

Secure Model for Session Hijacking using Hashing Algorithm

Nidhi Thakkar¹, Rajvirsinh Vaghela²

¹Research scholar, IT Systems and Network Security PG School, Ahmedabad, Gujarat, India

²Assistant Professor, Computer Science & Engineering, NSIT, Jetalpur, Ahmedabad

ABSTRACT

Session Hijacking is one of the most used attacks by the attacker. Session Hijacking is the second most attack as per the OWASP latest release in the year of 2017. It is the most crucial attack through which attacker can gain the access of the client's running session. In Session Hijacking attacker steals the session id of the victim and with the use of that session id attacker can able to access the current session. In this time, the session is being secured with the use of SSL or TLS. There are few drawbacks and less security of the SSL. Here a new approach is proposed to face the issue of Session Hijacking attack. The system is specially designed to address the issue of session hijacking in local network. The proposed system is using the MAC address and client's id and generates the session id with MD5 algorithm. MD5 algorithm is providing the 128 bits session id. At the time of request Server will authenticate the user bases on Session Id as well as MAC address of the system. In case of successful session key theft, attacker will not be able to access the session as the MAC is also required to access the session. To make it more secure and reduces the chances of MAC spoofing, the use of ARP protocol can be restricted in the network. The proposed system is best solution for attack of Session Hijacking in the local network. After successful implementation the system will provide the security against this type of attack.

Keyword: - Session Hijacking, Session ID, Session, Local Area Network., MAC address

1. INTRODUCTION

1.1 What is Session?

A Session Is created when a user access or log-in to a particular web page or program, computer and ends when the user shut down or log out the computer, close the web page or program. A session can temporarily store information associated with the activities of users whereas connected. [9].

1.2 Why Session is required?

The main requirement of the session is to maintain the record of the user's login details and with the use of active session user can access the application [9].

1.3 What is Session Hijacking?

When an attacker gets access of particular user's session state it is known as session hijacking. The attacker stealth the legitimate session ID which is utilized to get into the system and gather the information. [10].

1.4 Types of Session Hijacking

1.4.1 Active Session Hijacking

In Active Session Hijacking they targets to an already authenticated session [9]. In Active session hijacking the attacker steal the cookies to hijack the active session when the client has signed-in his account and after that disconnect the initial client from the server [9].

How Active Session Hijacking Works

In these hijacking techniques attackers utilize the client-side scripts to theft the cookies of original client by using social engineering techniques which include emails, private messaging on forums and on different social networking sites [9]. It is called active session hijacking because attackers required to interact and require few actions to be performed by the victim to theft the session effectively which may raise the suspicion level [9].

1.4.2 Passive Session Hijacking:

Passive hijacking means that the attacker does not need to hijack the active session but attackers capture the login credentials whereas the original client is trying to attempt to create a new connection with the server, and attacker is sitting mutely on an equivalent network and save the clients login credentials [9].

How Passive Session Hijacking Works

In the Passive Session hijacking the attacker use the network sniffing tools that captures data packet and exploit the vulnerability of ARP protocol by poisoning the network and attacker analysed that data to find clients login credentials information [9]. It is called passive session hijacking because the attacker does not require interacting with the client and building him to any specific actions. There's less risk of suspicion level then active session hijacking [9].

1.5 Hashing

Transformation of a string into a usually shorter fixed length value or key that represents the original string is known as Hashing. Hashing is used to check the integrity of the message.

2. RELATED WORK

Author in paper [1] recommend that the developers need follow correct security practices as well as uses the encryption for encrypting user's credentials within the database or employing a strong hashing function with unique salt for the password. The author in this paper [2] utilize the threat analysis methodology proposed by Stango et A1. [3]. The approach surveys security of a system by deciding threats and vulnerabilities. It includes seven steps: Description, Identifying Assets, Architectural Overview, Threat Identification, Threat prioritization, Determining Vulnerabilities and Possible ways to prevent it. Paper [4], To secure communication in a web session, cryptographic techniques like one-way (unidirectional) hash chain(OHC) technique that depend on one-time passwords proposed by Lamport have been utilizes. In this paper they proposed a solution that benefits from the advantages of the sparse caching strategy, in addition to efficiency of the two-dimensional one-way hash construction.

3. Proposed System

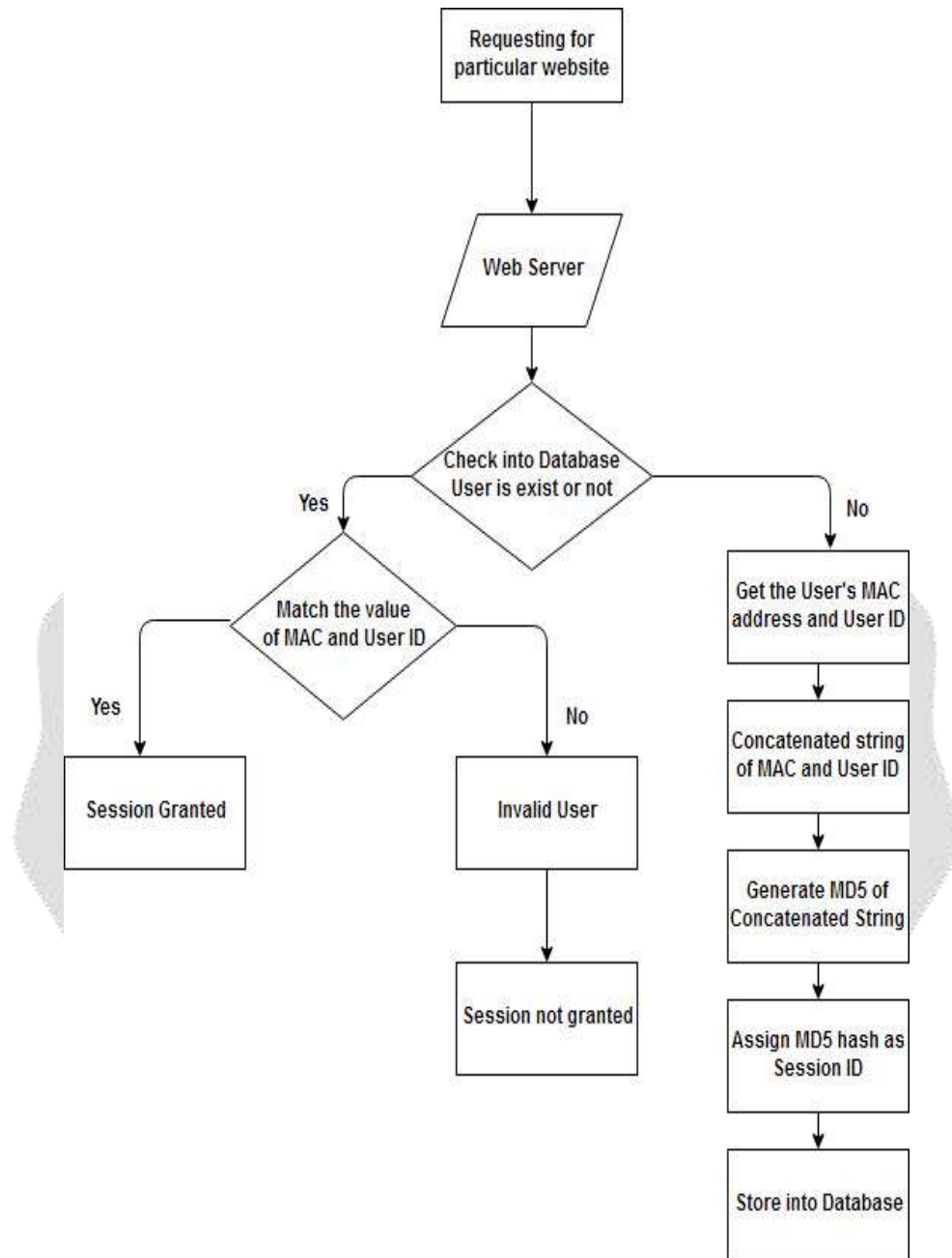


Fig 1: Proposed System

Above diagram describe the proposed system for Secure model for session hijacking using hashing algorithm.

Step 1: Start

Step 2: User or client request for particular website

Step 3: User try to login

Step 4: Web server check into Database User exist or not

Step 5: if user exist into Database

```

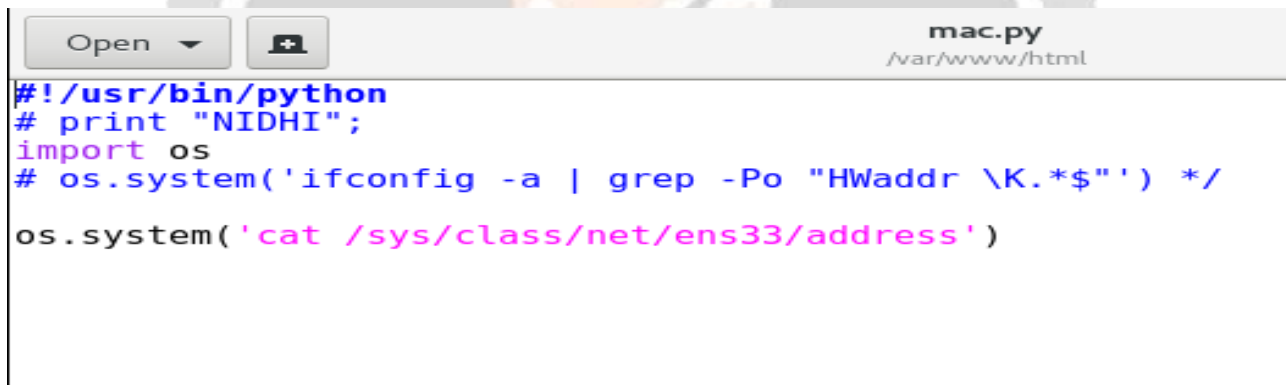
Match the Value of MAC and User ID
if (Match)
    Session Granted
else
    Invalid User
else
    Get User's MAC
    Get User ID from User input
    Concatenated MAC + User ID
    Generate MD5 hash of Concatenated String
    Assigning MD5 hash as a session Id to client
    Store MD5 hash into Database

```

Step 6: Stop

4. Experimental Work

In the First Step, Find MAC address using Python script.



```

mac.py
/var/www/html

#!/usr/bin/python
# print "NIDHI";
import os
# os.system('ifconfig -a | grep -Po "HWaddr \K.*$"') */
os.system('cat /sys/class/net/ens33/address')

```

Fig 2: Finding MAC

In Second Step, Script for Generating Hash Value.



```

md5.py
/var/www/html

#!/usr/bin/python

import sys
import hashlib

uname = sys.argv[1];
mac = sys.argv[2];
md = uname+mac

print(hashlib.md5(str(md).encode('utf-8')).hexdigest())|

```

Fig 3: Generate Hash

Create Session ID using User's MAC Address and User ID

```

function my_session_start()
{
    if (!empty($_SESSION['deleted_time']) && $_SESSION['deleted_time'] < time() - 180)
    {
        session_destroy();
        session_start();
    }
}

function my_session_regenerate_id()
{
    if (session_status() != PHP_SESSION_ACTIVE)
    {
        session_start();
    }
    $_SESSION['deleted_time'] = time();
    session_commit();
    ini_set('session.use_strict_mode', 0);
    echo "<br><br>";
    echo "<br>";
    $uname=$_SESSION['username'];
    echo "USERNAME IS :- ",$uname."<br>";
    $mac = (string) shell_exec('python mac.py');
    $smd = shell_exec('python md5.py $uname $mac');
    echo "MAC ADDRESS OF THE SYSTEM IS :- ",$mac."<br>";
    echo "HASH of UNAME + MAC :- ",$smd."<br>";
    session_id($smd);
    session_start();
    echo "Session ID IS :- ";
    echo session_id();
}

ini_set('session.use_strict_mode', 0);
my_session_start();
my_session_regenerate_id();
    
```

Fig 4: Generate Session ID

Successfully Generated Session ID

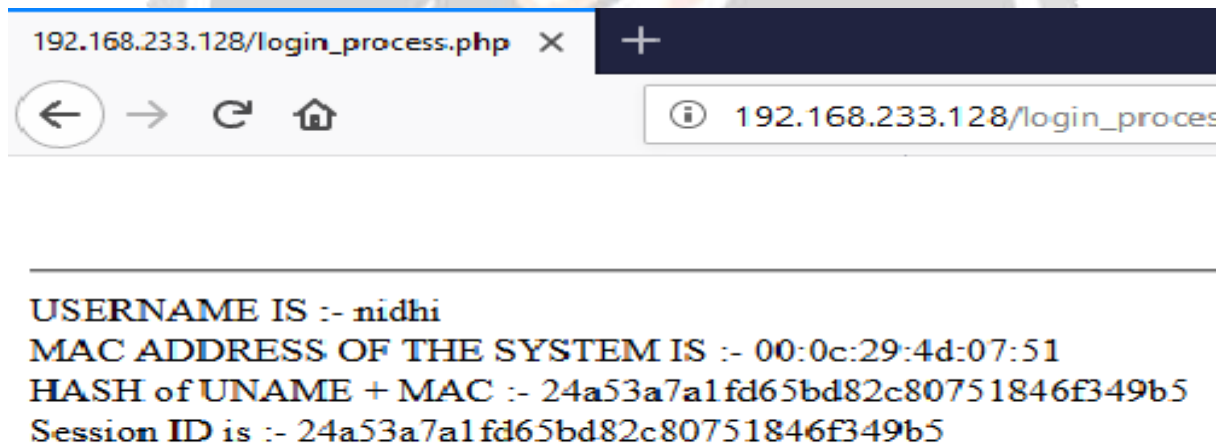


Fig 5: Successfully Generated Session ID

5. RESULT AND ANALYSIS

Session hijacking attack is performed on web applications here. Session hijacking attack is successfully mitigated by the proposed system solution. All the testing scenario are performed to check the efficiency of the proposed work. There are 50 different web applications are tested with proposed solution. Session hijacking is successfully mitigate from all the 50-web application. The results indicate that the overall efficiency of the proposed work is 100%.

6. CONCLUSIONS

With the emergence of the Web technologies there is a need to secure the web application from session hijacking attack. Generally, in the web application the session id is randomly generated, so the attacker easily steal the session ID using some browser access. By using this session id they can get the access of legitimate user's session. The main purpose this work is to mitigate the session hijacking attack using hash function to generate hash string of combined string of MAC address of user machine and user ID. The proposed system precisely mitigates session hijacking attack from the web application.

5. REFERENCES

- [1]. Al-Khurafi, O. B., & Al-Ahmad, M. A. (2015, December). Survey of Web Application Vulnerability Attacks. In *Advanced Computer Science Applications and Technologies (ACSAT), 2015 4th International Conference on* (pp. 154-158). IEEE.
- [2]. Jain, V., Sahu, D. R., & Tomar, D. S. (2015). Session Hijacking: Threat Analysis and Countermeasures. In *Int. Conf. on Futuristic Trends in Computational Analysis and Knowledge Management*.
- [3]. Stango, A., Prasad, N.R. and Kyriazanos, D.M., 2009, June. A threat analysis methodology for security evaluation and enhancement planning. In *Emerging Security Information, Systems and Technologies, 2009. SECURWARE'09. Third International Conference on* (pp. 262-267). IEEE.
- [4]. Alabrah, A. and Bassiouni, M., 2014, October. Preventing session 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.
- [5]. Cheng, K., Gao, M. and Guo, R., 2010, April. Analysis and research on HTTPS hijacking attacks. In *Networks Security Wireless Communications and Trusted Computing (NSWCTC), 2010 Second International Conference on* (Vol. 2, pp. 223-226). IEEE.
- [6]. Lin, M., 2005. An overview of session hijacking at the network and application levels. SANS Institute InfoSec Reading Room.
- [7]. Noiumkar, P. and Chomsiri, T., 2008, November. Top 10 free web-mail security test using session Hijacking. In *Convergence and Hybrid Information Technology, 2008. ICCIT'08. Third International Conference on* (Vol. 2, pp. 486-490). IEEE.
- [8]. Kapoor, S., 2006. Session hijacking exploiting TCP, UDP and HTTP sessions. infosecwriters.com/text_resources/.../SKapoor_Session Hijacking.pdf
- [9]. <https://supportforums.cisco.com/t5/security-blogs/session-hijacking-and-web-based-attacks/ba-p/3106660>.
- [10]. <http://www.hackingarticles.in/session-hijacking/>.
- [11]. <https://qcboss.wordpress.com/2011/08/22/two-levels-of-session-hijacking/>.
- [12]. <https://qcboss.files.wordpress.com/2011/08/sessionhijackinglevels4.jpg>.
- [13] <https://gpgtools.tenderapp.com/kb/how-to/introduction-to-cryptography>.
- [14] <http://kryptophone.kryptotel.net/faq/encryption/index.html>.
- [15]. <https://stackoverflow.com/questions/4948322/fundamental-difference-between-hashing-and-encryption-algorithms>.
- [16]. <https://www.sans.org/reading-room/whitepapers/windows/session-hijacking-windows-networks-2124>.