Volume: 13 Issue: 01 Pages: 4861-4873(2021) ISSN: 0975-0290

Practical Security Testing of Electronic Commerce Web Applications

P. Raghu Vamsi
Assistant Professor,
Department of Computer Science, Jaypee Institute of Information Technology, Noida, India
Email: prvonline@yahoo.co.in
Agrah Jain*
Solution Advisor, Delloitte USI, Gurugram, India
Email: agrahjain@gmail.com
* *Corresponding Author*

----------**ABSTRACT**----------
The availability of the internet and cheaper data tariffs made the effective use of Electronic Commerce (e-commerce) applications by the people for purchasing the daily needs and regular household items. The success of the e-commerce platforms is based on the trust and security that they maintain regarding users personal and payment data. However, the poor design and development, unnoticed mistakes in coding of the E-commerce websites and applications lead to many vulnerabilities and thereby becomes the simple target for the hackers. Along with conventional security testing methods, application dependent methods need to be applied on the e-commerce web applications which are built using various programming environments. To this end, this paper presents various possible practical security methods followed by penetration testers along with countermeasures that can be applicable for avoiding vulnerabilities in e-commerce websites.

Keywords - **e-commerce, penetration testing, security, testing, trust, vulnerability.**

Date of Submission: July 08, 2021 Date of Acceptance: Aug 01, 2021

I. INTRODUCTION

In today's scenario many companies are going online to sell their products via both mobile and web based applications. Nowadays, we can purchase daily needs and regular household items such as groceries, garments, decorative, vehicles, furniture and others from online platforms. This way of buying and selling over the internet is termed as e-commerce [1].

As e-commerce platforms are increasing rapidly cyber crimes and security issues are also increasing. The success of the e-commerce platforms is based on the trust and security that they maintain regarding users personal and payment data. Due to recent COVID-19 pandemic, many virus infected countries announced lockdown for the public safety. During the unlocking period, public preferring e-commerce sites to get the daily or house hold items. In this connection, existing business agencies which does not processes e-commerce sites are building up their own platforms or getting hired to be part of existing platforms. The business agencies which are already in the e-commerce field are upgrading their applications to ease the customers. In both these cases, the e-commerce website development is expected to be fast. In this process, the poor design and development, unnoticed mistakes in coding of the e-commerce websites and applications lead to many vulnerabilities and thereby becomes the simple target for the hackers. It is found that till June 2020, nearly seventy percent of the e-commerce websites on the internet faced cyber attacks, and more than nine thousand COVID-19 themed attacks faced by various websites in India [2,3-5].

However, the security attacks on e-commerce platform are not new and they are happening from starting of Web 2.0 [1]. There are more than 24 million e-commerce websites are present on the internet. According to a Forgenix survey [4, 41, 42], 75% of e-commerce websites are at the risk of some cyber attacks. Even reputed and branded companies have vulnerabilities in their websites taking advantage of them these websites are compromised by attackers. Taking the example of big brands who already have hit by any type of cyber attacks are predicted to be more than 350 million dollars during the years 2018-19 [4]. The main area to focus to curb these attacks is systematic security testing (or penetration testing) and vulnerability assessment.

Vulnerability is the weak point in the software application; by exploiting it the attacker launches security attacks to compromise the system. Due to its importance, the security testing and vulnerability assessment became a popular research area in computer networking and received significant attention from the research community. Security testing and various related issues such as privacy and others in web applications in general are articulated in recent survey articles [6-9]. Segendoet et al. [10] Analyzed and suggested improvements in e-commerce management in Ecuador by representing various security mechanisms such as personal identity theft, internal political polarization, fake news, populist and native agencies identification, cyber attacks with the

Volume: 13 Issue: 01 Pages: 4861-4873(2021) ISSN: 0975-0290

aim of data and money loss, political and economic confrontation and others. Khera et al. [11] presented a general vulnerability assessment (VA) and penetration testing (PT) process. However, it is not specific to the e-commerce applications. Rahman et al. [12] analyzed web application vulnerabilities of e-commerce sector in Bangladesh. The authors utilized web application scanners such as Acunetix and Nikto to evaluate e-commerce application vulnerabilities. This study concluded that Cross Site Request Forgery (CSRF) is evolved as the most frequent vulnerability in e-commerce websites. However, web application scanner Acunetix is a proprietary tool which requires licensing and training. A study presented in [13-15] shows that the exploring vulnerabilities depend on the type of programming environments and application specifications. Therefore, customized scanning is equally important for exploring the vulnerabilities.

In [16], the authors presented static and dynamic application vulnerability assessment using Interactive Application Security Test (IAST) method introduced by Synopsys Company. This study reported that the IAST methodology not only detects the vulnerabilities, but also spot its location in the application code. Asaduzzaman et al. [17] presented a study on core of e-commerce business, the security aspects of e-payment systems, are presented. This study presented the impact of Man-In-Middle (MIM) attacks and their prevention methods. Gautam et al. [18] presented a study of vulnerability assessment and penetration testing of web applications which are developed using .NET technology. This study focus on assessing the vulnerabilities, such as Cross-Site-Scripting (XSS), SQL Injection, private IP disclosure, and, HTTP header inspection.

All the aforementioned studies used automated vulnerability scanning methods and tools. However, these scanners are giving false positive results on modern applications where various programming techniques involved. Alternative of the automated testing is the manual testing, which is a best option for modern applications. Further, these studies focused on analyzing very limited web application vulnerabilities. To this end, this paper presents manual testing of modern applications for assessment of various e-commerce websites for vulnerabilities.

The rest of the paper is organized as follows. Understanding the importance of vulnerabilities, hackers approach to vulnerability detection, and its related counter security testing approach is presented in Section II. Tools and setup used to present the e-commerce vulnerabilities are presented in Section III. Section IV presents the security testing of e-commerce applications with suitable countermeasures. Finally, Section V concludes the paper with future work.

II. PRELIMINARIES

This section presents the preliminaries to understand the vulnerability assessment of the e-commerce web applications.

A. Understanding Vulnerabilities

When an e-commerce application is developed, many levels of testing will be carried out by the development team. It includes both the manual and automated testing. However, security testing or penetration testing is a specialized testing process, which will be performed to uncover the bugs in the application. A bug is a defect that may occur after the execution of the program. These bugs will pave a path for occurring of vulnerabilities, which are the direct target of any hacker. Bugs in an application may occur due to several reasons [19]. Some of the reasons are

1) *Coding mistakes:* it may happen as developers are under pressure to meet deadlines. Coding mistakes happen in general. Most developers are committed to developing the applications in deadline, but they do not have time to go back to verification and validation of the developed code, auditing the code, looking into what are the right testing techniques, how to design and develop the software such that it is testable and secure. Addressing these issues by a software development team may minimize the occurring of bugs.

2) *Importing the libraries:* Many software developers use third party libraries such as cryptography, image processing, statistics, and others for coding the applications. These third party libraries itself may contain bugs and the same may be imported to the application code. To avoid the bugs, it is recommended to import entire library instead of partial import. For example, heart bleed vulnerability is available in OpenSSL 1.0.1 version. Hence, when entire library is imported if any bugs exists in it, they may be rectified in the later versions. Hence, it will be easy for the developers to make modification in the application code with new library versions.

3) *Lack of clarity in library usage:* Developers may not have sufficient clarity on the use of library functions. In C programming, the old and deprecated functions *starcat(), strcpy(), sprintf()*, and others will be little confusing and some functions are null terminates, some functions returns integers etc. Hence, developers must possess complete knowledge of the libraries, inbuilt function used for coding the applications.

Based on the severity of the bug, impact level will be assigned by the security tester. These rankings are categorized as low, medium and critical. Among these, medium and critical bugs or vulnerabilities require quick fix.

Volume: 13 Issue: 01 Pages: 4861-4873(2021) ISSN: 0975-0290

B. Understanding Security Testing

An important question to be addressed is how to look for vulnerabilities or how to find them in an application. This is done via bug discovery in an application manually or via automation. There are three ways to find bugs in code which are as follows

1) *Fuzzing*: It is an automated black box testing method to discover bugs in an application. To do this, wrong, unexpected, and random input will be provided to the application. In turn, due to the input, the application may fail, crash, ends up with an exception, or run successfully. If the successful running of the application is observed, then the application is likely to be bug free. Various web application vulnerabilities such as HTML injection, AngularJS template injection, host header attack, open redirection, XSS, dictionary attacks, local file inclusion, parameter tampering, CSRF and others can be detected using fuzzing.

2) *Static Analysis:* It is the analysis carried out for identification of bugs without executing the code of application like searching for known bug patterns in code or API (Application Programmer Interface).

3) *Dynamic Analysis:* It is the security study of the behavior of the code during execution. Importantly, application code during execution may interact with other components or data sources to run the functions or to manage data. For example, a web application may interact with the back-end database for login verification. Inserting parameters to application code while executing and verifying the output by interaction with the other components is the central idea of the dynamic code analysis. From the fuzzing and static analysis, dynamic analysis requires deep knowledge of the application architecture.

This paper focuses on dynamic analysis since static analysis can be easily done on code development stage. To show how static analysis works, this paper presents a sensitive vulnerability assessment that can be found using static analysis in Section IV.

C. Understanding Hacker's Approach

Even after the general software testing process, the security team should device test cases to identify bugs mentioned in Section 2.1. Hackers will attempt to do the similar testing to uncover bugs. As bugs are the main source of vulnerabilities, when the bugs are found, the related vulnerability will be exploited and the application may be compromised. Therefore, the general steps followed by a hacker need to be understood to plan the security testing. This is because a penetration tester prepares a plan of application testing by thinking in line with the hacker's approach.

The approach of a hacker in compromising an application is as follows: the first step is to understand the functionality of the application. This procedure is said to be Reconnaissance. In this attacker will make use of all options available with the application and performs in-depth exploration to understand the application's functionality. Second step is to identify the vulnerabilities of the application. An attacker may use already available tools or device a script to identify the vulnerabilities of the application. In case, if the vulnerabilities are pre-existing, then the attacker will exploit them. Otherwise, if the vulnerabilities are identified are new (also called as zero day vulnerabilities), then they develop an exploitation script and execute it to exploit the identified vulnerabilities. However, the identifying zero day attacks require high patience and specialized skills. In general, for an application developed via systematic software development methods, checking for pre-existing vulnerabilities are sufficient for passing security test. However, an additional care must be taken in identifying vulnerabilities if the application is developed via various programming elements, libraries and multiple components. To do this, automatic testing may not be a suitable method as many paths of the application code need to test. To this end, this paper focuses on various dynamic security checks that must be used for testing e-commerce web applications as they are developed using various programming elements and interact with multiple components.

D. Understanding security tools

For performing manual security testing various tools are used in the process of finding the vulnerabilities in the websites. Some popular tools are Burpsuite [20] (which is available in both community and free version), dirb, nmap, cain and abel, sublister, aquatone, iplogger, gmap, apiscanner and others. Apart from Burp suite all others are free and open source tools available on github [21]. These tools may be downloaded and installed individually or specialized penetration testing operating systems are available with all necessary security tools. Some popular operating systems are Kali Linux [22], Parrot OS [23], Security Onion [24], etc. This paper uses various security tools and the Burpsuite tool for presenting the security testing of e-commerce web applications. This tool is used by nearly every penetration tester for performing both dynamic analysis and fuzzing. Also, its functionality can be enhanced using an extender, at a high level, the Burp extender is an API that allows you to extend Burp's functionality either within your own code or within some third party code. It supports many other extensions via Brup store tab as shown in Figure 1.

Volume: 13 Issue: 01 Pages: 4861-4873(2021) ISSN: 0975-0290

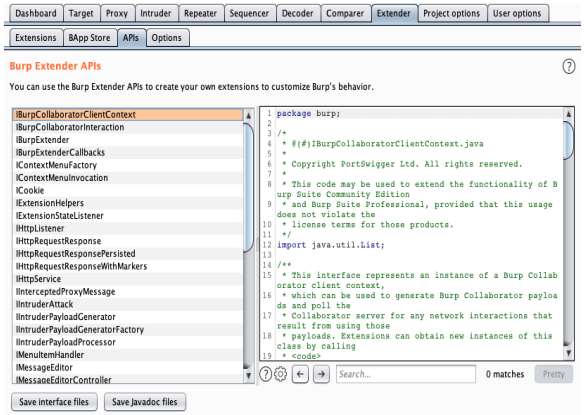

Figure 1: Functionality of Extender Tab in Brupsuite

Using Burp extender tab it can integrate with third party tools like SQLMap. The Burp Extender tab is a user interface that allows loading and managing plug-ins. With this either loading extensions from the Brup store written by other programmers or loading extensions written from application code can be done. Burp plug-ins can be used to modify HTTP requests and responses, perform tedious tasks such as specialized fuzzing, or customizing Burp UI with new tabs specific to required plug-in.

Figure 2 shows the Burp intruder tab with all the functionalities of adding a parameter to fuzz between dollar symbols, adding payloads in payload tab, etc. Penetration tester could also add custom intruder, proxy (which helps in dynamic analysis), or scanner checks to name a few of the possibilities of what pen-tester can do with Burp plug-in.

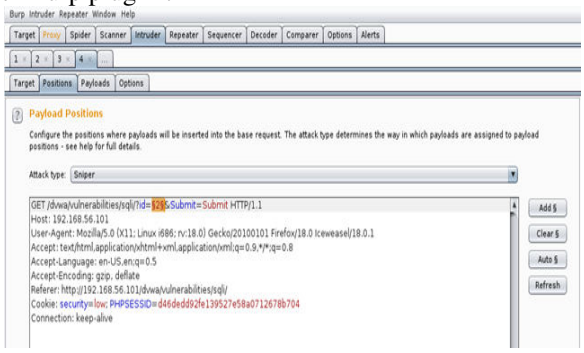

Figure 2: Attacking with Brupsuite Intruder Tab

Burp also provides a way to perform directory brute-forcing attacks using a professional-only feature called discover content. Pen-testers can use this feature with custom word lists in order to better tailor brute-force attacks to the targeted system. Brute-force attacks hope to uncover hidden functionalities such as unlinked folders, discovery of unmapped administrative console pages etc. Discover content is a professional-only feature that allows pen-testers to brute force with custom word lists.

III. RECONNAISSANCE OF E-COMMERCE WEBSITES

Reconnaissance (or Recon) is the most important phase in finding the security issues in any website. Sometimes in the Recon phase itself pen-tester can find severe vulnerabilities in the website like sensitive data exposure, exploited related to a particular version, license key leakage etc. Recon can be done in both manual and automated modes. This section presents Recon of websites with various popular tools.

A. Scanning website using nmap tool

First step in identifying vulnerabilities is scanning a website. The objective of scanning is to know the OP address of the webserver, application server, services running on server system, open and closed ports, firewall and intrusion detection systems (IDS) status, operating system information and others. Nmap is the tool which is used by almost all the pen-testers, this tool is available in both graphical user interface which is named *zenmap* and a command line interface. It helps security analysts to do host discovery, banner grabbing, testing against the script, firewall detection, and different types of port scanning. The following command an example to scan with nmap

 # nmap –T4 –v –PN –n –sS –top-ports 100 –max-parallelism 10 -oA nmapSYN 192.168.1.1

The nmap command followed by the T4 argument specifies the speed that nmap will execute. The value 1 is the slowest and 5 is the fastest. The option -v specifies verbose mode, -pn option means to nmap not to ping to identify the active system. The -n option means that no DNS resolution will be made. -ss flag, which means that it is a SYN packet type of scan [25], followed by the top 100 ports to scan. The option max-parallelism means maximum possible thread that will be equal to the number given by the user. The --oa switch followed by the file name outputs the results to file formats such as Normal, grippable, and XML. And finally the target IP address of e-commerce website. Apart from this nmap can be used to perform different types of scans like TCP, UDP, SYN, ACK etc. Table 1 shows commonly used switches used with nmap. In this way system scanning will be performed.

Table 1: Commonly used switches with namp

Switch	Description
-sS	TCP SYN port scan
-sT	TCP connect port scan (default without root privilege)
-sU	UDP port scan
-sA	TCP ACK port scan
-sW	TCP Window port scan
-sM	TCP Maimon port scan

B. Scanning for hidden directories

Finding hidden directories is also common things that hackers look for since hackers can get some important information, credentials, directory listing and sensitive data in directories since it is very difficult to navigate

Volume: 13 Issue: 01 Pages: 4861-4873(2021) ISSN: 0975-0290

through different pages. To ease this renowned tool called dirbuster (Dirb) can be used. *Dirb* is a tool, which is inbuit in Kali Linux, used for hidden directories retrieval. Along with directories it also displays the status code which can be 1XX, 2XX, 3XX, 4XX, 5XX. This tool can also brute force directories with supplied wordlist and can save the result in the file. The command used to do this task is as follows

 # dirb https://hackthissite.org /wordlist_location.txt

The best thing about this tool is it also has its own pre-defined word list Pen-tester can use prepared wordlist or can also download payloads and wordlist from *wfuzz* github repository [26].

C. *Sub-domain enumeration*

It is one of the important steps since there are more chances of finding the vulnerabilities in sub domains as compared to the main domain so hackers usually focus on sub domains since they are easy to exploit. *Sublister* is the tool use for finding sub domains of the target website. This is the python based OSINT (Open Source INTelligence) tool used to find sub domains of the target website. There can be much vulnerability associated with sub-domains. The most famous attacks are sub-domain take over, sub domain hijacking, running vulnerable services, etc. The command used to enumerate sub-domains are as follows

$ python sublist3r.py -d hackthissite.org

Figure 3 shows the sub domain listing after executing the above command. Along with the tools mentioned in this section, Google Hacking Database (GHDB) is another resource widely used by pen-testers. GHDB is also known as Google dorks which means using the Google search engine for Recon purpose to find the sensitive information using Google search queries in a smarter way.

Figure 3: Sub domain enumeration using sublister3

Consider the following 2 examples of a Google dork to understand how hackers make use of it.

site: website/auth.html intitle:login

site: website.com inurl:adminlogin.htm

The above are the example published by us in GHDB. Figure 4 shows our contribution in Exploit database (exploit-db) [27]. Google dorks is the message to Google which asks the Google search engine to return all listed and reserved assets identified with the space model *.com*.

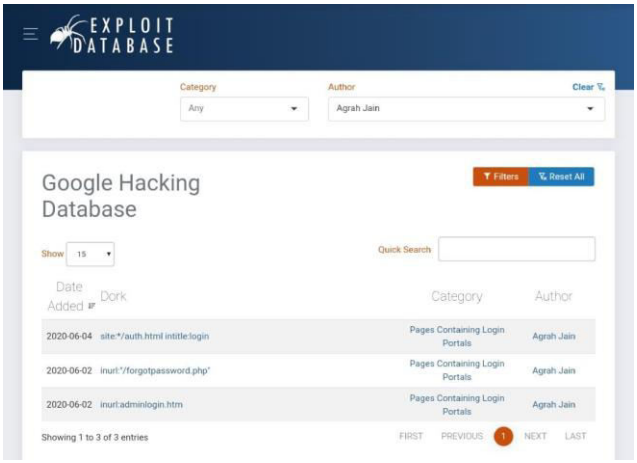

Figure 4: Snapshot of our contribution in Google dork

With this, pen-testers need to experience the rundown of returning results to check whether the delicate information has been stored. In addition, pen-tester can enter progressively exact inquiry directions in Google Search, and pen-tester can reveal to Google that he need just the outcomes, for instance, a string token in the URL. For this reason, pen-tester has to utilize a catchphrase, *inurl:* and afterward pen-tester has to enter intriguing string such as token. This is the means by which pen-tester can utilize Google Search to check if some important information from web application has been reserved by Google.

IV. FINDING VULNERABILITIES IN E-COMMERCE WEBSITES

This section presents findings of the medium and critical vulnerabilities in e-commerce websites. These vulnerabilities are discussed by OWASP (Open Web Application Security Project) and some of them are listed under OWASP top ten vulnerabilities. This section discusses cross site request forgery, sensitive data exposure, cookie attacks using XSS and headers, authentication issues, dictionary attacks, transport layer protection, heart-bleed vulnerability, mixed content vulnerability, insecure password storage, AangularJS injection, HTTP parameter pollution (HPP), click-jacking, Insecure Direct Object Reference (IDOR). Along with vulnerability assessment, this section also discusses the countermeasures which can be followed to secure an e-commerce website from these vulnerabilities.

Volume: 13 Issue: 01 Pages: 4861-4873(2021) ISSN: 0975-0290

A. *Cross Site Request Forgery (CSRF)*

In this attack, hacker tricks the internet user via sending e-mails, messages, etc., and makes normal users to click unwanted links or executable to take control of the ongoing web application session on the user's computer. When the control over the application session is taken by hacking, he may perform state changing, fund transfer, e-mail address modification, or may compromise the whole web application.

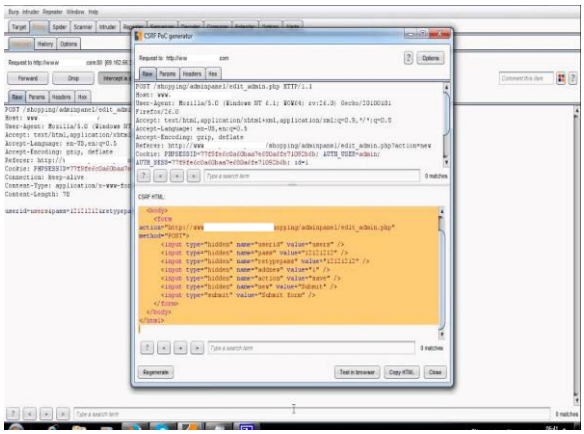

Figure 5: Generating CRSF POC using Brupsuite

Figure 5 shows the screenshot of how Burp's intercept option is turned on to perform the specific action from browser to test for CSRF. In the workspace, right click on the intercepted request and generate CSRF POC (Point of Contact) from engagement tool, drop the existing and the further request and then open that CSRF POC HTML (Hyper Text Markup Language) file in pen-tester's browser. If this action is getting completed using POC then vulnerability exist and attacker can easily manipulate that HTML file.

Now the question is what kind of data can be changed by the attacker as a result of a CSRF attack? Here are some examples: a user's email, a user's password, a user's profile image, and others. A CSRF attack changes the integrity of the data in the user's account and hence it is considered as a dangerous vulnerability.

Countermeasure: The countermeasure to this vulnerability is implementing anti-CSRF token. This token is assigned to the user and should be appended to the outgoing request. This token should be long and unpredictable. When the request with the anti-CSRF token is received by the web application, then the web application first verifies the anti-CSRF token. If the anti-CSRF token is correct, then the request is processed by the web application, otherwise, the request is rejected by the web application. As the attacker does not know the anti-CSRF token of the user, he cannot reconstruct the legitimate request that changes the user's data in the web application hosted on another domain. Hence, in this way the CSRF attack can be prevented.

B. *Sensitive data exposure*

It is one of the most important vulnerabilities. It happens quite frequently as too much information is returned in error messages of a web application. Such error messages are called verbose error messages. Attackers look for verbose error messages as they can extract a lot of interesting information from these messages. Information disclosure via metadata is another type of sensitive data exposure. It is a bit tricky and is often overlooked by the security team. When a user reads a file in a standard way, then he does not see the metadata. The fact that metadata is not seen means that there is no metadata in the file. The metadata will be available with every file, but it is hidden. The components such as robots and directory listing are very helpful to the attacker in finding what files are stored in the given directory; thereby the attacker does not have to guess the names of the files.

Further, processing of a file with nonstandard extension can be very risky. Such a file is processed by the default handler, which returns the content of the file; it can lead to disclosure of very sensitive data. Information disclosure via metadata can lead to very severe consequences. One of the most underestimated types of sensitive data exposure is disclosure of programming variants are exceptionally useful to the attacker. They may revel effectively that how an attacker can go from the revelation of programming flaw to remote code execution on the creation server. e*xif* is a tool used to identify what kind of sensitive information can be found in the metadata. Figure 6 shows the screenshot of how *exif* tool can extract information from an image file.

Countermeasure: It is advised to delete the sensitive metadata before the document is published, sanities inputs, never disclose the software and service versions that the application is using since it is easy to find CVE (Common Vulnerabilities and Exposures) [28] corresponding to it and also exploits can be found on exploit-DB with versions and services.

C. *Cookie based attacks*

A cookie is a small text that resides on the client desk for storing frequently accessed information during web session. It resides in the system and maintains by the web browser to store information such as session identity, etc. In practical terms an attacker can impersonate a legitimate person when there is an exposure of a cookie with session-ID over HTTP. Cookies with private information can leak over unsecure HTTP, even if the web app is backed by HTTPS (Hyper Text Transfer Protocol Secured).

Volume: 13 Issue: 01 Pages: 4861-4873(2021) ISSN: 0975-0290

```
Agrahs-Air:Desktop home$ exiftool Canon_40D.jpg
ExifTool Version Number         : 11.85
File Name                       : Canon_40D.jpg
Directory                       : .
File Size                       : 7.8 kB
File Modification Date/Time     : 2020:07:07 00:15:20+05:30
File Access Date/Time           : 2020:07:07 00:15:26+05:30
File Inode Change Date/Time     : 2020:07:07 00:15:24+05:30
File Permissions                : rw-r--r--
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
JFIF Version                    : 1.01
Exif Byte Order                 : Little-endian (Intel, II)
Make                            : Canon
Camera Model Name               : Canon EOS 40D
Orientation                     : Horizontal (normal)
X Resolution                    : 72
Y Resolution                    : 72
Resolution Unit                 : inches
Software                        : GIMP 2.4.5
Modify Date                     : 2008:07:31 10:38:11
Y Cb Cr Positioning             : Co-sited
Exposure Time                   : 1/160
F Number                        : 7.1
Exposure Program                : Manual
ISO                             : 100
Exif Version                    : 0221
Date/Time Original              : 2008:05:30 15:56:01
Create Date                     : 2008:05:30 15:56:01
Components Configuration        : Y, Cb, Cr, -
Shutter Speed Value             : 1/166
Aperture Value                  : 7.0
Exposure Compensation           : 0
Metering Mode                   : Multi-segment
Flash                           : On, Fired
Focal Length                    : 135.0 mm
```

Figure 6: Metadata extraction from a file using Exif tool

A sensitive cookie's value attribute should always be a secure attribute, then only the leakage of a cookie over an insecure HTTP is no longer possible. It is because the cookie with a secure attribute sent over the secure HTTPS protocol. With the leakage of critical information via referrer header, the attacker can act as a user when there is a password reset link present in the referrer header to an external domain. Even two-factor authentication will not help if cookie with session ID is insecurely processed.

One of the most underestimated cookie attacks are XSS (Cross Site Scripting) via cookie. XSS via cookie is not only results to local exploitation, but the attacker can also set a cookie remotely in a cookie jar of the user and then when the domain is visited by the user, XSS via cookie is automatically executed. The attacker can use cross origin exploitation to launch XSS via cookie remotely. An XSS attack on one sub-domain can be used to launch remotely XSS via cookie on another sub-domain. Some of the payloads for XSS are used as follows

<script>alert("Xss")</script>
"><script>alert("Xss")</script>
"><img onmouseover=alert("Xss")>
"><test onclick=alert(/Xss/)>Click Me</test>
"><a href=javascript:alert(/Xss/)Click Me

Figure 7 shows the snapshot of stored XSS pop up in one website on which we have injected *<script> alert (document. cookie) </script>* payload in the name field of the user form.

Countermeasure: To Prevent the XSS, proper sanitization of special characters need to be done. Regeneration of cookies with sensitive data is very important to implement. Disabling external entities processing will also prevent XSS.

D. Authentication Issues

The one of the main and important bug that comes under this category is SQLi (Structured Query Language injection) attacks [29].

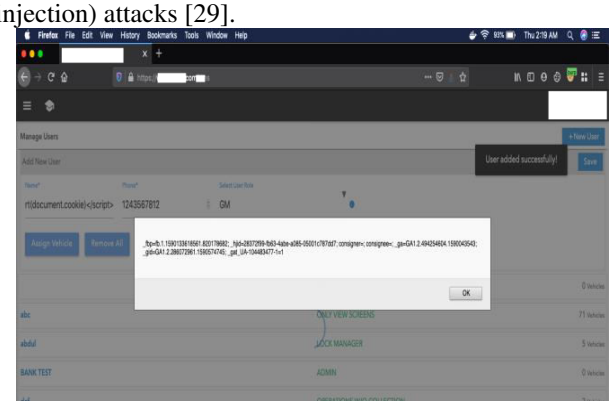

Figure 7: Stored XSS on live website

SQL query is a question fired on database to get desired results. SQLi attack can be done in various ways such as classic SQLi, blind SQLi, Boolean based SQLi, Time based SQLi. This paper focus on classic SQLi. SQLi is about changing the underlying SQL query. or logic of the query. It can lead to very severe consequences such as authentication, data leakage, and others. In practice, test with payloads in the input field that can be best done using fuzzing through Burpsuite. For fuzzing payload, required list can be taken from wfuzz or can make own query. Some basic SQL injection payloads are as follows:

" OR "" = "
' OR '1'='1
' OR ' ' = '
'=0--+
' AND id IS NULL; --

The following SQL query shows use of the payload

*select * from employee where empid = 1001' OR '1' = '1'*

The above query will return details of all employees even though there is no employee with id 1001. It is because of the OR condition. Although the first condition is false, the second condition (which is after OR) will always true. As a result, entire query will become true and the details of all employees will be returned by the database.

Countermeasure: To prevent this attack, user input data need to be sanitized. If it is done then the data coming from the user cannot change the logic of the underlying SQL query. Also to get user to be protected from Authentication attacks, it is recommended to use two factor authentication. The two-factor authentication saves user even after first password is compromised.

E. Dictionary Attacks

Consider the following case to understand dictionary attacks. Assume that Rob has a login account in a web application with email (rob@example.com) and password. Attacker goal is to learn Rob's password. For this, attacker

Volume: 13 Issue: 01 Pages: 4861-4873(2021) ISSN: 0975-0290

may prepare the following steps to make attack happen. First, prepared a list of commonly used passwords (also called as dictionary of passwords), and attacker tries to find Rob's password on this list. This may be a reasonable assumption because the majority of people do not use strong passwords. Attacker may prepare an automated script to login to the web application with Rob's email by trying every single password from this list. In addition to this, the script has to know whether the correct or incorrect password has been found. Attacker observes this via web application response. Dictionary attacks can be easily done using a tool called *hydra*, and the command to process is as follows:

 # hydra -L user.txt -P pass.txt 192.168.1.108 ftp -o results.txt

Here, usr.txt contain list of usernames and pass.txt contains a list of common passwords. The success or failure of the login attempt will be stored in result.txt

Countermeasure: To get rid of any type of brute force or dictionary attacks on e-commerce websites best practice is to put a rate limit on every sensitive action. Also implementing IP based blocking which is if someone is trying to make requests more than a limit then simply block that IP for some hours or for a day.

F. Transport Layer Protection

HTTPS is HTTP plus transport layer protection as a result of transport layer protection and it is used for securing the communication channel between the browser and the web application. The data transmitted between the browser and lower layers of the protocol stack cannot be read and modified by the attacker sitting in the middle of the communication channel. And there are two protocols that are used to secure this communication channel such as SSL (Secured Socket Layer) and TLS (Transport Layer Security) [30]. HTTPS is secure provided the transport layer protection is configured properly. It is because of different types of problems with the transport layer protection. The first problem is use of insecure protocols at transport layer. In this case, the attacker can launch a pool attack and as a consequence attacker can read the data that was supposed to be protected by SSL. Another problem is use of insecure cipher suites. This cipher suite may consist of insecure RC4 [31] chiper compliments and as a consequence that attacker can recover the plain text from encrypted connections. The next problem is related to an invalid certificate. The certificate might have expired or it has been issued with an insecure signature. This may end up in more problems with the transport layer of protection.

Countermeasure: It is recommended to know full details to configure the transport layer of protection. In other words, it is essential to know what the secure protocols are and secure cipher suits. Pen-tester may utilize a great online website (www.ssllabs.com) to check if there are any problems with transport layer protection in your Web application.

G. Mixed Content Vulnerability

Mixed content vulnerability arises when every content in a web application not backed by HTTPS. It means in many web applications, basic HTML is backed by HTTPS, but the supporting contents such as multimedia is not loaded on HTTPS. Consider the Figure 8 for mixed content vulnerability in an e-commerce web application with the URL *https://mixcontentexapmples.com/Test/NonSecureCSSLink* is viewed via Google Chrome web browser developer tools option. It was loaded over https but requested an insecure CSS (Cascading Style Sheet) i.e., styles.css content is not served over https.

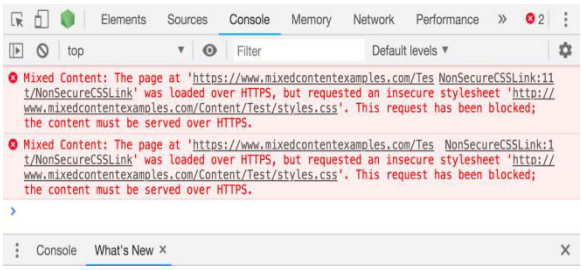

Figure 8: Mixed content vulnerability demo

Countermeasure: The countermeasure for this vulnerability is manual checking of web page when it is loaded in the browser. From Figure 8, it can be noticed that the occurrence this vulnerability has been shown with line number by the browser. With this, pen-tester needs to inspect how this vulnerability looks like in the source code. That is, are the images are loaded over insecure HTTP, and are they included on an HTTPS protected page.

H. Session Randomness Analysis

Any user login session is backed by random session number called session ID. It is a very sensitive piece of data. When the user is authenticated, the only piece of data that is used by the Web application to recognize the user is a cookie with session ID, session ID should be long and unpredictable. If the attacker can predict this session ID, then that attacker can gain access to the session of other users. Pen-tester should perform session randomness analysis in order to check if session IDs have been generated by a secure cryptographic random number generator. It is important to analyze the randomness of a session, to this end, pen-testers can experiment this concept using *sequencer* option in *Brupsuite*. The randomness analysis should not only be performed for session ID, but it is recommended to check the randomness of all other tokens and API keys in the Web application.

Volume: 13 Issue: 01 Pages: 4861-4873(2021) ISSN: 0975-0290

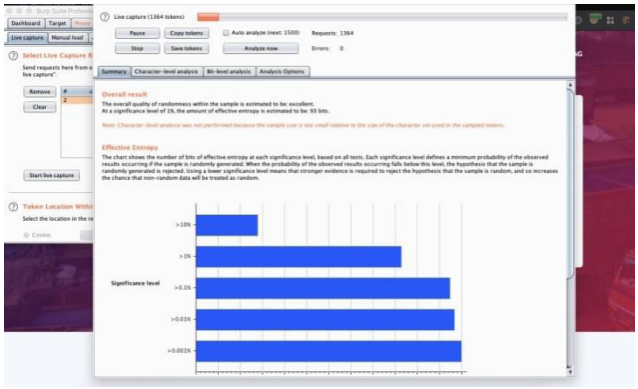

Figure 9: Session Randomness Analysis in Brupsuite

Figure 9 shows the session randomness analysis using Brupsuite. To do this, open a website in the browser and turn on the intercept in Brupsuite. And do the task which generates the token (mostly login) capture that request in Burpsuite, right click and send request to sequencer. In sequencer tab select the token from response so that burp can regenerate the specific thing in the response. Then after click on *Analyze Now* option to see lot of details about the token as shown in the screenshot. The summary shows the quality and entropy of the token also pen-tester can do character and bit level analysis of the token.

Countermeasure: Using Brupsuite, pen-testers may check Session Ids, API keys, tokens for high randomness such that it must be very hard to guess by the attacker.

I. Insecure Password Storage

This is all about storing the password in encrypted form. Assume that an attacker gained unauthorized access to a Web server. If the user's password is stored in plain text, then the attacker can read these passwords and thereby application can be compromised. This is the reason the passwords should never store in plain text.

Countermeasure: To avoid this, it is recommended to store the hash of the password. A hash of the password is created by a cryptography hash function, for example, Secure Hash Algorithm 256-bit (SHA-256) [32]. Generally, two different passwords will have different hashes. Further, a hash of the password is irreversible. Therefore, attacker cannot learn the password from the hash. But the question is that how legitimate user login if database using hashing since hash is irreversible. The database is stored only with hashes, however, when a user supplies password, web application computes hash on the password and matches with the hash in the database. If the match is successful then login takes place else login is rejected.

J. Heartbleed Vulnerability

Recall that for transport layer protection (Section 4.6) the protocols and cipher suits are important. This cipher suits are implemented using cryptographic functions imported from libraries. But, these libraries do may have vulnerabilities. Heartbleed vulnerability is one of the most famous vulnerabilities in modern crypto library called openSSL [33]. This vulnerability is identified as very dangerous because it allows the attacker to read sensitive data such as user's credentials from the memory of the website page. It turns out that the attacker can exploit this vulnerability remotely with publicly available scripts. A very good script for detection and exploitation on *exploit db* [27] or *pentestemonkey* [34] and *pentester. land* [35].

Figure 10 shows the working of Heartbleed vulnerability [36]. If attacker finds that there is heartbleed vulnerability he can use the existing payload available on explit-db and can perform any task like information retrieval to remote code execution. This exploiting works as follows: attacker sends a small payload (say 64kb) stating it as of much bigger size (say 128kb). After request gets completed this payload is written in to memory then when attacker again queries for payload and sends back the actual payload along with additional information. Since the queried payload is more than the requested size, bleed in the memory takes place.

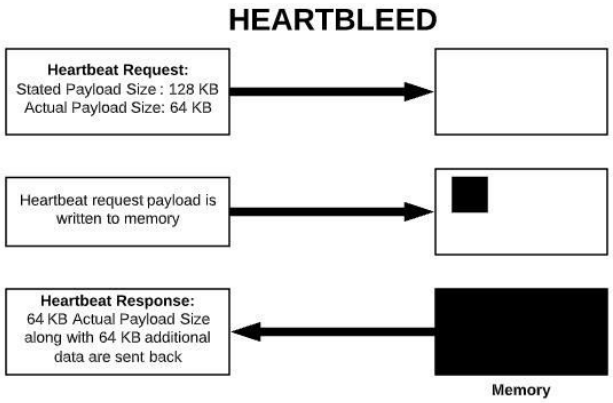

Figure 10: Heartbleed vulnerability

Countermeasure: It is recommended not to use an old cryptographic library since they have security flaws so it is strongly advised to keep every service up to date by installing security patches.

K. Sub-resource Integrity Protection

Sub-resource integrity is used to protect the integrity of scripts and style sheets in Web applications. Scripts are hosted on Content Delivery Networks (CDN), and the interesting question is what happens if the attacker injects a malicious script to CDN. It turns out that the malicious script can be used to attack a website.

Countermeasure: This can be done in a similar way of password hash concept discussed previously. Compute the hash of the script before the script is used in the Web application. When the script is fetched by the browser it computes the hash of the script. Then the browser compares the hashes of the original script and the script picked by it. If this hashes are not equal then the script's integrity is assume to be violated and the script will not be processed by the browser.

Volume: 13 Issue: 01 Pages: 4861-4873(2021) ISSN: 0975-0290

L. AngularJS injection

AngularJS [37, 38] is one of the most popular client side frameworks in modern web applications. AngularJS template injection is one of the most dangerous attacks on AngularJS applications as the attacker can proceed from AngularJS template injection to a cross site scripting attack. Checking for AngularJS application vulnerability can be done via JS template injection. For this, pen-tester needs to provide the input to the web application in double curly brackets. For example, {{10-2}}. In Angular JS double curly brackets are used to evaluate an expression. When run in the browser, if the response is exactly the same data (i.e., {{10-2}}) then AngularJS template injection is not possible. If the response is the number eight as the result of this expression then AngularJS template injection is possible. In this way, the script provided by the attacker in between double curly brackets will be executed or not can be checked. Some of the payloads which can be used to fetch the information from server if AngularJS injection attack is possible are as follows:

*{{7*7}}*
{{constructor.constructor('alert(1)')()}}
{{'a'.constructor.prototype.charAt=[].join;
$eval('x=""')+''}}

Countermeasure: Use a static template instead of a dynamic template. AngularJs template injection is related to dynamic templates. Hence, using a static template instead of a dynamic template can avoid AngularJS template injection.

M. HTTP Parameter Pollution

HTTP Parameter Pollution (HPP) is occurring of a parameter more than once in a URL. In such cases, which occurrence of a given parameter will be taken into account is turns out to be understood differently by different technologies. It can lead to serious vulnerability such as bypassing authorization. Since the URLs are polluted with more than required parameters this vulnerability is known as HPP. Consider the following original request in a URL

https://www.redacted.com/zephyr-mini-alpha-bo
ard/p/&pid=ETYDBYSKDDZQGDJD&vi=XXXX

To pollute it, we have changed the request to by adding the parameter u as follows

https://www.redacted.com/zephyr-mini-alpha
*board/p/?**u=http://www.evil.com**&pid=ETYDBYS*
KDDZQGDJD&vi=XXXX

If the HPP vulnerability exists in the web application then the polluted link will open the content of *evil.com*.

Countermeasure: Trigger an exception when there is more than a single occurrence of a given parameter in either GET or POST request. When an exception is triggered, the request for more than a single occurrence of a given parameter will not be processed by the web browser.

N. Click Jacking

It is an ambush that tricks a customer to click a malicious website page segment prepared by an attacker [39]. This can make customers unintentionally download malware, visit vindictive pages, give accreditations or fragile information, transfer money, or purchase things on the web. Usually, click jacking is performed by indicating a subtle page or HTML segment inside an iframe on the webpage that the customer visits. The subtle page could be a malevolent page, or a genuine page the customer did not hope to visit. For example, a page on the customer's monetary site that endorses the trading of money.

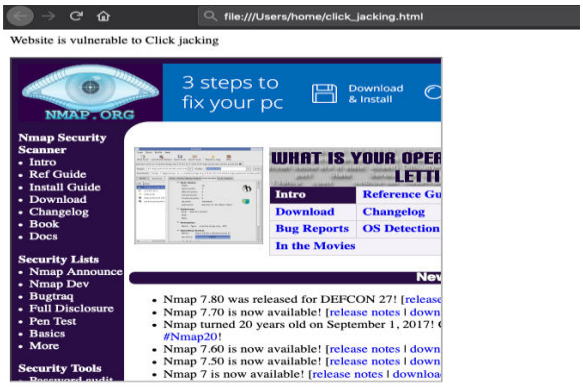

Figure 11: Click Jacking Vulnerability (nmap.org website embedded in iframe window)

There are two variants of the click jacking exists such as 1) *Like jacking*: It is a method where the "*Like*" button in social media or e-commerce website (ex. Facebook, Flipkart, etc.) is controlled, making customers "*Like*" a page or product that they truly did not plan to like. 2) *Cursor jacking:* It is a user interface in which, the investigating system changes the cursor from the position that the customer locates to another position. Cursor Jacking relies upon vulnerabilities in Flash program and the Firefox browser.

As shown Figure 11 the vulnerable website gets injected in iframe. We have written a HTML program and in an iframe, inserted the URL of the website to test if it is visible in iframe window or not. A website is said to safe from this vulnerability if it refuses to connect via iframe.

Countermeasure: The solution to click jacking vulnerability is to deny the insertion of X-FRAME-OPTION in the header of the webpage.

O. Insecure Direct Object Reference (IDOR)

This is one of the most dangerous vulnerabilities in Web applications and it can be easily detected and exploited by the attacker. Surprisingly, some of the famous companies and government organizations were found vulnerable to this attack. For example, Facebook was vulnerable to this attack in which the attacker could post an image as anyone

Volume: 13 Issue: 01 Pages: 4861-4873(2021) ISSN: 0975-0290

on Facebook. Fortunately for Facebook, this vulnerability was discovered and reported by an ethical hacker. Facebook was very grateful for this finding, and the ethical hacker was rewarded with $10,000. Starbucks was also vulnerable to IDOR. As a consequence, the attacker could transfer funds from one Starbucks card to another. The U. S. Department of Defense was found vulnerable to IDOR [40].

Consider the following example URL of attacker

https://socialmedia.com/messages?user_id=123

Attacker changes the value of the parameter to some other user because of the IDOR, the attacker is able to escalate the privileges to another user as shown below.

https://socialmedia.com/messages?user_id=121

The best countermeasure of this attack is to assign the random key value to the user id so that it cannot be guessed by attacker to perform IDOR. The modified link is as follows

https://socialmedia.com/messages?user_key=6MT9EalV9F7r9pns0mK1eDAEW

Countermeasure: Make sure that access control is implemented properly in e-commerce Web application. When access control is implemented properly, the attacker cannot refer directly to the object of another user, and then the attack can be stopped.

V. CONCLUSION AND FUTURE WORK

Apart from theoretical concepts, this paper focused on practical approach to find various vulnerabilities that commonly exist in e-commerce websites. In order to perform security testing of e-commerce websites, this paper has discussed various tools like nmap, burp suite, sublister, hydra etc. Then this paper discussed the Recon phase in which finding vulnerabilities such as sensitive data exposure, API key leakage, hidden directories findings, sub domain enumeration, and others. Importantly this paper discussed the usage of GHDB in Recon process. Finally, this paper discussed various vulnerabilities that a security tester should look in an e-commerce websites like authentication issues, HTTP parameter pollution, XSS, CSRF, IDOR and others.

As part of future work, we will be focusing on more critical vulnerabilities detection like remote code execution, blind XSS, time based and blind SQLi, web cache deception, XEE (XML External Entity) attacks, SSRF (Server Side Request Forgery) and others. Most importantly, we attempt to perform both static and dynamic analysis of Android applications vulnerability assessment. It is because along with the web application, every e-commerce website is also having its mobile application running on widely used Android platforms. Since manual security testing consumes a lot of time and effort, we are in the process of building a tool in python which will reduce efforts of security testers. Currently, we are in the process of automating the tasks which produce only true positive or true negative results.

REFERENCES

[1]. Berryman, K., Harrington, L., Layton-Rodin, D., & Rerolle, V. (1998). *Electronic commerce: Three emerging strategies*. The McKinsey Quarterly, (1), 152-160.

[2]. Kashif, M., Javed, M. K., & Pandey, D. (2020). A Surge in Cyber-Crime during COVID-19. *Indonesian Journal of Social and Environmental Issues*, *1*(2), 48-52.

[3]. Lallie, H. S., Shepherd, L. A., Nurse, J. R., Erola, A., Epiphaniou, G., Maple, C., & Bellekens, X. (2020). Cyber Security in the Age of COVID-19: A Timeline and Analysis of Cyber-Crime and Cyber-Attacks during the Pandemic. *arXiv preprint arXiv:2006.11929*.

[4]. Foregenix Survey*: https://www.foregenix.com/blog /over-75-of-global-magento-websites-at-high-risk-from-hackers-due-to-a-simple-security-oversight* (Last visited 09/07/2020).

[5]. Baptist, M. R. R., Raj, M. N., Banerjee, P., & Kumar (2020), B. An Empirical Study on Usability and Security of E-Commerce Websites. *International Journal of Computer science engineering Techniques*, 5(3), 1-10.

[6]. Seng, L. K., Ithnin, N., & Said, S. Z. M. (2018*).* The approaches to quantify web application security scanners quality: a review. *International Journal of Advanced Computer Research*, 8(38), 285-312.

[7]. Toch, E., Bettini, C., Shmueli, E., Radaelli, L., Lanzi, A., Riboni, D., & Lepri, B. (2018). The privacy implications of cyber security systems: A technological survey. *ACM Computing Surveys* (CSUR), 51(2), 1-27.

[8]. Thomas, T. W., Tabassum, M., Chu, B., & Lipford, H. (2018, April). Security during application development: An application security expert perspective. *In Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (pp. 1-12).

[9]. Humayun, M., Niazi, M., Jhanjhi, N. Z., Alshayeb, M., & Mahmood, S. (2020). Cyber Security Threats and Vulnerabilities: A Systematic Mapping Study. *Arabian Journal for Science and Engineering*, 1-19.

Volume: 13 Issue: 01 Pages: 4861-4873(2021) ISSN: 0975-0290

[10]. Toapanta, S. M. T., Caicedo, H. A. M., Sanchez, B. A. N., & Gallegos, L. E. M. (2020, March). Analysis of Security Mechanisms to Mitigate Hacker Attacks to Improve e-Commerce Management in Ecuador. *In 2020 3rd International Conference on Information and Computer Technologies (ICICT)* (pp. 242-250). IEEE.

[11]. Khera, Y., Kumar, D., & Garg, N. (2019, February). Analysis and Impact of Vulnerability Assessment and Penetration Testing. In *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)* (pp. 525-530). IEEE.

[12]. Rahman, M. A., Amjad, M., Ahmed, B., & Siddik, M. S. (2020, January). Analyzing Web Application Vulnerabilities: An Empirical Study on E-Commerce Sector in Bangladesh. In *Proceedings of the International Conference on Computing Advancements* (pp. 1-6).

[13]. Lis, A. (2019). *Comparison and analysis of web vulnerability scanners* (Bachelor's thesis).

[14]. Makino, Y., & Klyuev, V. (2015, September). Evaluation of web vulnerability scanners. In *2015 IEEE 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)* (Vol. 1, pp. 399-402). IEEE.

[15]. Amankwah, R., Chen, J., Kudjo, P. K., & Towey, D. An empirical comparison of commercial and open-source web vulnerability scanners. *Software: Practice and Experience*.

[16]. Pan, Y. (2019, August). Interactive Application Security Testing. In *2019 International Conference on Smart Grid and Electrical Automation (ICSGEA)* (pp. 558-561). IEEE.

[17]. Asaduzzaman, M. (2020). *Security Aspects of e-Payment System and Improper Access Control in Microtransactions* (No. 3717). EasyChair.

[18]. Goutam, A., & Tiwari, V. (2019, November). Vulnerability Assessment and Penetration Testing to Enhance the Security of Web Application. In *2019 4th International Conference on Information Systems and Computer Networks (ISCON)* (pp. 601-605). IEEE.

[19]. Felderer, M., Büchler, M., Johns, M., Brucker, A. D., Breu, R., & Pretschner, A. (2016). *Security testing: A survey. In Advances in Computers* (Vol. 101, pp. 1-51). Elsevier.

[20]. Mahajan, A. (2014). Burp Suite Essentials. Packt Publishing Ltd.

[21]. GitHub, I. (2016). GitHub. *URl: https://github. com/ (visited on 25/06/2021).*

[22]. Kali Linux. *URl: https://kali.org/ (last visited on 25/06/2020).*

[23]. Parrot Security. *URl: https://parrotlinux.org/ (last visited on 25/06/2021).*

[24]. Security Onion. *URl: https://securityonion.net/ (last visited on 25/07/2021).*

[25]. Sy, E., Mueller, T., Burkert, C., Federrath, H., & Fischer, M. (2020). Enhanced performance and privacy for TLS over TCP fast open. *Proceedings on Privacy Enhancing Technologies*, *2020*(2), 271-287.

[26]. Mendez, X. Wfuzz—The Web Fuzzer. Available online: *https://github.com/xmendez/wfuzz (last accessed on 25/06/2021).*

[27]. Exploit database. Available online*: https://www.exploit-db.com/ (last accessed on 05/07/2020).*

[28]. Common Vulnerabilities and Exposures. Available online*: https://cve.mitre.org/ (last accessed on 05/07/2020).*

[29]. Halfond, W. G., Viegas, J., & Orso, A. (2006, March). A classification of SQL-injection attacks and countermeasures. In *Proceedings of the IEEE international symposium on secure software engineering* (Vol. 1, pp. 13-15). IEEE.

[30]. Turner, S. (2014). Transport layer security. *IEEE Internet Computing*, *18*(6), 60-63.

[31]. Klein, A. (2008). Attacks on the RC4 stream cipher. *Designs, codes and cryptography*, *48*(3), 269-286.

[32]. Burrows, J. H. (1995). *Secure hash standard*. Department of Commerce Washington DC.

[33]. Viega, J., Messier, M., & Chandra, P. (2002). *Network security with openSSL: cryptography for secure communications*. " O'Reilly Media, Inc."

[34]. *Pentest monkey. http://pentestmonkey.net/ (last accessed on 05/07/2020).*

[35]. *Pentester Land. https://pentester.land/ (last accessed on 05/07/2020).*

[36]. Durumeric, Z., Li, F., Kasten, J., Amann, J., Beekman, J., Payer, M., ... & Halderman, J. A. (2014, November). The matter of heartbleed. In *Proceedings of the 2014 conference on internet measurement conference* (pp. 475-488).

Volume: 13 Issue: 01 Pages: 4861-4873(2021) ISSN: 0975-0290

[37]. Campesato, O. (2020). *Angular and Machine Learning Pocket Primer*. Stylus Publishing, LLC.

[38]. AngularJS. *Url. https://angularjs.org/ (last accessed on 05/07/2020).*

[39]. Calzavara, S., Roth, S., Rabitti, A., Backes, M., & Stock, B. (2020). A Tale of Two Headers: A Formal Analysis of Inconsistent Click-Jacking Protection on the Web. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*.

[40]. Sołtysik-Piorunkiewicz, A., & Krysiak, M. (2020). The Cyber Threats Analysis for Web Applications Security in Industry 4.0. In *Towards Industry 4.0—Current Challenges in Information Systems* (pp. 127-141). Springer, Cham.

[41]. Ramesh Kumar, "A Reliable Authentication Protocol For Peer To Peer Based Applications", *International Journalof Advanced Networking and Applications*, Vol. 12, Issue 05, Pages 4714-4718, 2021.

[42]. Harikrishna Bommala, S. Kiran, T.Venkateswarlu, M. Asha Aruna Sheela, "Fibonacci Technique For Privacy And Security To Sensitive Data On Cloud Environment", *International Journalof Advanced Networking and Applications* , Vol 11, Issue 04, Pages 4374-4377, 2020.

AUTHOR'S BIOGRAPHY

Agrah Jain received integrated B.Tech and M.Tech in Computer Science and Engineering from Jaypee Institute of Information technology, Noida, India in 2020. Currently he is working as a Solution Advisor, Delloitte USI, Gurugram. Before joining Delloitte, he worked as Security Analyst in the Quality Analysis Department, Wheelseye Technology Pvt Ltd, Gurgon, India. He is the corresponding author of this paper.

P. Raghu Vamsi is currently working as Assistant Professor (Senior Grade) in the Department of Computer Science and Engineering (CSE), Jaypee Institute of Information Technology (JIIT), NOIDA, UP, India from July 2016. He received B.E in CSE from University of Madras, Chennai, India in 2003, M.Tech (Software Engineering) from Kakatiya University, Warangal, India in 2007, M.B.A in Human Resource Management from Indira Gandhi National Open University, New Delhi, India, in 2010, and M.A in Astrology from Potti Sree Ramulu Telugu University, Hyderabad, India in 2014, and PhD in CSE from JIIT, NOIDA, India in 2016. He secured Kakatiya Univeristy First rank in M.Tech (Software Engineering). He has more than 12 years of teaching experience in various engineering institutions and 4 year of research experience. His research interests include security in wireless ad hoc networks, security in wireless sensor networks, communication protocols, distributed computing, computer networks security, trust-aware security methods, and localization in wireless sensor networks, Internet of Things, and next generation wireless communications. He is a life member of Indian Society for Technical Education (ISTE) India.