IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 9, Issue 6, June 2020

Detection and Prevention of Session Hijacking in Web Application Management

Israel O. Ogundele[1], Abigail O. Akinade[2], Harrison O. Alakiri[3],

Adewale A. Aromolaran[4], Benedette O. T. Uzoma[5]

Lecturer, Computer Technology Department, Yaba College of Technology, Yaba, Lagos, Nigeria[1,2,3,4]

Lecturer, Office Technology and Management Department, Yaba College of Technology, Yaba, Lagos, Nigeria[5]

Abstract: Web applications are programs that are available on the go. The increase in the number of customers accessing the web demands for technological complexity to manage the operation. The session established between the user and the server can be hijacked by an attacker by masquerading as an authorized user called Man-in-the-Middle (MITM). The target of the attacker is to have access to users' confidential records in the server for their own financial gain. It was predicted by Juniper research that by 2023 over 146 billion records will be tampered with and also electronic commerce will progressively increase by 66% in 2024 as the number on online transaction reaches $18.7 trillion. The security of Web applications have been a great concern to many online services. The paper, therefore developed a web application for e-Commerce for the detection and prevention of session hijacking in order to protect individual records from unauthorized user.

Keywords: Session Hijacking, Security, Vulnerability, Authentication, HTTP, Web Application, MITM.

I. INTRODUCTION

There are various security threats that are associated with web applications based on the transactions that takes place online daily. The dynamics of the content and functionalities of the web application have enable the users to communicate with the server effectively and displaying information through the browser platform [1]. Some of the web applications used the users for transaction purposes are e-commerce, online banking, shopping sites, online training etc. Web applications are increasing in features, programming and content [2]. The improvement in the web application is as a result of the users that interact on the web daily for one transaction or the other. Software development have also been on the increase in the release of different versions of web application that could create platform for users to interact with the server [3]. Over four billions attacks had been recorded as web application attacks between January 2018 and June 2019 [4]. Cybercriminals mostly attacks web applications in order to access user's data that are related to their financial records from the web browsers by inserting malicious program.

The web is the main sources of data breaches which takes higher percentage of compromised data in the year 2019, also web attacks can also be to commit fraud, manipulate data and to have access to internet network. It was predicted by Juniper Research that by 2023 over 146 billion records will be compromised through the web and it has also been predicted that by 2024 digital commerce will gain a rapid increase of over 66% to a rise of $18.7 trillion transaction that takes place on the web [5], [6]. Cybercriminals are focusing more in hijacking the session on the web, most especially e-commerce websites since it becoming more difficult for them to use credit cards, MasterCard or physical stores. Cybersecurity researchers identified the need to secure the web application platform from attackers due to various software tools used to perpetrate fraudulent act [7]. The attacks to an individual or organization may be hazardous because of the vulnerabilities of the websites, either to public and private sector that uses the web to run their day to day transactions.

Session Hijacking is the most critical issues confronting the web application in the current trend of technology [8], [9]. As the number of internet users keeps increasing, the vulnerabilities of the web increases also and session hijacking becomes a great concern to every owner of the web application in securing the environment. The most common attack is called man-in-middle attack (MITM) which can sometime cause denial-of-service. MITM attack attempts to hijack the session through the users platform, then tend to communicate with the server by masquerading as the actual user without any authorization. As the attacker continues to maintain communication between the browser and the server, then will inject a malicious program to gain full access to user record for his financial benefit. A user who is making an attempt to login or already logged in to the server, do so to gain complete access to the session [10] from the web browser and establishes communication with the server.

Session management in a web application ensures that user who logged in, is still the same person connected to the server and the integrity of the network is still intact. Web sessions are focus point for every cybercriminals in order to gain access to the system without having to authenticate thereby perpetrating a deceitful act on user records for financial gain [11]. In Section II of this paper, talked about previous research and related work in this area. Section III presents the methodology, architecture of the system, algorithm and system flowchart. Section IV contains the system development, discussion and result. Section V highlight precaution to take on system maintenance. Section VI conclude on the paper.

II. LITERATURE REVIEW

Session hijacking is one of the safety threats utilized by varied attackers round the world on the web. Session Hijacking plays a serious role to steal user's record and vital information that are transferred through the web. It has the potentials to takes data from the server without the consent of the users. Identity attacks are of different kinds and these uses various mode of action to tackle the challenges.

A. Session Hijacking

Session Hijacking is the most common type of attack. It is often preferred by attacker because of the penetration in accessing the session. When a user is logged in or about to login into the system and has established connection with the server, then attacker takes over the session by masquerading as if it's still the real user that is logged in. This has a great benefit to the attacker when access its gained into the server and will not have to border himself to crack the login key since he has been confirmed to have the session. Session Hijacking is the taking over of the user session id and having full control of system while the session is still in progress. The session hacking indicating the attack between the server and browser were presented in figure 1 below.

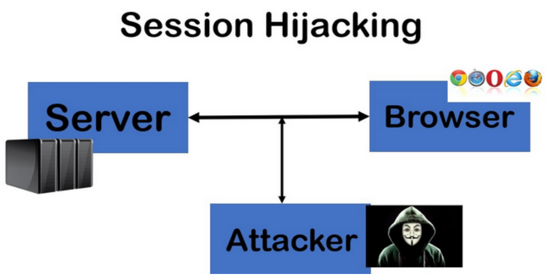


Figure 1: Session hijacking attack [12]

There are variety of ways a hacker use to steal the session ID, sort of a cross-site scripting attack accustomed hijack session IDs. An aggressor may also like better to hijack the session to insert themselves between the requesting pc and also the remote server, deceit to be the opposite party within the session. This permits them to intercept info in each direction and is often referred to as a man-in-the-middle attack. There are three types of session hijacking attacks:

i. Passive Session Hijacking: This type of attacks is the same as active one, however instead of taking out the user from the active session, the attacker supervises the traffic between server and the digital computer. Within a passive connection the hacker follows the information of the user and safe it in a self-database to the purpose of attack. It is advisable for attacker to start with passion session hijacking [13]. The figure 14 display the man-in-middle attack between the user and the server.

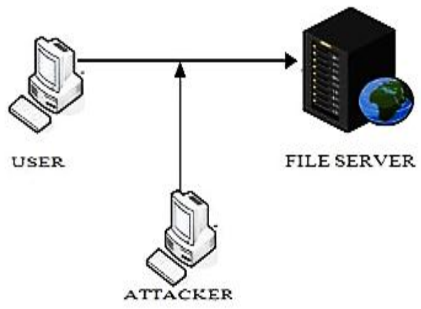


Figure 2: Passive Session Hijacking [14]

ii. Active Session Hijacking: This attack happens once the hacker takes over an active connection on a network. The hacker can mute one in all the devices, sometimes the shopper laptop, and overtake the clients' place within the communication exchange between the server and the digital computer and let go of the affiliation between the server and the user device [15]. There are different methods used to halt the connection with the server, one of the most common way is to send multiple quantity of traffic to attack the session which will cause a Denial of Service. This attack enable attacker to have full control over the session id with the server. . Figure 3 shows the actual state of affairs of the active session hijacking.

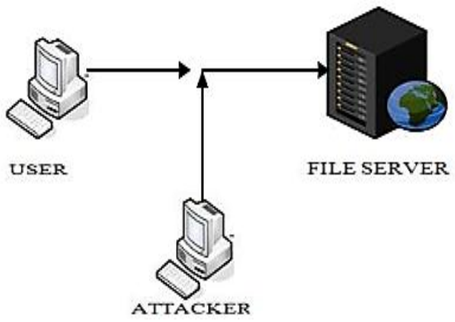


Figure 3: Active session hijacking [12]

iii. Hybrid Session Hijacking: This kind of attack is a combination of passive and active attacks that enables the assailant to concentrate to network traffic until one issue is found within it. The hacker can then make changes to the attack by taking the computer from the connection and assume their authentication.
This type of session hijacking depends on spoofing and it may be additional classified to 2 kinds namely:
• **Blind Spoofing attack:** In this type of intrusion, the assailant attacks its target without disrupting with the session (association). It merely captures all the packets between the server and connected user and attempts to crack the transmission control protocol packet sequence variety in order that it will manifest with the server [16].
• Non-Blind spoofing attack: In this type of attack the assailant will really monitor the traffic between the server and an active user. This manner it's simple for an assailant to predict consequent packet just in case it needs to forecast the TCP sequence range of consequent packet. In the application level, assailant takes over the session moreover and attempts to form new session with freshly made session token which may be taken or predicted in such method that it authenticates the assailant with the machine targeted to require over an existing connection or create new connections [14].

B. Levels of Session Hijacking

There are two levels of session hijacking, this are the platform by which attacker use to gain access to the session id [17]. They are network level and Application level.
i. **Network level:** It is the interruption of packets transmitted between the user and server through User Datagram protocol (UDP) session. The appliance level means gaining a session IDs to achieve UDP management as presented by the web application. The network session hijacking are classified as TCP Hijacking and UDP Hijacking.
ii. **Application level:** This level is concerned about hijacking established sessions but also try to create new session to take advantage. Application level of session hijacking focuses on gaining or obtaining an already established session ID by using some attack techniques, then uses the session ID to create an innovative session. It tries to work out identity in order to access the server from the application level. The session hijacker works in a way, by guessing the session ID or invading into privacy of users to steal it from their location. The application session Hijacking are classified as phishing attacks, malwares, SQL injection attack, cross-site scripting, Denial-of-service (DoS), Credential reuse.
Application layer is concerned about attacks on net as our attack concerned in universal resource locator (URL) hijacking. I analysed below the methods or techniques that may forestall this type of attacks. Some of the methods used to curtail the application attacks are making use of Strong Session ID, Generating a Random ID for each session, Using Encrypted IDs, Regenerating Token for users, Detecting Brute Forcing attacks on Session ID etc.

C. Techniques For Preventing Session Hijacking

• **Encryption**: This is mostly used by e-commerce services to encode information in order to prevent from unauthorized user. Here, a set of information is been encoded to hide the content before sending to the receiver. Encrypted data is called Ciphertext.
• **Use of a protracted random variety or string because the session key**: It is also been used to prevent application level breaches. It limit the opportunity of attacker guessing a session ID or way of trial and error or to suddenly hijack by using passphrases with the hope of guessing rightly
• **Regenerating the session id once a fortunate login**: This automatically create a session ID of the user once gained access to the server. This limit the attacker in hijacking a valid user session ID.
• **Making secondary checks:** This ensures a double check to validate the authentication of the user that logged in and the user currently using the session. The information request must matches with the right user and ensures delivery to the appropriate channel.
• **Changing the cookie value:** Some services can modify the worth of the cookie with users demand in the server. This will enable web services to spot if an attack has been launched but this can result to some technical problems.

D. Cyber Security

Cyber-protection or security plays a vital role in the ongoing deployment of information technology, and Internet services [18]. Improving cyber security and protecting critical data infrastructures are necessary to each country's financial and security welfare. The process in securing the web has become very important to organization and services and also to the government. Prevention and detection of cybercrime is germane to the government of a nation and how well to protect the web infrastructure, cyber security cannot be over emphasized to concerning the effect and financial losses. To further secure the web, it is necessary to adopt the appropriate legislation against the improper use of ICTs infrastructure for criminal or activities that could negate the integrity of the infrastructure [19]. There are five main cyber security agenda areas to be properly focused on in securing the web.

- Capacity building
- International cooperation
- Legal measures
- Technical and procedural measures
- Organizational structures

E. Tools Used to Perpetuate Session Hijacking

Part of the tools used to carry out session Hijacking are:

- T-Sight
- TTY Watcher
- Hamster and Ferret
- Wire shark
- Ethereal
- Juggernaut
- Hunt

F. Related Works

Several other studies have stressed the importance of privacy, security and usability of identity management, each focusing on specific issues or looking at the problem from a particular perspective. From the previous review work, it shows there are still vulnerabilities in an online transactions and these calls for an urgent need to put in place a high level of security in web applications in terms of privacy, confidentiality and integrity. Table 1 present some of the techniques that have been used in the previous study and the limitations of the research work.

Table 1: Related works based on the previous study.

S/N	Author	Title	Methodology	Limitation
1.	[20]	Sub-Session Hijacking on the Web: Root Causes and Prevention.	Improving protection against sub-session hijacking	There is a need to perform a large-scale analysis in securing the web application as a step ahead in improving the current system.
3.	[21]	An Effective Method for Preventing SQL Injection Attack and Session Hijacking	Hashing Technique	The technique proposed is used to prevent SQL injection and session hijacking but did not consider other Web application vulnerabilities
4.	[22]	Prevention of Session hijacking using OneTime Cookies	OneTime Cookies	Dependent on the Reverse proxy server and Only 1 session/user. This might cause a shake in the web services in creating session for web application and in establishing connection.
5.	[23]	An Analysis of Seven Concepts and Design Flaws in Identity Management Systems	Design of identity management.	It design system to identify defects in a network but focus was not on web applications
6.	[24]	A prevention model for session hijack attacks in wireless networks using Strong & encrypted session ID	Strong and Encrypted Session ID was used	This is restricted to some length of session ID to generate the encryption in Hijacking.
7.	[25]	An Analytical Study of Web Application Session Management Mechanisms & HTTP Session Hijacking Attacks	Performance evaluation of session management mechanisms & HTTP	Only consider the evaluation performance of the web management and HTTP attacks but detection of the attack of the web was not looked into.
8.	[26]	Prevention of Session Hijacking and I spoofing with Sensor nodes and Cryptographic Approach	Sensor Nodes and Cryptographic Approach	This focused on the fake access point and IP spoofing to detect session hijacking but didn't consider other aspect of hijacking.

9.	[12]	Wavelet Based Real Time Session Hijack Detection Based On Bluetooth Signal Analysis	Bluetooth signal	The focus was based on the signal received from the Bluetooth and these might not be able to detect all the vulnerabilities of experienced online
10.	[27]	Cookie-Based Virtual Password Authentication Protocol	Virtual password authentication	There is no guarantee that the user will not be able to delete the cookie from his computer so that computational cost of authentication increases for the attacker.

III. METHODOLOGY

This session is centred on the system architecture to prevent session hijacking in a web application, the various component of the system and session hijacking module was explained. The system flowchart and algorithm for the session ID creation, time out session and re-authentication were presented. The system architecture will therefore help to detect and prevent session hijacking in order to make user records more confidential, secure and reliable. Figure 4 shows the system architectural model for the detection and prevention of session hijacking in a web application.

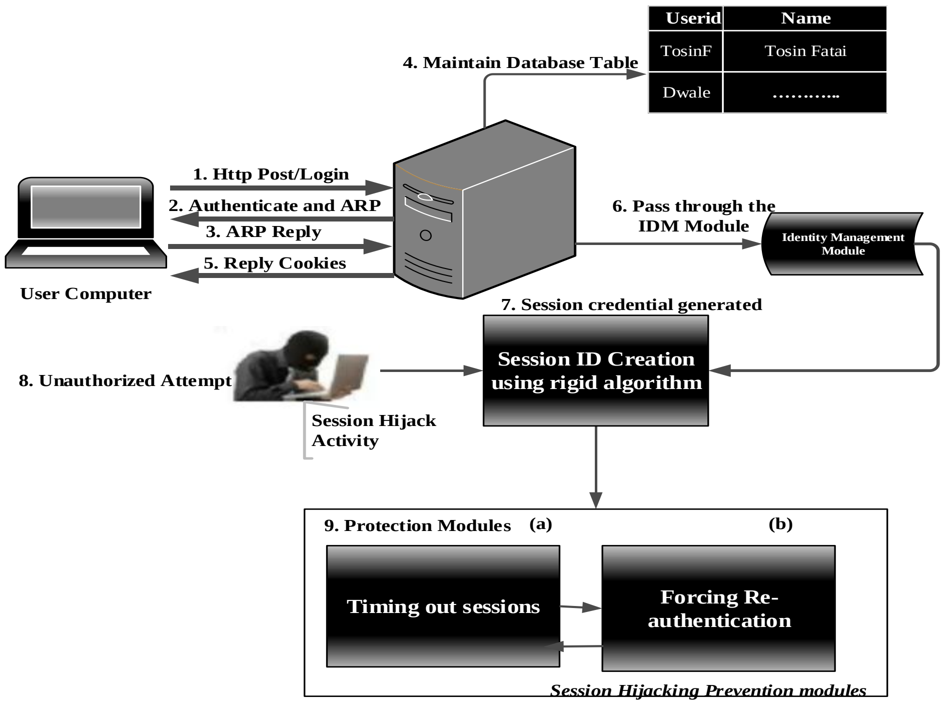

Userid
Name
TosinF
Tosin Fatai
Dwale
………..
4. Maintain Database Table
1. Http Post/Login
2. Authenticate and ARP
3. ARP Reply
5. Reply Cookies
User Computer
6. Pass through the IDM Module
Identity Management Module
7. Session credential generated
8. Unauthorized Attempt
Session Hijack Activity
Session ID Creation using rigid algorithm
9. Protection Modules
(a)
(b)
Timing out sessions
Forcing Re-authentication
Session Hijacking Prevention modules

Figure 4: System architecture for session Hijacking Detection and Prevention

A. System components

Web application are programs that runs on the web server. User of a web application uses web browser to access the server and then establish session. Hypertext Transfer Protocol (HTTP) enable the user to request for information from the server through the browser. User requests a web page from the server to fetch information and this respond to the HTTP which is identified by the session created. The session ID is uniquely identified on the web. Figure 4 and 5 maintain the database table and respond to cookies by the previous action of the user, the request consist of the session ID and also the cookies.

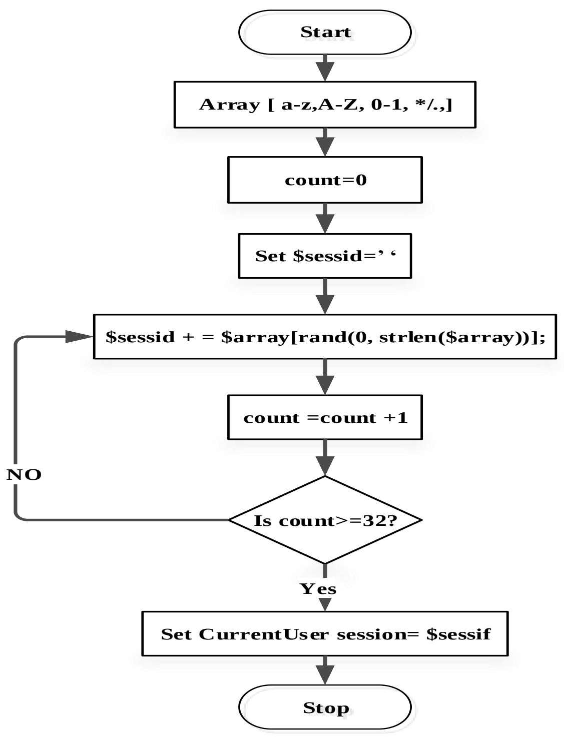
B. Session hijacking prevention module

i. **Session ID creation using rigid algorithm:** Fragile/Short session ID's can be expose to attacks. The application of cryptographic algorithm can enable us to detect the attack. The attacker can study the session ID generation to drawn knowledge in creating a new session. Hence, to avoid this risk, an algorithm to generate long random alphanumeric character is used as a session key.

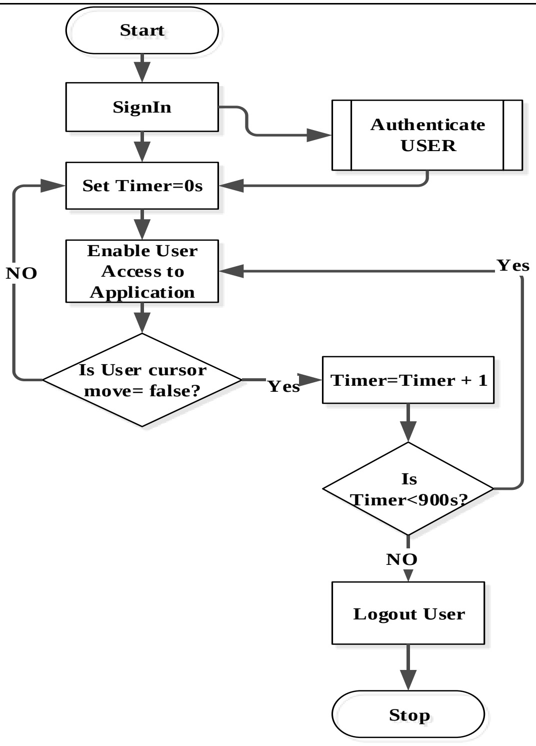
ii. **Timing out Sessions:** If the system is left idle and user didn't perform any operation on it and did not log out. Attacker can steal the session ID and thereby hijack the session. It is therefore necessary for user to timeout after a constant period of time if idleness to prevent from attack. In the system developed the timeout session is set to 15 minutes as a stronger control, leaving not enough time for the attacker to penetrate.

iii. **Forcing Re-authentication:** This module enable user to access the system after a constant period of time to re-login. Here, the session ID is recreated and connection established. The previous session ID becomes outdated and it is therefore no longer useful for the attacker. The previous session will require re-authentication which the attacker cannot have access to due loss of session after a constant period of 15 minutes.

Session ID Creation using rigid algorithm

System Flowchart for Session ID creation	Algorithm for Session ID creation
 Start Array [a-z,A-Z, 0-1, */.,] count=0 Set $sessid=' ' $sessid + = $array[rand(0, strlen($array))]; count =count +1 NO Is count>=32? Yes Set CurrentUser session= $sessif Stop Figure 5: System Flow for Session ID Creation	Step 1: Start **INPUT**: Step 2: Input username and password **PROCESS** Step 3: Define array = All lower and uppercase letters + numeric 0 to 9 Step 4: count =0 Step 5: session_id=' ' Step 6: Nextval = Randomize (array) Step 7: session_id = session_id +Nextval Step 8: increment count by 1 Step 9: If count is less than or equal to 32 goto step 6 **OUTPUT** Step 10: Display session_id Step 11: End

Time-Out Feature

Timeout Session	Algorithm for Timeout Session
 Start SignIn Authenticate USER Set Timer=0s Enable User Access to Application NO Yes Is User cursor move= false? Yes Timer=Timer + 1 Is Timer<900s? NO Logout User Stop Figure 6: System Flowchart for Timeout Session	Step 1: Start **INPUT** Step 2: sign in and authenticate **PROCESS** Step 3: Set timer = 900 Step 4: if mousemove = True goto step 10 Step 5: Decrement timer by 1 Step 6: If timer = 0 **OUTPUT** Step 7: Display timeout message Step 8: Logout Step 9: Goto Step 11 Step 10: Display index Page Step 11: Stop

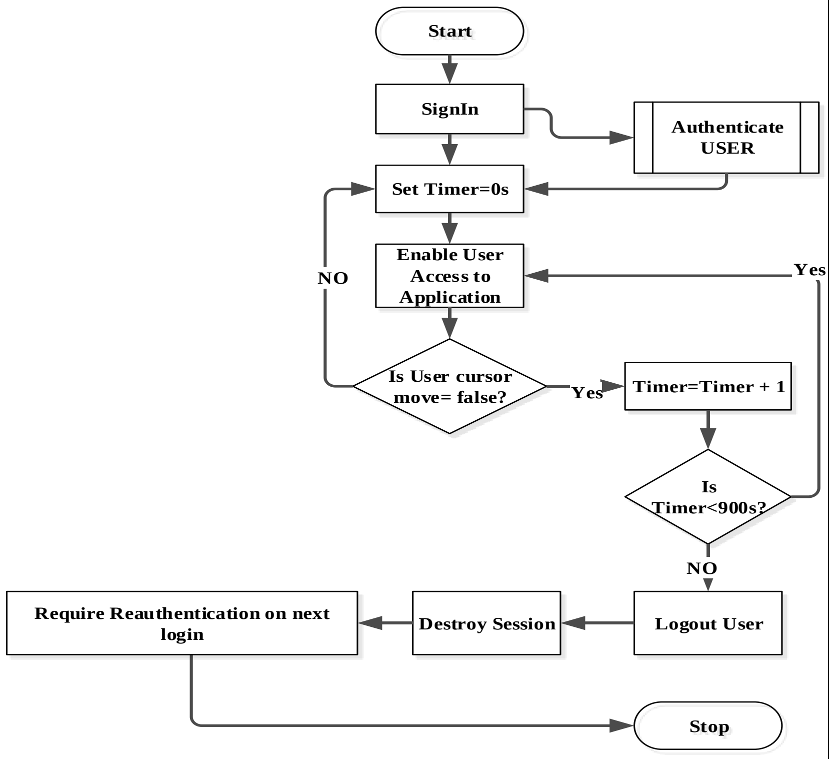
ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 9, Issue 6, June 2020

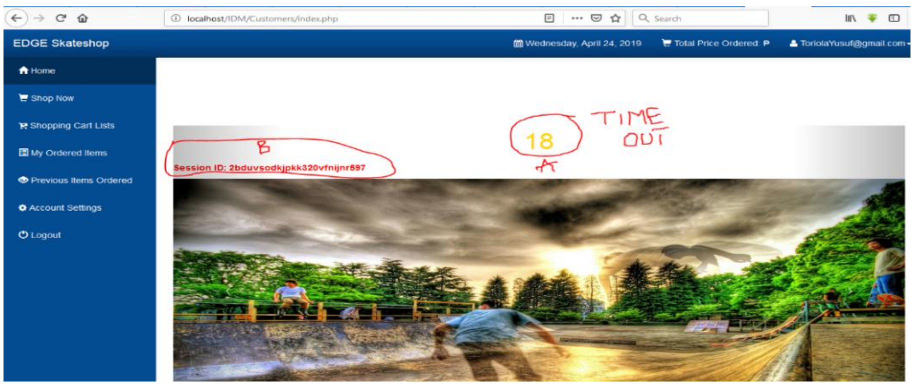
The timeout serve as a security measures to monitor the operation of user in the web. In this session each user is automatically terminated once the system discover that the cursor of the computer currently in use no longer moves for 900s (900/50) which is equivalent to 15minutes. This reduces the chances of a session hijacker taking over an idle session in case the user is distracted or leaves the platform active for a long time. Upon resumption, the user will automatically be required to sign in to continue.

Forcing Re-Authentication
The session auto-generate the session-ID so that all existing connections are closed and the user are re-authenticated to the web application without loss of records.

Forcing Re-Authentication and Session Recreation	**Algorithm for Re-Authentication & Session Recreation**
 Start SignIn Authenticate USER Set Timer=0s Enable User Access to Application NO Yes Is User cursor move= false? Yes Timer=Timer + 1 Is Timer<900s? NO Require Reauthentication on next login Destroy Session Logout User Stop Figure 7: System Flowchart for Re-Authentication and Recreation	Step 1: Start **INPUT** Step 2: sign in and authenticate **OUTPUT** Step 3: Set timer =900 Step 4: if mousemove = True goto step 12 Step 5: Decrement timer by 1 Step 6: If timer =0 **OUTPUT** Step 7: Display timeout message Step 8: Logout **PROCESS** Step 9: Destroy session Step 10: Create new session id upon login Step 11: Goto Step 13 **OUTPUT** Step 12: Display index page Step 13: Stop

IV. SYSTEM DEVELOPMENT, DISCUSSION AND RESULT

The system is developed using the following computer tools. These are HTML, PHP, CSS, JAVASCRIPT programming languages and MySQL as database. The development was based on e-commerce web application where a user/customer uses the platform to transact business. On launching the application the user is presented with the Homepage where they are required to either signup or login and enter their credentials.



Session ID: 2bduvsodkjpkk320vfnijnr597

Figure 8: Current session ID

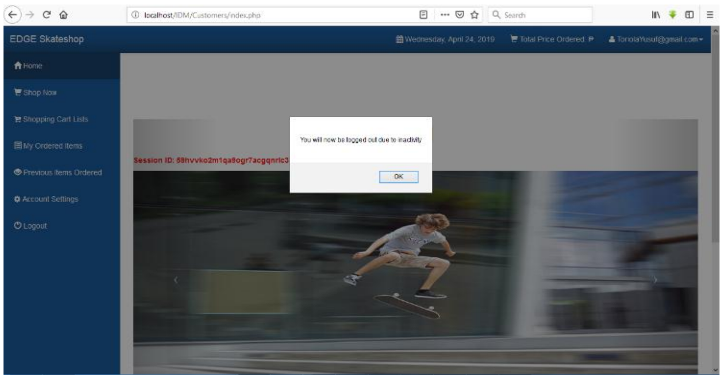
Vol. 9, Issue 6, June 2020

Figure 8 shows the current session ID assigned to the signed on user. This session ID has been set in such a way that it is not reusable by any other user at any time. The figure also shows a countdown timer on the current session if user is idle (No activity on the webpage).
Note: Current session ID of the User is circled in red in figure 8 below and Current username is Toriolayusuf@gmail.com

Figure 9a below shows the state of the website after the user has not done anything on the page for 5minutes (300/60). It is referred to as idle state and the following takes place.

i. User session is destroyed
ii. Cookies stored on all transactions are also destroyed.
iii. User will be session id will be regenerated upon next login.

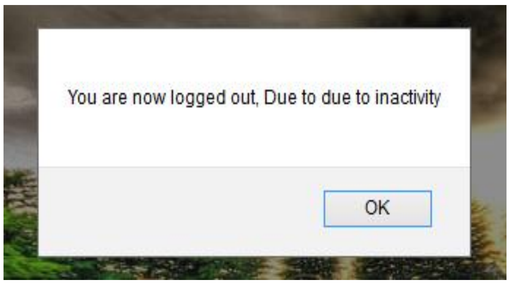
The message in Figure 9b will be displayed upon automatic sign-out.



You will now be logged out due to inactivity

Session ID: 59hvvko2m1qa9ogr7acgqnric3

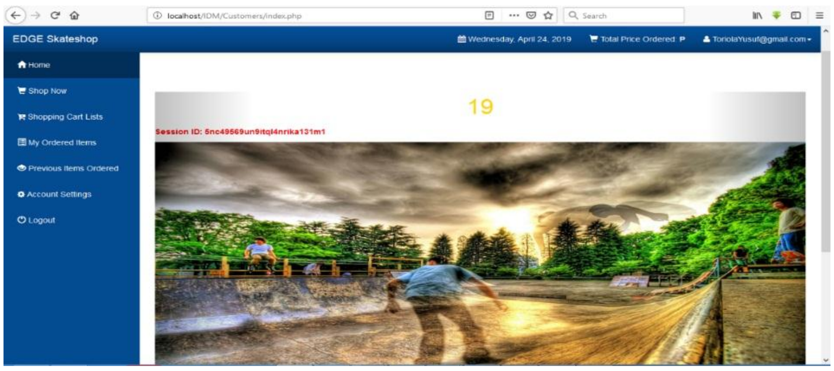
Figure 9a: Timeout when no activity



You are now logged out, Due to due to inactivity

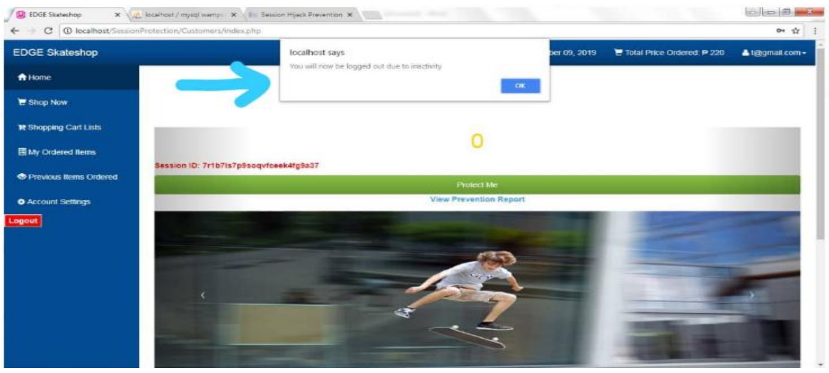
Figure 9b Timeout Message received

The Figure 10 below shows the new session id of the same user signed in on Figure 8 with proof that it has been recreated upon next login. Normally a particular user account will hold on to a session ID usually between 30minutes to 24 hours and this gives session Hijackers enough time to perform all their actions and the session is taken over. The recreation of the ID means that stolen session IDs will not be usable once it changes to another upon logoff either by the user or the timeout feature.
Note: The session ID in Figure 10 (5nc49569un9itql4nrika131m1) is different from the one in Figure 8.



Session ID: 5nc49569un9itql4nrika131m1

Figure 10: Session ID regeneration



localhost says
You will now be logged out due to inactivity

Session ID: 7r1b7is7p9soqvfceek4fg9a37

Figure 11: Auto Signout

Figure 11 shows the auto sign-out page. After timer turns zero, you will be logged out to prevent inactivity.

The system requires that a user clicks on protect me after signing on to the system. This is required to activate some internal function of the system which is not on the home page. Figure 12 shows protect me page that sign in user needs to click to secure the web browser.

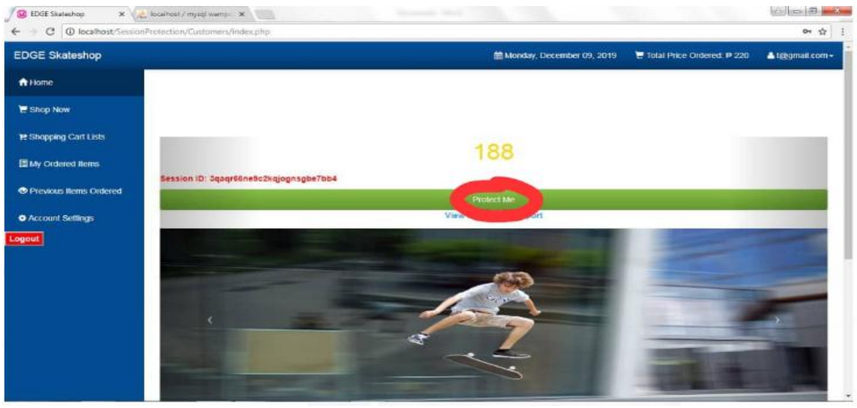
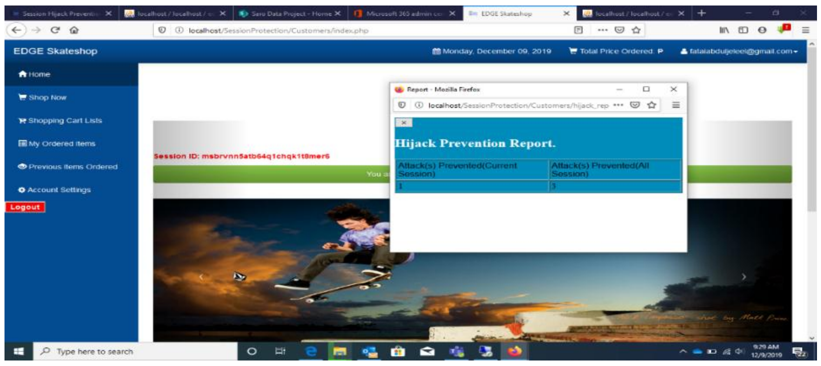

EDGE Skateshop
localhost/SessionProtection/Customers/index.php
Monday, December 09, 2019
Total Price Ordered: ₱ 220
Home
Shop Now
Shopping Cart Lists
My Ordered Items
Previous Items Ordered
Account Settings
Logout
188
Session ID: 3qaqr66ne9c2kqjognsgbe7bb4
Protect Me

Figure 12: Protect Me feature

Figure 13 shows the attack detection page where hijacking of user record is been discovered and immediate preventive measure is been taken based on the security of the web application to timeout the session and re-authenticate the user with a different session ID. This will enable optimal security from unauthorized user thereby preventing the system from session hijacking.


EDGE Skateshop
localhost/SessionProtection/Customers/index.php
Monday, December 09, 2019
Total Price Ordered: ₱
Home
Shop Now
Shopping Cart Lists
My Ordered Items
Previous Items Ordered
Account Settings
Logout
Session ID: msbrvnn5atb64q1chqk1t0mer6
Report - Mozilla Firefox
localhost/SessionProtection/Customers/hijack_rep

Hijack Prevention Report.

Attack(s) Prevented(Current Session)	Attack(s) Prevented(All Session)
1	3

Figure 13: Attack detection report

V. SYSTEM MAINTENANCE

The following are ways that could be used to keep this information system checked and in perfect condition.

i. Ensure that every user uses a password that can be easily remembered but strong and must not be revealed to ensure maximum security.
ii. The system's administrator should publish data input deadlines for the processing period.
iii. The administrator should also provide technical advice on the use of the software and on potential or proposed enhancements and development.
iv. Any deviation in the function of the system must be immediately reported to a system analyst for ratification.
v. An up-to-date antivirus must be installed on every system that uses the application.
vi. There must be stable power supply or UPS so as to be able to operate the information system as at any time required especially during result collation and checking.

VI. CONCLUSION

Session hijacking is a serious issue that every web user and organization need to place priority on in securing their information on the web. This research paper provides all the information on the vulnerabilities of using the web application and the attack that may be encountered by unauthorized user to cause damage to confidential information. The paper presented various types of session hijacking and how the attack is been done affect the operation on the web. Most of the session hijacking that occur on the web are associated with application level such as phishing attacks, malwares, SQL injection attack, cross-site scripting etc. The techniques for preventing session hijacking and the tools used by attacker in carrying out destructive act on the web were discussed. Previous literature shows that, there are still vulnerabilities in an on transactions and these calls for an urgent need to put in place a high level of security in web applications in protecting individual records from attacks. The research also presented an architectural model that were used as a blueprint for the development of the e-commerce website to optimally detect and prevent session hijacking in a web application. Session ID creation using rigid algorithm, Timing out Sessions and Forcing Re-authentication are the three modules that were discussed for the detection and prevention. The web application developed will be useful for a wide-range of users and organizations in the prevention of attack from unauthorized user having the most reliable and efficient system.

ACKNOWLEDGMENT

We want to appreciate our colleague in the Computer Technology Department and Office Technology Management Department, Yaba College of Technology, Yaba-Lagos, Nigeria for their immense contribution in the course of writing this research paper and also want to thank the management of Yaba College of Technology for providing an enabling environment and other contributors in developing the web application.

REFERENCES

[1] A. Alzahrani, A. Alqazzaz, Y. Zhu, H. Fu, and N. Almashfi, “Web application security tools analysis,” in 2017 ieee 3rd international conference on big data security on cloud (bigdatasecurity), ieee international conference on high performance and smart computing (hpsc), and ieee international conference on intelligent data and security (ids), 2017, pp. 237–242.
[2] V. S. Subrahmanian, M. Ovelgonne, T. Dumitras, and B. A. Prakash, “The Global Cyber-Vulnerability Report,” 2015.
[3] J. Offutt, “Quality attributes of web software applications,” IEEE Softw., vol. 19, no. 2, pp. 25–32, 2002.
[4] A. Razzaq, K. Latif, H. F. Ahmad, A. Hur, Z. Anwar, and P. C. Bloodsworth, “Semantic security against web application attacks,” Inf. Sci. (Ny)., vol. 254, pp. 19–38, 2014.
[5] E. Matta, “Kansans at Risk: Strengthened Data Breach Notification Laws as a Deterrent to Reckless Data Storage,” U. Kan. L. Rev., vol. 67, p. 823, 2018.
[6] F. Ullah, M. Edwards, R. Ramdhany, R. Chitchyan, M. A. Babar, and A. Rashid, “Data exfiltration: A review of external attack vectors and countermeasures,” J. Netw. Comput. Appl., vol. 101, pp. 18–54, 2018.
[7] C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis, “Intrusion detection in 802.11 networks: empirical evaluation of threats and a public dataset,” IEEE Commun. Surv. Tutorials, vol. 18, no. 1, pp. 184–208, 2015.
[8] N. Nikiforakis, W. Meert, Y. Younan, M. Johns, and W. Joosen, “SessionShield: Lightweight protection against session hijacking,” in International Symposium on Engineering Secure Software and Systems, 2011, pp. 87–100.
[9] E. Bursztein, C. Soman, D. Boneh, and J. C. Mitchell, “Sessionjuggler: secure web login from an untrusted terminal using session hijacking,” in Proceedings of the 21st international conference on World Wide Web, 2012, pp. 321–330.
[10] S. Kamuni and S. ShreehaTejaswini, “Bhaskar,” J, Dr. G. Manjunath “Wavelet Based Real Time Sess. Hijack Detect. Based Bluetooth Signal Anal. ISSN, pp. 1945–2249, 2012.
[11] M. Hoekstra, R. Lal, P. Pappachan, V. Phegade, and J. Del Cuvillo, “Using innovative instructions to create trustworthy software solutions.,” HASP@ ISCA, vol. 11, no. 10.1145, pp. 2487726–2488370, 2013.
[12] S. Kamuni and S. ShreehaTejaswini, “Bhaskar,” J, Dr. G. Manjunath “Wavelet Based Real Time Sess. Hijack Detect. Based Bluetooth Signal Anal. ISSN, pp. 1945–2249, 2012.
[13] Kamal, Parves. "State of the Art Survey on Session Hijacking." Global Journal of Computer Science and Technology 16.1 (2016).
[14] Alabrah, Amerah, and Mostafa Bassiouni. "Preventing session hijacking in collaborative applications with hybrid cachesupported one-way hash chains." Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), 2014 International Conference on. IEEE, 2014
[15] Jain, Vineeta, Divya Rishi Sahu, and Deepak Singh Tomar. "Session Hijacking: Threat Analysis and Countermeasures." Int. Conf. on Futuristic Trends in Computational Analysis and Knowledge Management. 2015.
[16] S. Sivakorn, I. Polakis, and A. D. Keromytis, “The cracked cookie jar: HTTP cookie hijacking and the exposure of private information,” in 2016 IEEE Symposium on Security and Privacy (SP), 2016, pp. 724–742.
[17] T. Chomsiri, “A comparative study of security level of Hotmail, Gmail and Yahoo mail by using session hijacking hacking test,” Int. J. Comput. Sci. Netw. Secur., vol. 8, no. 5, pp. 23–26, 2008.
[18] H. Brechbühl, R. Bruce, S. Dynes, and M. E. Johnson, “Protecting critical information infrastructure: Developing cybersecurity policy,” 2010.
[19] S. V Raghavan and E. Dawson, An investigation into the detection and mitigation of denial of service (dos) attacks: critical information infrastructure protection. Springer Science & Business Media, 2011.
[20] S. Calzavara, A. Rabitti, and M. Bugliesi, “Sub-session hijacking on the web: Root causes and prevention,” J. Comput. Secur., vol. 27, no. 2, pp. 233–257, 2019.
[21] K. D’silva, J. Vanajakshi, K. N. Manjunath, and S. Prabhu, “An effective method for preventing SQL injection attack and session hijacking,” in 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), 2017, pp. 697–701.
[22] A. M. Sathiyaseelan, V. Joseph, and A. Srinivasaraghavan, “A proposed system for preventing session hijacking with modified one-time cookies,” in 2017 International Conference on Big Data Analytics and Computational Intelligence (ICBDAC), 2017, pp. 451–454.
[23] J. J. Calixto and F. S. Ferraz, “An Analysis of Seven Concepts and Design Flaws in Identity Management Systems,” 2015.
[24] S. S. Manivannan and E. Sathiyamoorthy, “A prevention model for session hijack attacks in wireless networks using strong and encrypted session ID,” Cybern. Inf. Technol., vol. 14, no. 3, pp. 46–60, 2014.
[25] S. Wedman, A. Tetmeyer, and H. Saiedian, “An analytical study of web application session management mechanisms and HTTP session hijacking attacks,” Inf. Secur. J. A Glob. Perspect., vol. 22, no. 2, pp. 55–67, 2013.
[26] A. K. Bharti and M. Chaudhary, “Prevention of Session Hijacking and Ipspoofing with Sensor Nodes and Cryptographic Approach,” Int. J. Comput. Appl., vol. 76, no. 9, 2013.
[27] S. K. Sood, “Cookie-Based Virtual Password Authentication Protocol,” Inf. Secur. J. A Glob. Perspect., vol. 20, no. 2, pp. 100–111, 2011.