Journal of Engineering Science and Technology
Vol. 17, No. 2 (2022) 0874 - 0882

SURVEY OF BROKEN AUTHENTICATION AND SESSION MANAGEMENT OF WEB APPLICATIONVULNERABILITY ATTACK

MOHAMMAD ALAHMAD[1,*],
ABDULRAHMAN ALKANDARI, NAYEF ALAWADHI

[1]Computer Science Department, College of Basic Education, Public Authority of Applied Education and Training, Ardiya Campus, Kuwait City, Kuwait
Computer Science Department, College of Basic Education, Public Authority of Applied Education and Training, Ardiya Campus, Kuwait City, Kuwait
Senior Technical Support Engineer (MIS Department) Employment Communication & Information Technology Regulatory Authority (CITRA): Kuwait City, KW
*Corresponding Author: malahmads@yahoo.com

Abstract

Web applications are used in our daily basis every day. In recent years, web applications are exposing to security threats and breaches. Security researchers, corporates and organizations are collaborating together to stop or at least mitigate such attacks and threats. The Open Web Application Security Project (OWASP) is non-profit security organization reported and classified ten vulnerability attacks that web applications suffer from today. Broken authentication and session management vulnerability attack is the second top attack of the OWASP list report. This paper discusses and overviews the broken authentication and session management vulnerability attack by illustrating the types, examples, and prevention mechanisms to stop such attack. Also, this research provides other security layers at the top of the suggested prevention mechanisms that will enhance and strengthen the overall security system.

Keywords: Broken authentication, Credential stuffing attack, Password spraying attack, Phishing attack, Session hijacking attack, Session ID URL rewriting attack, Session management.

1. Introduction

Roughly, 4.66 billion people use Internet of 2021 year and that forms 60% of world total population "7.83 billion". In the past year, 319 million users have joined the Internet which means 875,000 users joining every day. In fact, 7% Internet users are increasing annually but this percentage is increasing since online economy is getting larger every year. Another interesting fact is that Internet users spend 7 hours averagely every day. Adding all these facts together, the worlds Internet users spend more than 1.3 billion years of human time online until 2021 year [1]. These daily mass data traveling over the Internet via web applications required security protection(s) to ensure its integrity from end-to-end users. Internet crime complaint center or simply IC3 reported the amount of monetary damage caused by cybercrimes from 2001 until 2020 as shown in Fig. 1.

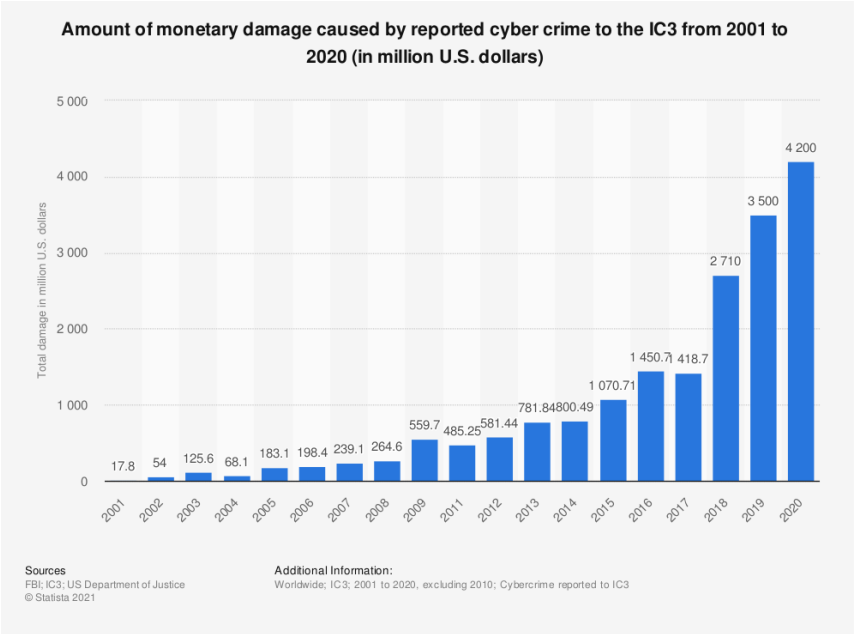

Fig. 1. Amount of monetary of damages caused by cybercrimes from 2001 until 2020 reported to IC3 [1].

As shown in Fig. 1, the tremendous amount of monetary of damages caused by cybercrimes is increasing rapidly over the last 20 years due to the vulnerabilities of computer systems and web applications. In 2020, the monetary damage reached 4.2 million dollars caused by cybercrimes. And that reflects the size of the problem and the great threat suffered by computer systems and web applications.

Today, web application vulnerability attacks attracted security researchers, communities, organizations, and companies due to the urgent need to stop or at least mitigate such attacks. The Open Web Application Security Project or simply OWASP is a non-profit organization established in 2001 to produce articles, documentations, and tools to improve the security level of web applications. OWASP created a report in 2017 which contain the top 10 vulnerability attacks of web applications. Over 500 organizations and individual contributors collaborated to

produce OWASP 2017 report. Based on OWASP industry survey and security experts' submissions, OWASP categorized the 10 most critical web application vulnerability attacks. These web application vulnerability attacks are injection, broken authentication and session management, sensitive data exposure, XML External Entities (XXE), broken access control, security misconfiguration, Cross-Site Scripting (XSS), insecure deserialization, using components with known vulnerabilities and insufficient logging & monitoring.

Broken authentication and session management is the second top vulnerability attack of the OWASP 2017 risk web applications. Broken authentication is an umbrella term that consists of more than one vulnerability that attackers use to impersonate a legitimate user identity as shown in Fig.

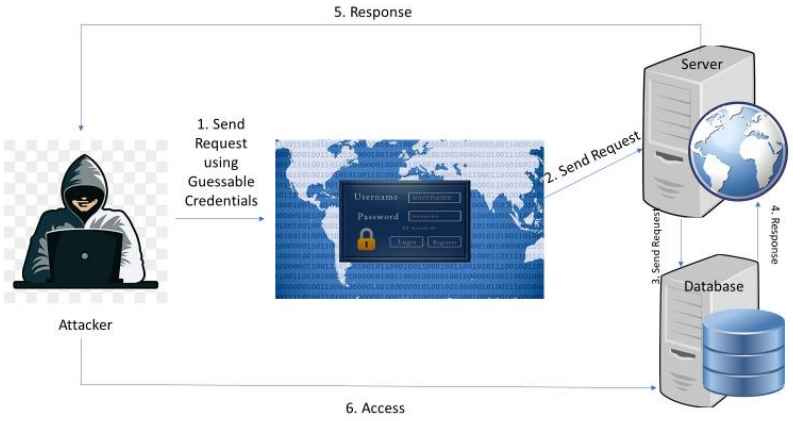

Fig. 2. General broken authentication and session management [1].

Generally, broken authentication and session management is divided into 6 steps as follows:

i. The attacker sends a request to the targeted website by guessing credentials of a legitimate user from her/his device.
ii. The website which is installed into a server sends the requested information of step 1 to the database.
iii. The database verifies the username and the password sent by the attacker
iv. It responses back to the server confirming the information validity.
v. The server sends back a response to the attacker instantiating a new session.
vi. Now, the attacker can access the database directly impersonating the legitimate user identity.

This paper analysed broken authentication and session management vulnerability attack. The analysis consists of the types, examples and how to prevent broken authentication and session management vulnerability attack. This paper is organized in six sections. Introduction and literature review are discussed in section one and two respectively. Diagnoses of broken authentication and session management vulnerability attack is explained in section three. While types, examples and prevention mechanisms of broken authentication and session management

vulnerability attack is described in section four. Finally, the paper conclusion is illustrated in section five.

2. Literature Review

There have been some research papers that conducted web application vulnerabilities. OWASP 2017 [2] reported the top ten vulnerabilities attack on web applications based on studies performed on industries and security experts to raise the awareness among developers, security communities and companies. Also, individual studies on some of OWASP 2017 top ten vulnerabilities attack report elaborated such as SQLi, Inclusion Attack [3], XSS, Brute Forcing Attack [4] and Insecure Cryptographic Storage [5] where authors suggested recommendations and countermeasures to stop suck attacks. Atashzar et al. [6] presented a survey of some critical web application vulnerabilities, hacking tools and some approaches to develop the security level of web applications and websites as well. Also, a survey is performed on 110 websites by Deepa and Thilagam [7] to detect the existence web application vulnerabilities. While Rexha et al. [8] concluded three main reasons of web vulnerability attacks: lack of experience, lack of knowledge in web security programming and neglecting of using encryption methods.

Few research papers studied broken authentication and session management attack in recent years. Al-Khurafi and Al-Ahmad [9] suggested some guidelines for developers to follow to secure the web application after analysing a code problem level of application layer and its weaknesses. While Huluka and Popov [10] explored the employment of Root Cause Analysis (RCA) in session management and broken authentication vulnerabilities to improve security level of web application. They were able to identify 11 root causes by session management vulnerabilities and 9 root causes by broken authentication vulnerabilities. Also, they suggested some effective solutions to mitigate recurrence of attacks in web applications. Murphey [11] defined session management attack and how to stop them. Also, they showed how to implement good session management using holistic defence-in-depth approach. Nagpal et al. [12] discussed a case study of a broken authentication and session management of a web application, i.e., Bangladesh web applications. Their study included a manual penetration test to examine the security weakness of their web applications and it showed the high impact of such attack.

3. Broken Authentication and Session Management

Broken authentication is an attack performed by an attacker to impersonate a user identity on online services and web applications. Authentication is broken when the attacker able to compromise a user password, session token and a user information identity. Specifically, broken authentication refers to weakness in two areas: session management and credential management. Session management and credential management considered broken authentication because an attacker can hijack a session ID or steal a user credential. After entering a valid user credentials "could be a legitimate user or an attacker" in a web application, a request will be sent from website to the web server of that application as shown in Fig. 1. The web server sends a query to its database to verify the username and the password entered by that user. Once the user credentials are matched with the record in the database and validated, a session will be established between the user and the web application with a specific unique ID. This allows the web application securely and easily to communicate with

the user as he/she move through the website. These commonly session IDs take the form of cookies and URL parameters. Based on the system administrator, the user who access that web application database will be granted privileges for getting some services of that web application. Also, a certain duration of time of the session established between the user and the web application server should be specified by the system administrator. In fact, browsers store the user credentials in an authentication cookie so the session will remain continue operating once its time duration expired by sending all the information to the server side. As a result, session management concerns how the system administrator defined the session duration. For example, how long for a session last before the automatically log out the use, or how is the IP address is linked with a session ID, and how is the administrator revoke a session ID.

Broken authentication occurred due the poor implementation of identity and access control. System administrator can diagnose broken authentication weaknesses if the web application:

i. Permits brute force attacks
ii. Permits credential stuffing attack
iii. User credentials are not encrypted when stored inside the web server
iv. Permits weak passwords
v. Do not rotate session ID after successful login
vi. Exposed to URLs rewriting
vii. Vulnerable to session fixation attack
viii. Uses public unencrypted connections to send password, credentials and session ID over them.

4. Types and Examples of Broken Authentication Attack

As stated earlier, session managements and credential managements weaknesses are both classified as a broken authentication vulnerability attack.

4.1. Session management attacks

Session managements attacks are classified into five vulnerability attacks as follows [2]:

4.1.1. Session hijacking attack

Session hijacking occurs when a verified session ID used by an attacker to impersonate a legitimate user identity. The simplest example of session hijacking attack is when a user login into his account using a public computer then he forgets to logout and walked away, then, an attacker can use the same public computer continuing the usage of that user account using the same session ID. Another example of session hijacking attack occurs when an attacker uses a session cookie, he stole to impersonate a legitimate user identity. In fact, the attacker doesn't need to find out the username and the password of that user, only the attacker needs the session cookie that contain all information required to login to the web application server. For example, an attacker might use XSS attack for session hijacking, the code might send the session key to the attacker's own website, for instance:

```
http://www.TrustedSearchEngine.com/search?<script>location.href='http://www
.SecretVillainSite.com/hijacker.php?cookie='+document.cookie;</script>
```

This code will read the current session cookie and send it to the attacker website by setting the location of the browser using location [13].

Prevention. Developers and administrator can minimize the session hijacking attack risk by

1. Using HTTPS to ensure encryption of all messages on session traffic.
2. Setting the HttpOnly attribute using Set-Cookie HTTP header to prevent XSS attack and accessing cookie session from client side.
3. Validating all incoming data.
4. Rotating and regenerating of session ID after successful login.

4.1.2. Session ID URL rewriting attack

Some browsers reject and do not accept cookie, which makes the session tracking impossible. Instead, session ID URL rewriting provides an alternative tracking solution. URL rewriting involves of rewriting session ID into a hyperlink instead of a session cookie that the user servlet sends back to the browser. Then, the server will extract the session ID from the hyperlink URL address. At this stage, an attacker can intercept the transferred hyperlink and extracts the session ID of the legitimate user. Figure 3 shows an example of a session ID URL rewriting attack.

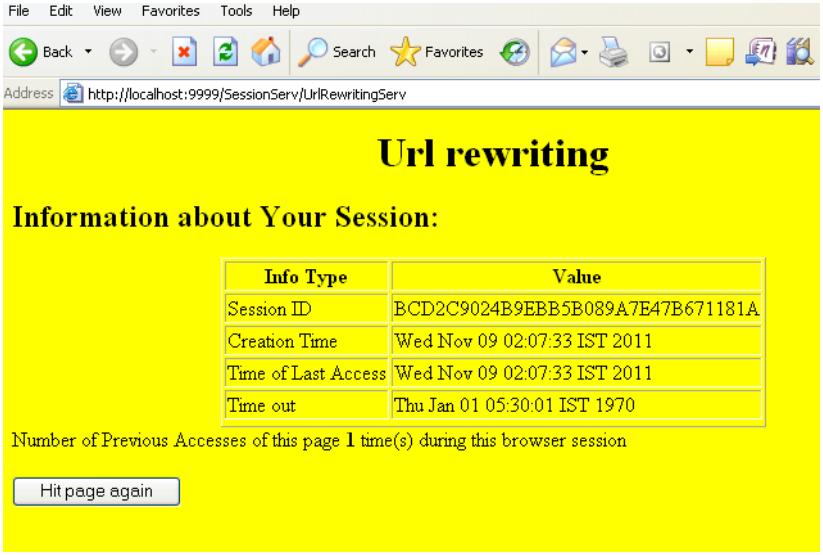

Fig. 3. General broken authentication and session management.

Prevention. Developers and administrator can minimize the session ID URL rewriting attack risk by:

i. Using cookies generated by secure session manager.

ii. Invalidating after logout or idle.

iii. Setting absolute timeout for sessions.

4.1.3. Credential stuffing attack

Using a known list of passwords is a common attack performed by attackers. Also, some attackers have an access of a database that contains unencrypted usernames and passwords. These two types called credential stuffing attack. Attackers uses these compromised credentials to gain access to other online accounts for legitimate users since most of Internet users' setup predictable and reuse the same passwords for their online accounts.

Prevention. Developers and administrator can minimize the credential stuffing attack risk by

- Not setting up the account with the default credentials or mainly for administrators.
- Implementing weak password check system to avoid users to choose predictable or weak passwords and avoid the highest 1000 worst passwords
- Limiting or delaying the failed login attempts and alert the system administrator when suck attack is attempted or detected.

4.1.4. Password spraying attack

A 2019 survey report by the UK's National Cyber Security Centre (NCSC) found 23.3 million users set "123456" as their password. While other millions use common password for example, sport names, their kids' names, their birthdays and so on [14]. Password spraying is a type of brute force attack keep trying these common passwords in a database server to gain an access of legitimate user accounts. But usually, the server will block the attacker IP address after several failed login attempts. Consequently, attackers could try to same password on several accounts rather than one single user account.

Prevention. Developers and administrator can minimize the password spraying attack risk by:

i. Enforcing a complex long password required by the web application server.
ii. Maintaining a regular awareness lectures for the company employee.
iii. Installing multi-factor authentication.

4.1.5. Phishing attack

Phishing attacks happen when attackers send "phishing" emails to users asking them their login credentials pretending to be from a trusted source [15]. While spear phishing attack is an attack technique aimed to specific target by manipulating someone emotions based on their personal information. For example, an attacker sends an email to the victim containing malicious link with subject "picture of your sister" and it is more effective if he included his sister' name. The 2020 CrordStrike Services report found 35% successful network breaches by spear phishing attack, specifically, 19% used attachments, 15% malicious links [16]. By now, corporates and organizations are more familiar with phishing attacks by warning their employees and customers from such attack.

Prevention. Developers and administrator can minimize the phishing attack risk by:

i. Educating employees and customers not to open suspicious email from untrusted source
ii. Deploying a web filter to block malicious websites by network administrators
iii. Not to share credentials with others.

5. Conclusions

Web applications are becoming more and more valuable to attackers since they are increasing rapidly in the recent years. OWASP 2017 report classified the top ten vulnerability attacks and risks of web applications. Broken authentication and session management attack is ranked number two attack on the latest OWASP 2017 report. This paper discussed and explained the types, examples and prevention mechanisms of broken authentication and session management of web application vulnerability attack. At the top of the stated prevention mechanisms, developers and network administrators are required to deploy and configure multi-factor authentication (MFA) devices. MFA devices add on another security layer to the web application network. Technically, it reduces fraud and identity theft. Indeed, web application administrators in need to follow the latest security report of a non- profit organization such as OWASP 2017 report to update their systems with the needed security batches.

References

1. Kemp, S. (2021). Digital 2021: global overview report. Retrieved April 4, 2021, from https://datareportal.com.
2. Open Web Application Security Project (OWASP). (2017). OWASP Top 10-2017. The Ten Most Critical Web Application Security Risks. Retrieved April 8, 2021, from https://owasp.org.
3. Ami, P.V.; and Malav, S.C. (2013). Top five dangerous security risks over web application. *International Journal of Emerging Trends and Technology in Computer Science*, 2(1), 41-43.
4. Petsios, T.; Kemerlis, V.P.; Polychronakis, M.; and Keromytis, A.D. (2015). Dynaguard: armoring canary-based protections against brute-force attacks. 31*st Annual Computer Security Applications Conference*, Los Angeles , USA, 351-360.
5. Kaaniche, N.; and Laurent, M.; (2017). Data security and privacy preservation in cloud storage environments based on cryptographic mechanisms. *Computer Communications*, 111, 120-141.
6. Atashzar, H.; Torkaman, A.; Bahrololum, M.; and Tadayon, M.H. (2011). A survey on web application vulnerabilities and countermeasures. 6th *International Conference on Computer Sciences and Convergence Information Technology* (*ICCIT*), Seogwipo, Korea (South), 647-652.
7. Deepa, G.; and Thilagam, P.S. (2016). Securing web applications from injection and logic vulnerabilities: Approaches and challenges. *Information and Software Technology*, 74, 160-180.
8. Rexha, B.; Halili, A.; Rrmoku, K.; and Imeraj, D. (2015). Impact of secure programming on web application vulnerabilities. *International Conference on Computer Graphics, Vision and Information Security* (*CGVIS*), Bhubaneswar, India 61-66.

9. Al-Khurafi, O.B.; and Al-Ahmad, M.A. (2015). Survey of web application vulnerability attacks. 4*th International Conference on Advanced Computer Science Applications and Technologies* (*ACSAT*), Kuala Lumpur, Malaysia, 154-158.
10. Huluka, D.; and Popov, O. (2012). Root cause analysis of session management and broken authentication vulnerabilities. *In World Congress on Internet Security* (*WorldCIS*), Guelph, Canada, 82-86.
11. Murphey, L. (2005). Secure session management: preventing security voids in web applications. *The SANS Institute*, 1-29.
12. Nagpal, B.; Singh, N.; Chauhan, N.; and Sharma, P. (2014). Preventive measures for securing web applications using broken authentication and session management attacks: A study. *In International Conference on Advances in Computer Engineering and Applications* (*ICACEA*), Ghaziabad, India, 31-33.
13. Banach, Z. (2019). What is session hijacking: your quick guide to session hijacking attacks. Retrieved April 13, 2021, from https://www.netsparker.com.
14. Crick, T.; Davenport, J.H.; Irons, A.; and Prickett, T. (2019). A UK case study on cybersecurity education and accreditation. *Frontiers in Education Conference* Covington, USA, 1-9.
15. Bin Sulaiman, R.; and Rahi, M.A. (2021). A framework to mitigate attacks in web applications. *IUP Journal of Computer Sciences*, 15(1), 22-59.
16. Fredj, O.B.; Cheikhrouhou, O.; Krichen, M.; Hamam, H.; and Derhab, A. (2020). An OWASP top ten driven survey on web application protection methods. *In International Conference on Risks and Security of Internet and Systems*, Paris, France, 235-252.