iJRASET
International Journal For Research in
Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH
IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 9 Issue: IV Month of publication: April 2021

DOI: https://doi.org/10.22214/ijraset.2021.33679

Efficient System to Prevent Session Hijacking Attack on Servers using Proxy Server

Anuradha Maurya[1], Veena Kulkarni[2]
[1, 2]Computer Engineering, Thakur College of Engineering and Technology Kandivali, Mumbai, India

Abstract: HTTP is a stateless protocol thus we tend to use cookie to maintain session. For maintaining a session server creates a unique identifier to remember on going session for particular client. This distinctive identifier is nothing but a randomly generated text which is stored on client browser in the form of cookie. These cookies are generally created in the login process, after successful authentication process, server generates unique id for requested client and send it to client. Browser sends this cookie on every request where authentication of user is required. Authentication of cookie become temporary replacement of user password authentication for the entire session. Cookies are static in nature, they do not change in the session life time, because of this nature anyone can steal and use this cookie for their benefit. Use of cookies introduces a number of risks in security especially in session authentication. For secure communication use of HTTPS is not easy for those applications that are highly distributed due to performance and financial issues and HTTPS provide security at network only there are many ways were attacker can steal cookie by using different attack like cross site scripting attack, cross site tracing attack, domain related attack etc. Hence, creating a system using Concept of one-time dynamic cookie (OTDC) for authentication instead of HTTPS. OTDC will prevent various attacks on servers. A reverse proxy server with OTDC, IP Address, session ID and browser fingerprinting are used to prevent opponent from capturing session credentials. Setting up HTTPS to HTTP Reverse Proxy Server, Session time out implementation on Proxy server because "The less time you give your account to be cracked, the better for you." Generating log on attack detection and reporting to administrator.

Keywords: prevention of session hijacking, cookies, reverse proxy server, one-time dynamic cookie (OTDC), dot net core

I. INTRODUCTION

In today's world, web application security is a major concern, where session vulnerabilities are common in web applications, attacker can take benefits of poorly configured websites or web-portals, he/she can take control over one's session identity.

In session management session ids or cookies plays a vital role for maintaining a session. The prevention mechanism on session hijacking is required for web applications security.

There are different types of attacks can be performed by attacker on website. Attacker can gain unauthorised control to perform fraud, illegal activities on others information. Session management is a dignified part of web applications; as it provides benefit to track user specific state as authenticated user account, across enormous requests. Unfortunately, various applications provide session management on ensure HTTP, which led to vulnerable session. On vulnerable session it is easy for attacker to hijack or perform session fixation attack.

Other side deployment of web application on secure HTTPS, causes some disadvantages such as latencies, browser caching, mix mode issue. Session hijacking attack is essentially of 2 types: -Active session hijacking and Passive session hijacking. In active session hijacking authenticated user's session is being hijacked, in this method of hijacking user would be already remains logged into the active session of one's profile or account. Here hackers try to steal cookies from network and led to hijack the active session the primary user cannot a lot of login into his/her profile and he is disconnected from the server. during this methodology of hijacking, the hacker doesn't attack any active session.

They follow some completely different method to induce the whole data of the login credentials of the user. once the user enters, his/her credentials in to the system and tries to access his/her account on the network, hacker then steals user credentials and hacks account and access all account data. In this paper, we've got analysed a hindrance technique for session hijacking. during this technique, we tend to bind the network layer and application layer along through reverse proxy server. This reverse proxy server can generate session credentials like session ID, browser ID and one-time dynamic cookie (ODTC). This mechanism detects the modification in browser thanks to that an adversary cannot get the illegal access. Since users are bind with machine and browser and with new cookie for every request within the session.

II. LITERATURE SURVEY

There are big number of scientists are coming up with and developing different detection tools for various attacks and different bar techniques for remedy of session hijacking. Some of the Detection Tools and Techniques for Session hijacking [2] are Arp-ON, ARP-PING, ANTI-SNIFF, Cookie Monster, Wavelet Based Detection, Cisco Intrusion Detection System (IDS), and Sans Intrusion Prevention System (IPS) are shown in the Table I.

TABLE I
Session Hijacking Detection Tools

Tool name	Author & year	Tool function	Type of attack to avoid
Arp-ON	Darknet, 2000	To secure the Address resolution Protocol	MITM (Man-In- The-Middle) Attacks
ARP- PING	Beyond- Security, 1998	Allows the user to ping MAC address Directly	To detect Sniffer on the network
Anti- sniff	Storm, 2011	Used for packet capturing	Packet sniffer
Cookie- Monster	Pauli, Engebrets on, Ham &Zautke, 2011	Analyzing the strength of the cookie by Archiving & analyzing	Cookie stealing
Wavelet- based-detection	Long &sikdar, 2008	Analysis the signal strength using wavelet Transform	Session hijacking

There are some algorithms and Techniques for preventing Session Hijacking [2], they are

1) Ensuring a secure Cookie generation
2) Implementing the CIA technology, C-Confidentiality, I-Integrity, A-Authentication.
3) Use encrypted connections or protocols for the transmission of data such as HTTPS, Open SSH protocol suite.
4) Locking a particular session to its corresponding user’s i.e. Session Lock [5].

A. *Ensuring a Secure Cookie Generation*

Cookie is the one important component in web-based applications, which can be used to authenticate a client and cookie information is sent to the server system for that user. The cookie information contains user name, password, timestamp, session timeout. Whenever a session is hijacked, the attacker analyses this cookie information and comprises the user’s account[5].

B. *Implementing the CIA Technology*

This is the low-cost solution to secure session cookies. This also prevents cross-site-scripting. This is achieved by creating a java script on the server side and cookie is available only to the client’s browser. It develops the standard plug-in for the Chrome, Mozilla Firefox, etc[5].

C. *Encrypted Connections for The Transmission*

The connection was encrypted with HTTPS (Secure Hyper Text Transfer Protocol) and this protocol uses an SSL (Secure Socket Layer) protocol stack. The drawback is cost. Implementing and maintaining this service is cost effective[5].

D. *Locking a Particular Session*

This is an effective technique to prevent session hijacking. Even if the session is hijacked, the attacker has no use. A unique HMAC (Hashed Message Authentication Code) algorithm with secret key is shared between client and server. A token is generated from an initial login over the SSL. This is an efficient and low-cost solution to secure the session[5].

There are different research papers found in literature survey, different session hijacking prevention techniques mentioned in Table II.

Table II
Existing Session Hijacking Prevention Techniques

Author	Prevention Technique	Summary	Gap Identified
Nikiforakis	Session shield	Protect from Xss	Limited to browser protection
Dacosta	OTC (one-time cookie)	Light weight, disposable good performance	Always need to be configured with proxy server and long session live with no session time out and SQL Injection Attack can happen
Asif & Tripathi	Double authentication to user id	Double factor authentication system	Venerable to many threats
Amerah & Mostafa	One-way hash chain (OHC)	effective protection against hijacking	Need high computation storage capacities, Not working on mobile devices
Unknown	Dynamic ID generation Using Session Id and Client Time using MD5	Analysis the signal strength using wavelet Transform	Session hijacking
Annies Minu Sathiyaseelan, Vincy Joseph, Anuradha Srinivasaraghavan	A proposed system for preventing session hijacking with modified one-time cookies	Protect from MITM	Just Proposed System, Not implemented yet
Karis D'silva, J Vanajakshi; K N Manjunath; Srikanth Prabhu	Hashing of session Id, browser name, Browser Platform, Browser major and minor version	Dynamic hash ID generation on each time when session is created	On registration time it takes unique finger printing and use the same hash value so MITM attack can happen and hash digest are vulnerable

III. PROBLEM DEFINITION

To Implement a system that will protect web application on server level were no need to implement additional functionality on client side and server side. Proxy server will provide security from session hijacking attack. Design a system that will detect & prevent session hijacking using the concept of OTDC (One-Time-Dynamic-Cookie) with Reverse Proxy Server. Setting up HTTPS protocol environment to the HTTP Reverse Proxy server to improve application performance and Implementing Session time out on Proxy server. On detection of malicious activity system will generates log and at the same time it will Report to the Administrative about the malicious activities. For client authentication, system will use Browser Fingerprinting concept which will include client Ip address, client system time and browser information.

Contribution of this thesis will be to make web application more secure and protected from attackers by using One-time Dynamic Cookie and browser fingerprinting Concept, improve robustness of web applications. Implemented system will protect from various types of session hijacking attack and will provide real time updates about ongoing activities on server to administration of the server.

IV. METHODOLOGY

This section can describe the strategies to investigate the work analysis queries and also the reasons for selections created. Here also will take a glance at different challenges and also the limitations relating to these strategies. the most goal of the analysis to review a way to stop session hijacking attack for web application using reverse proxy server. In this research, will learn about how the attacker perform session hijacking attack and finding out the prevention solution for it, to provide a safe and secure way to guard server by them self at server level using proxy server. This kind of network layout, with one proxy/reverse proxy acting as middle ware between end-machines, is common in organization work-groups and which makes this method appropriate for insider session hijacking prevention. The below mentioned flow chart diagram is shown in Fig.1 showing the entire flow of the implemented system which will prevent from different session hijacking attack, this system will prevent from man in middle attack, session fixation attack and auto session time out implementation on proxy server, log generation on malicious activity and real time reporting to the server administration about these ongoing malicious activities.

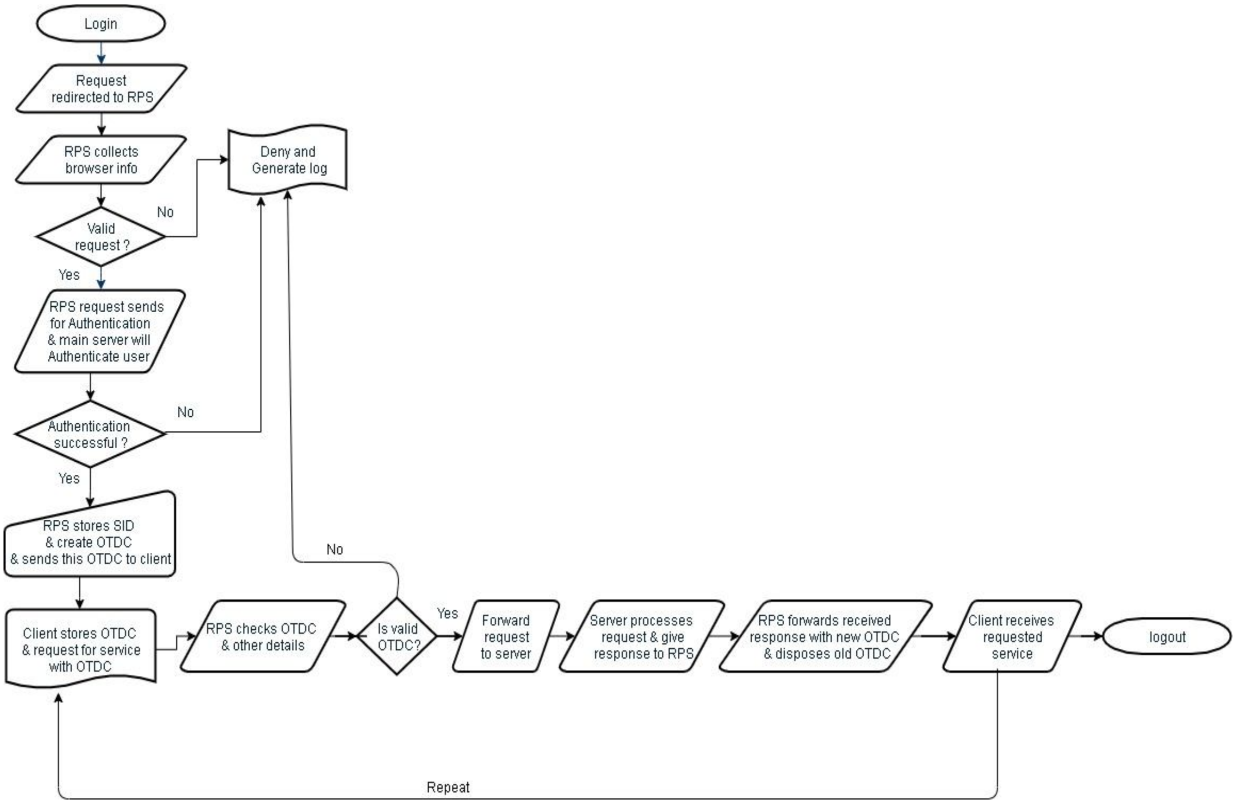

Login
Request redirected to RPS
RPS collects browser info
Valid request ?
No
Yes
Deny and Generate log
RPS request sends for Authentication & main server will Authenticate user
Authentication successful ?
No
Yes
RPS stores SID & create OTDC & sends this OTDC to client
Client stores OTDC & request for service with OTDC
RPS checks OTDC & other details
Is valid OTDC?
No
Yes
Forward request to server
Server processes request & give response to RPS
RPS forwards received response with new OTDC & disposes old OTDC
Client receives requested service
logout
Repeat

Fig. 1 Flow chart of implemented system

V. SYSTEM ARCHITECTURE

The session hijacking preventer system will consist of three major components, these components are mentioned below:

1) *Client:* Client or user of the Web Application who initiates a request for resource to server. Consider user want to access some page, he/she will send a request using the login credential to the server. Server will authenticate the user, after successful authentication, user will be given an OTDC by which user will be authenticated for every new request. Every new request will be having an OTDC which will be sent along with the request.
2) *Reverse Proxy Server:* Proxy Server is nothing but a computer that acts as intermediary between client and server. Here we are using RPS at server side. Thus, every request from client has to be pass via RPS. The RPS will get IP Address, browser finger print, Set OTDC, session ID and Time for every request, if one of these parameters would change then RPS will redirect to another web page.
3) *Server:* This the actual main server where request will be processed, server authenticate user and process all the request and give response to all client request.

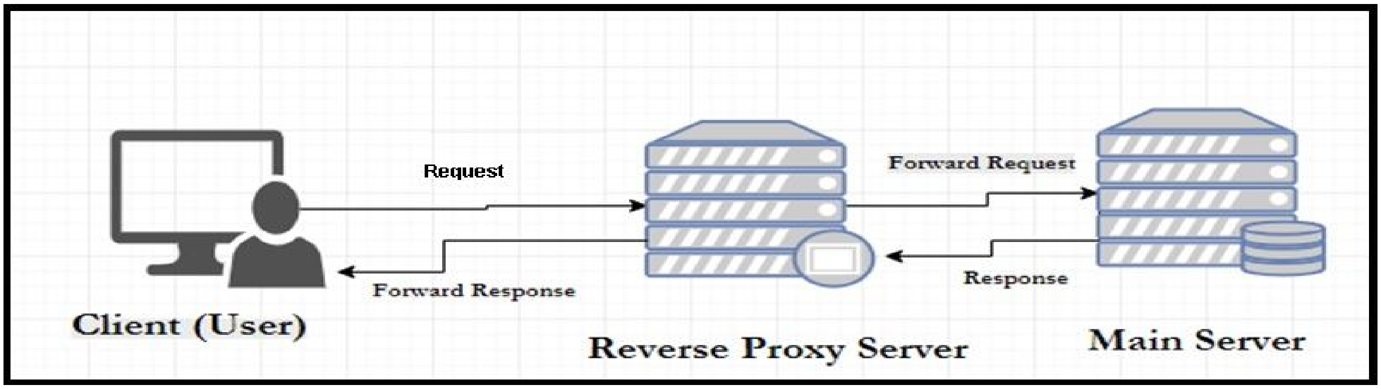


Fig. 2 System architecture

In below Fig 3 shows the block diagram of existing system in which OTC (One-Time-Cookie) concept has been used for prevention of session hijacking attack[1], in our implemented system modified this concept to provide more security, setting up HTTPS to HTTP Reverse Proxy server and implementing session time out on Proxy server, to authenticate client, system will use browser fingerprinting concept that include client Ip address , client system time and browser information and generating log to inform administration about ongoing or attempted attacks.

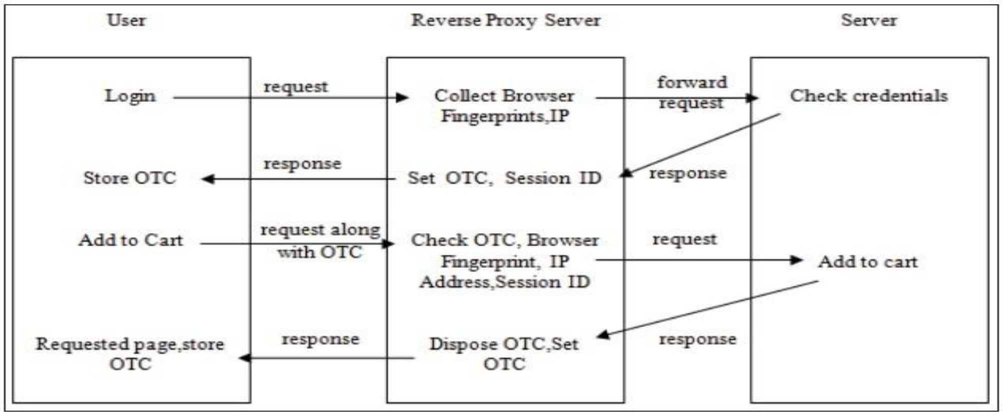


Fig. 3 Block diagram of existing system

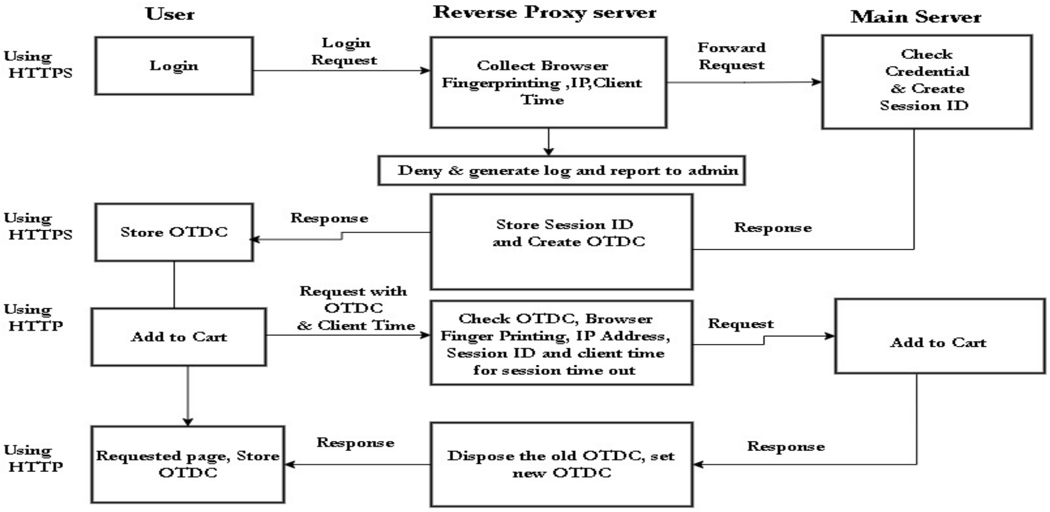

User
Reverse Proxy server
Main Server
Using HTTPS
Login
Login Request
Collect Browser Fingerprinting ,IP,Client Time
Forward Request
Check Credential & Create Session ID
Deny & generate log and report to admin
Using HTTPS
Store OTDC
Response
Store Session ID and Create OTDC
Response
Using HTTP
Add to Cart
Request with OTDC & Client Time
Check OTDC, Browser Finger Printing, IP Address, Session ID and client time for session time out
Request
Add to Cart
Using HTTP
Requested page, Store OTDC
Response
Dispose the old OTDC, set new OTDC
Response

Fig. 4 Block diagram of implemented system

The implemented system will work as follows:

a) User request for authentication by doing login.

b) Request will go on Reverse proxy server (RPS)

c) RPS will collects browser information, time, IP Address

d) RPS will forward request to main server for authentication of the user. Otherwise drop the activity and generate log and inform to administration.

e) Server will check client's credential, authenticate the user after that generate session id and sent it to the RPS.

f) RPS will save session id and generate new OTDC and sends this OTDC to client

g) Client will receive OTDC and start requesting resources or pages form server using current OTDC.

h) RPS receives request with OTDC and check with saved OTDC if it get matched then request is from actual user then RPS will forward this request to main server otherwise it will deny the requested page and generate error log.

i) Forwarded request is received by Main Server then server will process request and send it to the RPS.

j) RPS will receive response and will generate new OTDC and sent response to the user along with new OTDC and disposes old OTDC.

k) This will be continued with every request in web application for entire session.

VI. RESULT AND DISCUSSION

1) System would collect following information on every request is made to access server:

a) client's machine and browser information

b) client's request time

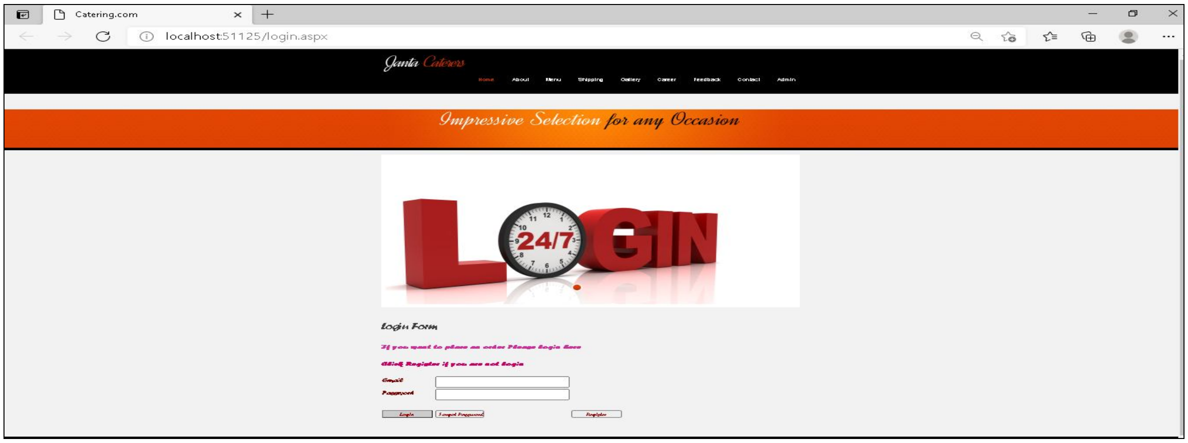
c) client's IP information and above all data will be maintained in OTDC session table

2) A IsValidRequest method will be responsible for checking incoming request which includes device, browser information, existing OTDC Token validation (if any) and session Id.

3) If method returns request is invalid then log will be generated into table to notify admin.

4) Log Report will content all necessary data from where and when this invalid request has been made with failure of status code

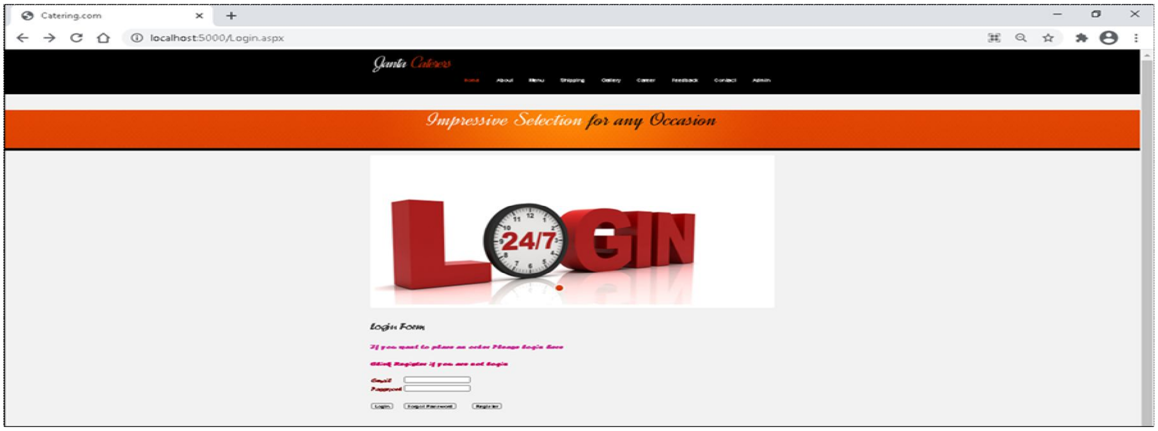
Below Fig. 5 is sample website hosted on http://localhost:51125 which is acting as actual server here, this website will be integrated with reverse proxy server (RPS), now this sample website will be accessible by using http://localhost:5000. In reverse proxy server (RPS) requests will get directed to actual server on http://localhost:51125 and taking response from actual server, the RPS will forward response to client.



If you want to place an order Please login here

Click Register if you are not login

Fig. 5 Sample website hosted on actual server (port 51125)

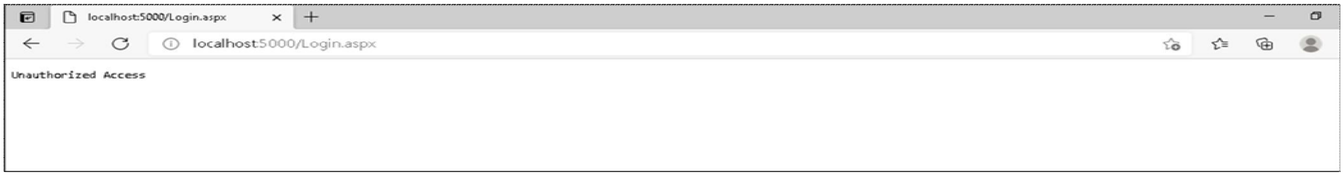


If you want to place an order Please login here

Click Register if you are not login

Fig. 6 Sample website hosted on reverse proxy server (port 5000)

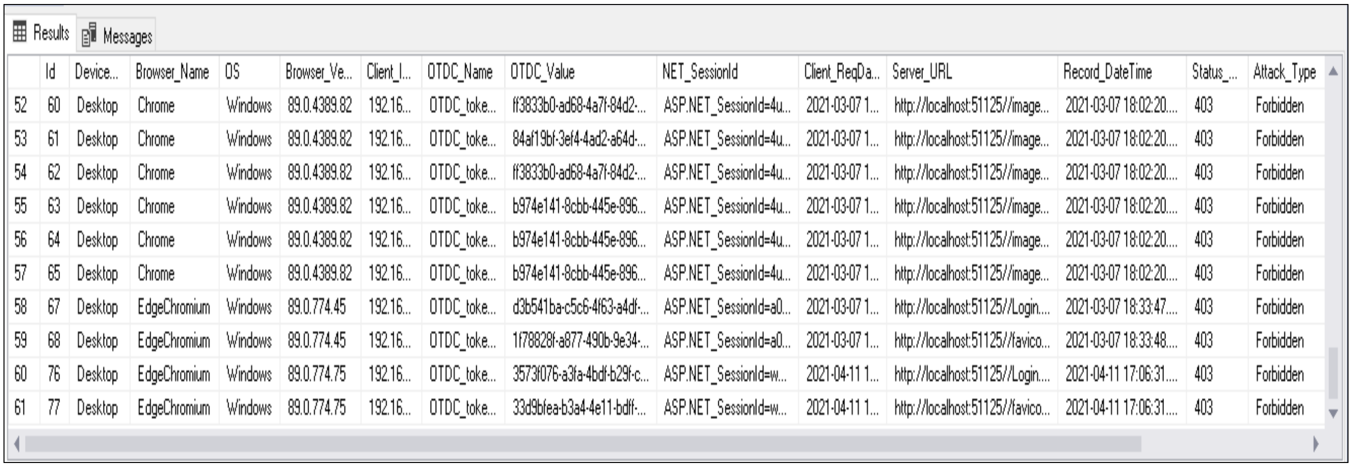
In Fig. 7 if a hacker tries to perform session hijacking by stealing user’s session cookie, implemented system will respond with forbidden status, gives message as “Unauthorized Access” and prevent further information access. Log will be generated with status code and system information of hacker.



```
Unauthorized Access
```

Fig. 7 Prevention from attack using implemented system

In Fig. 8 Log has been generated when hacker tries to perform malicious activity by using stolen session cookie, device and browser information like device type, browser's name, browser's version etc. with Ip address, used session cookie id, requested URL, date time and type of attack with status code.



	Id	Device...	Browser_Name	OS	Browser_Ve...	Client_I...	OTDC_Name	OTDC_Value	NET_SessionId	Client_ReqDa...	Server_URL	Record_DateTime	Status_...	Attack_Type
52	60	Desktop	Chrome	Windows	89.0.4389.82	192.16...	OTDC_toke...	ff3833b0-ad68-4a7f-84d2-...	ASP.NET_SessionId=4u...	2021-03-07 1...	http://localhost:51125//image...	2021-03-07 18:02:20....	403	Forbidden
53	61	Desktop	Chrome	Windows	89.0.4389.82	192.16...	OTDC_toke...	84af19bf-3ef4-4ad2-a64d-...	ASP.NET_SessionId=4u...	2021-03-07 1...	http://localhost:51125//image...	2021-03-07 18:02:20....	403	Forbidden
54	62	Desktop	Chrome	Windows	89.0.4389.82	192.16...	OTDC_toke...	ff3833b0-ad68-4a7f-84d2-...	ASP.NET_SessionId=4u...	2021-03-07 1...	http://localhost:51125//image...	2021-03-07 18:02:20....	403	Forbidden
55	63	Desktop	Chrome	Windows	89.0.4389.82	192.16...	OTDC_toke...	b974e141-8cbb-445e-896...	ASP.NET_SessionId=4u...	2021-03-07 1...	http://localhost:51125//image...	2021-03-07 18:02:20....	403	Forbidden
56	64	Desktop	Chrome	Windows	89.0.4389.82	192.16...	OTDC_toke...	b974e141-8cbb-445e-896...	ASP.NET_SessionId=4u...	2021-03-07 1...	http://localhost:51125//image...	2021-03-07 18:02:20....	403	Forbidden
57	65	Desktop	Chrome	Windows	89.0.4389.82	192.16...	OTDC_toke...	b974e141-8cbb-445e-896...	ASP.NET_SessionId=4u...	2021-03-07 1...	http://localhost:51125//image...	2021-03-07 18:02:20....	403	Forbidden
58	67	Desktop	EdgeChromium	Windows	89.0.774.45	192.16...	OTDC_toke...	d3b541ba-c5c6-4f63-a4df-...	ASP.NET_SessionId=a0...	2021-03-07 1...	http://localhost:51125//Login....	2021-03-07 18:33:47....	403	Forbidden
59	68	Desktop	EdgeChromium	Windows	89.0.774.45	192.16...	OTDC_toke...	1f78828f-a877-490b-9e34-...	ASP.NET_SessionId=a0...	2021-03-07 1...	http://localhost:51125//favico...	2021-03-07 18:33:48....	403	Forbidden
60	76	Desktop	EdgeChromium	Windows	89.0.774.75	192.16...	OTDC_toke...	3573f076-a3fa-4bdf-b29f-c...	ASP.NET_SessionId=w...	2021-04-11 1...	http://localhost:51125//Login....	2021-04-11 17:06:31....	403	Forbidden
61	77	Desktop	EdgeChromium	Windows	89.0.774.75	192.16...	OTDC_toke...	33d9bfea-b3a4-4e11-bdff-...	ASP.NET_SessionId=w...	2021-04-11 1...	http://localhost:51125//favico...	2021-04-11 17:06:31....	403	Forbidden

Fig. 8 Generated log for attacks

Following are features of implemented system

A. Prevent server from session hijacking attack using concept One-Time Dynamic Cookie (OTDC)
B. A system which will be independent to client and server or vice versa
C. A system using Reverse proxy server which will act as intermediary between user and server to protect from malicious attacks.
D. Generate logs on attack and inform administration about attack is going on or happened on server.
E. To authenticate client, system will use Browser Fingerprinting concept that include client Ip address, client system time and browser information.

VII. CONCLUSIONS

Implemented a system that will detect and prevent session hijacking using concept OTDC(One-Time-Dynamic-Cookie) with reverse proxy server. For authentication of client, system will use browser fingerprinting which includes client Ip address, client system and browser information. This system will setup Https to http reverse proxy server, Implementing session time out on proxy server. Generates logs on malicious activity.

VIII. FUTURE SCOPE

Scope to provide more security to main server from SQL injection attack, by doing sanitization before server execution.
With advancement in technology, scope to work on data loses and latency by improving system framework.

IX. ACKNOWLEDGMENT

The we hereby take the privileged to present our paper on Efficient System to Prevent Session Hijacking Attack on Servers Using Proxy Server. We are very thankful to Mrs. Veena Kulkarni whose guidance and support was an immense motivation for us to carry on with our paper.
Also, suggestions have greatly contributed for the betterment of our paper.

REFERENCES

[1] Annies Minu Sathiyaseelan, Vincy Joseph, Anuradha Srinivasaraghavan, "A proposed system for preventing session hijacking with modified one-time cookies", 2017 International Conference on Big Data Analytics and Computational Intelligence (ICBDAC), IEEE, March 2017.
[2] D.Madhavi," A Survey on Detection Tools and Prevention Techniques for Session Hijacking Attack", International Journal for Scientific Research & Development (IJSRD), Vol.12, December 2015.
[3] Jerry Louis," Detection of session hijacking", University of Bedfordshire Repository, Department of Computer Science and Technology, Supervisor: Dr. Xiaohua Feng, 10547/211810, AY10/11, January 2011.
[4] Neil Patel, Neel Patel, Manan Doshi, Yash shah," A review on prevention for Session Hijacking using OneTime Cookies", International Journal of Advance research Science and Engineering, Vol. 6, December 2017.
[5] Anuj Kumar Baitha, Prof. Smitha Vinod," Session Hijacking and Prevention Technique", International Journal of Engineering & Technology, March 2018

[6] Kuldeep Kumar, Dr. Debasish Jena, Ravi Kumar,” A Novel Approach to detect SQL injection in web applications”, International Journal of Application or Innovation in Engineering & Management (IJAIEM), Volume 2, Issue 6, June 2013.

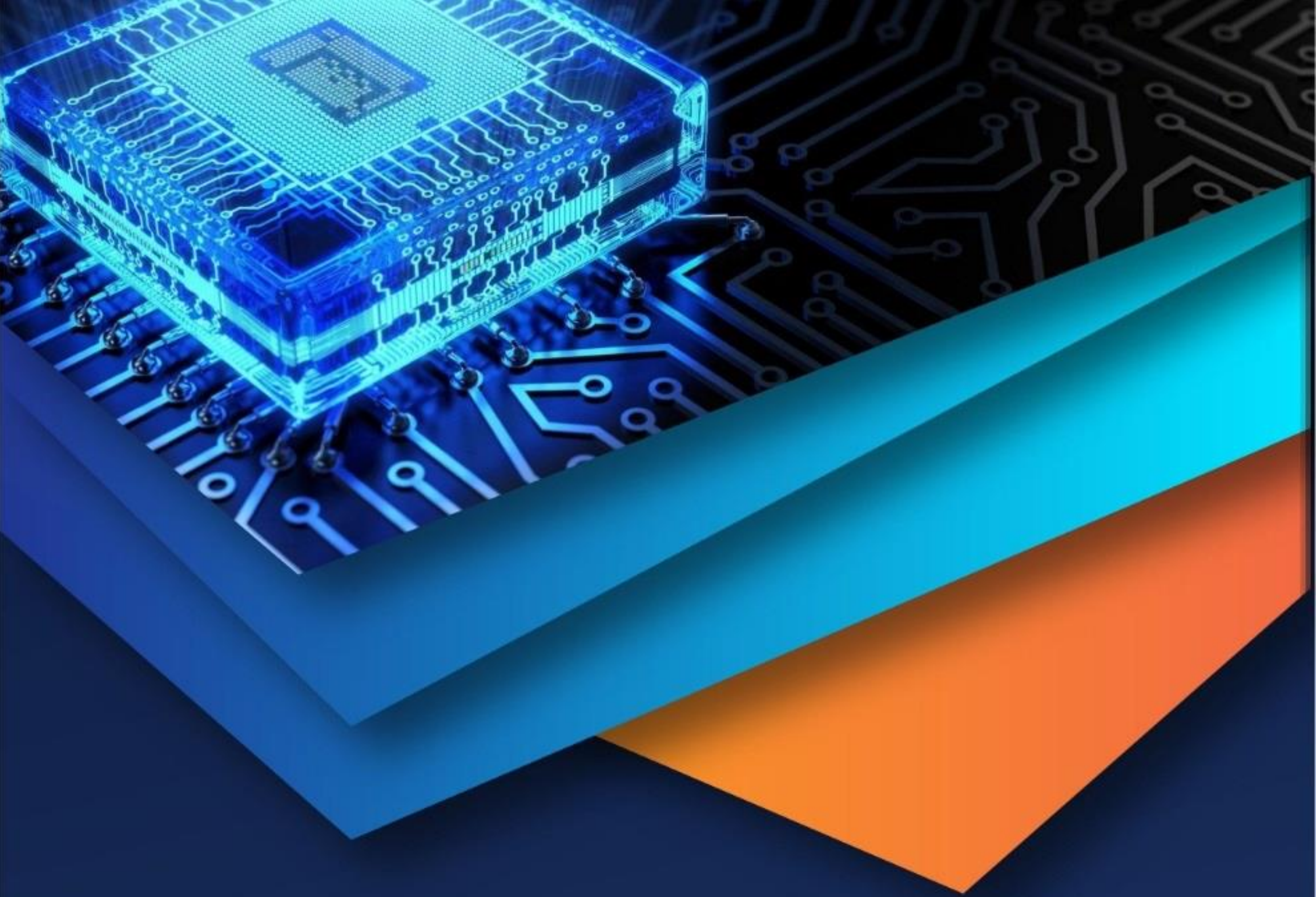
[7] Italo Dacosta, Saurabh Chakradeo, Mustaque Ahamad and Patrick Traynor,” One-Time Cookies: Preventing Session Hijacking Attacks with Disposable Credentials”, Converging Infrastructure Security (CISEC) Laboratory Georgia Tech Information Security Center (GTISC), June 2012.

[8] Karis D'silva, J Vanajakshi , K N Manjunath, Srikanth Prabhu, “An effective method for preventing SQL injection attack and session hijacking”, 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT),May 2017.

[9] Asif Muhammad, NitinTripathi, “Evaluation of Open ID-Based Double-Factor Authentication for Preventing Session Hijacking in Web Applications”, Journal of computers, Academy Publisher, Volume 7, No 11, Nov 2012. http://ojs.academypublisher.com/index.php/jcp/article/view/jcp071126232628/5787.

[10] Kefei Cheng, Meng Gao and Ruijie Guo, "Analysis and Research on HTTPS Hijacking Attacks", 2010 Second International Conference on Networks Security, Wireless Communications and Trusted Computing, 2010.

[11] Qijia Zeng, “Random Cookie protocol, a new solution to prevent against”, Helsingin yliopisto -Helsingfors universitet- university of Helsinki, May 2016.



10.22214/IJRASET

INDEX COPERNICUS
45.98

IMPACT FACTOR:
7.129

TOGETHER WE REACH THE GOAL

IMPACT FACTOR:
7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY