cookie
No.  Time         Source     Destination  Protocol  Length Info
105  10.376930    127.0.0.1  127.0.0.1    HTTP      452 GET /index2.php HTTP/1.1
301  14.085546    127.0.0.1  127.0.0.1    HTTP      453 GET /logout.php HTTP/1.1

    User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:70.0) Gecko/20100101 Firefox/70.0\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n
    Accept-Language: en-US,en;q=0.5\r\n
    Accept-Encoding: gzip, deflate\r\n
    Referer: http://127.0.0.1/login.php\r\n
    Connection: keep-alive\r\n
  v Cookie: PHPSESSID=l0vvkknolhshcaac8053hf54c6\r\n
      Cookie pair: PHPSESSID=l0vvkknolhshcaac8053hf54c6
    Upgrade-Insecure-Requests: 1\r\n
    \r\n
    [Full request URI: http://127.0.0.1/index2.php]
    [HTTP request 2/4]
    [Prev request in frame: 56]
    [Response in frame: 111]

6f 6e 2f 78 6d 6c 3b 71  3d 30 2e 39 2c 2a 2f 2a   on/xml;q =0.9,*/*
3b 71 3d 30 2e 38 0d 0a  41 63 63 65 70 74 2d 4c   ;q=0.8·· Accept-L
61 6e 67 75 61 67 65 3a  20 65 6e 2d 55 53 2c 65   anguage:  en-US,e
6e 3b 71 3d 30 2e 35 0d  0a 41 63 63 65 70 74 2d   n;q=0.5· ·Accept-
45 6e 63 6f 64 69 6e 67  3a 20 67 7a 69 70 2c 20   Encoding : gzip,
64 65 66 6c 61 74 65 0d  0a 52 65 66 65 72 65 72   deflate· ·Referer
3a 20 68 74 74 70 3a 2f  2f 31 32 37 2e 30 2e 30   : http:/ /127.0.0
2e 31 2f 6c 6f 67 69 6e  2e 70 68 70 0d 0a 43 6f   .1/login .php··Co
6e 6e 65 63 74 69 6f 6e  3a 20 6b 65 65 70 2d 61   nnection : keep-a
6c 69 76 65 0d 0a 43 6f  6f 6b 69 65 3a 20 50 48   live··Co okie: PH
```

- Capturing the packets using Wireshark.


```
packetcapture.pcap
http
No.  Time         Source     Destination  Protocol  Length Info
56   10.091318    127.0.0.1  127.0.0.1    HTTP      561 POST /login.php HTTP/1.1  (application/x-www-form-urlencoded)
71   10.364350    127.0.0.1  127.0.0.1    HTTP      221 HTTP/1.1 302 Found  (text/html)
105  10.376930    127.0.0.1  127.0.0.1    HTTP      452 GET /index2.php HTTP/1.1
111  10.378929    127.0.0.1  127.0.0.1    HTTP      928 HTTP/1.1 200 OK  (text/html)
301  14.086546    127.0.0.1  127.0.0.1    HTTP      453 GET /logout.php HTTP/1.1
307  14.089117    127.0.0.1  127.0.0.1    HTTP      396 HTTP/1.1 302 Found
337  14.110712    127.0.0.1  127.0.0.1    HTTP      406 GET /login.php HTTP/1.1
343  14.125698    127.0.0.1  127.0.0.1    HTTP      1487 HTTP/1.1 200 OK  (text/html)

Wireshark · Packet 56 · packetcapture.pcap
      [Content length: 60]
    Origin: http://127.0.0.1\r\n
    Connection: keep-alive\r\n
    Referer: http://127.0.0.1/login.php\r\n
    Upgrade-Insecure-Requests: 1\r\n
    \r\n
    [Full request URI: http://127.0.0.1/login.php]
    [HTTP request 1/4]
    [Response in frame: 71]
    [Next request in frame: 105]
    File Data: 60 bytes
v HTML Form URL Encoded: application/x-www-form-urlencoded
  > Form item: "user_email" = "admin@gmail.com"
  > Form item: "user_password" = "admin"
  > Form item: "login" = "Login"

0000  45 00 02 31 5b c8 40 00  40 06 00 00 7f 00 00 01   E··1[·@· @·······
0010  7f 00 00 01 e4 b2 00 50  11 65 21 e4 62 04 75 31   ·······P ·e!·b·u1
0020  50 18 08 05 af 8f 00 00  50 4f 53 54 20 2f 6c 6f   P······· POST /lo
0030  67 69 6e 2e 70 68 70 20  48 54 54 50 2f 31 2e 31   gin.php  HTTP/1.1
```

- Getting the session id from the captured packet.

VI. CONCLUSION

It is very important to secure the session id indeed session cookie so that no one can illegitimately acquire it and use it. Our proposed methodology uses a randomized string as salt value to be hashed along with session id. By doing this, we can make sure that even if the attacker attains the session id, it might not be useful to him as he does not know the user generated random string hence the server will not permit the attacker to access the data present in user's account. The aim of this paper is to send the authenticated cookie over a secure channel so that it can't be eavesdropped. Even if it is eavesdropped, it would be of no use for the attacker.

REFERENCES

1. LaCroix, K., Loo, Y. L., & Choi, Y. B. (2017, July). Cookies and Sessions: A Study of What They Are, How They Work and How They Can Be Stolen. In 2017 International Conference on Software Security and Assurance (ICSSA) (pp. 20-24). IEEE.
2. Li, B., Lv, S. J., Zhang, Y. S., & Tian, M. (2013, May). The application research of Cookies in network security. In PROCEEDINGS OF 2013 International Conference on Sensor Network Security Technology and Privacy Communication System (pp. 152-155). IEEE.
3. Park, J. S., & Sandhu, R. (2000). Secure cookies on the Web. IEEE internet computing, 4(4), 36-44.
4. Liu, A. X., Kovacs, J. M., Huang, C. T., & Gouda, M. G. (2005, October). A secure cookie protocol. In Proceedings. 14th International Conference on Computer Communications and Networks, 2005. ICCCN 2005. (pp. 333-338). IEEE.
5. Lee, W. B., Chen, H. B., Chang, S. S., & Chen, T. H. (2019). Secure and efficient protection for HTTP cookies with self-verification. International Journal of Communication Systems, 32(2), e3857.
6. Pujolle, G., Serhrouchni, A., &Ayadi, I. (2009, December). Secure session management with cookies. In 2009 7th International Conference on Information, Communications and Signal Processing (ICICS) (pp. 1-6). IEEE.
7. Yao, Y., Palaima, M., & Goldberg, A. (2007). U.S. Patent Application No. 11/172,625.
8. Gauravaram, P. (2012, November). Security Analysis of salt|| password Hashes. In 2012 International Conference on Advanced Computer Science Applications and Technologies (ACSAT) (pp. 25-30). IEEE.
9. Badra, M., Guillet, T., &Serhrouchni, A. (2009, September). Random values, nonce and challenges: Semantic meaning versus opaque and strings of data. In 2009 IEEE 70th Vehicular Technology Conference Fall (pp. 1-5). IEEE.
10. Kim, Y. I. (2008). U.S. Patent No. 7,386,876. Washington, DC: U.S. Patent and Trademark Office.
11. Qi, Z. (2007). U.S. Patent No. 7,299,355. Washington, DC: U.S. Patent and Trademark Office.
12. Swain, G., &Lenka, S. K. (2012). A technique for secret communication using a new block cipher with dynamic steganography. International Journal of Security and Its Applications, 6(2), 1-12.
13. Sandeep, P. M. V. K., Mukesh, V. B. S., & Swain, G. (2017). A SECURE INTRUSION DETECTION SYSTEM FOR MULTIPATH TCP. International Journal of Pure and Applied Mathematics, 116(5), 165-169.
14. Swain, G., & Lanka, S. K. (2012). A quick review of network security and steganography. International Journal of Electronics and Computer Science Engineering, 1(2), 426-435.
15. Kusuma, N., Tejaswi, N. S., Anitha, T., & Kiran, K. V. D. Network Security Using Quantum Cryptography.

AUTHORS PROFILE

PrasunaKotturuis working as an Assistant Professor in KoneruLakshmaiah Education Foundation in the Department of Computer Science and Engineering. Had 9+ years of experience in academics and industry. Her research interests include Machine learning, Wireless sensor networks, Network security and Data Science.

SyamPrajwalKammula is aB. Tech final year studentof Computer Science and Engineering inKoneruLakshmaiah Educational Foundation, Guntur. His areas of interest are Network security, Wireless sensor networks and Cyber security.

Sai Surya Bungais a B. Tech final year student of Computer Science and Engineering inKoneruLakshmaiah Educational Foundation, Guntur. His areas of interest are Network security, Wireless sensornetworks and Cyber security.

Praneeth Sai Atluriis a B. Tech final year student of Computer Science and Engineering inKoneruLakshmaiah Educational Foundation, Guntur. His areas of interest are Network security, Wireless sensor networks and Cyber security.