Web Application Penetration Testing

Nagendran K, Adithyan A, Chethana R, Camillus P, Bala Sri Varshini K B

Abstract: This paper describes the in-depth technical approach to perform manual penetration test in web applications for testing the integrity and security of the application and also serves as a guide to test OWASP top 10 security vulnerabilities. The paper is more focused on providing detailed knowledge about manual web application penetration testing methodologies in order to secure them from malicious black hat hackers.

Keywords: Web pentesting, Website Hacking, OWASP testing guide, web vulnerability scanning, bug hunting

I. INTRODUCTION

Information is Wealth. Each and every bit of information has a cost in this digital world. All that information is stored in the form of Data in Internet. There are two types of data, Public and Private. The public data are resources that are available publicly in the Internet. Ex: data that results from a Google search query. The private data are the resources that are bagged behind a wall of authentication. Ex: Your email data. Emails are protected by wall of authentication which requires your user name and password to authenticate successfully. But what if someone can read your emails without authentication? Or what if someone can read your emails by acquiring your credentials from you without your knowledge? There comes the need for Web Application Security. Everything is web based now. Most of the Softwares has their own web app version too. But all the Web Applications are prone to Hacking. This is why, Web Application Penetration emerge as need of the hour. Website need a defence in depth approach to mitigate against the security flaws[1]. It is essential to Penetration test every web application before it goes online and gets hacked by a Black Hat cyber warrior out there. Hackers constantly hunt for web app vulnerabilities[5]. The best way to mitigate against the hacker attacks is to learn their methodologies[2]. Here, we discuss about the most mandatory penetration tests that has to be done before the application goes Online and Techniques explaining how to perform those tests.

II. CLASSIFICATION OF WEB ATTACKS

A. Client Side Attacks

As the name refers, the client side attacks are deployed by the attackers against the clients of a particular website to steal their data.

Manuscript published on 30 August 2019.
***Correspondence Author(s)**

1Nagendran K, Assistant professor, Department of IT, Sri Krishna College of Engineering and Technology.
Adithyan A, lectures at various workshops and Universities including Defcon Trivandrum and IITM Research Park.
Chethana ravichandran, pursuing third year in the branch of information technology.
Camillus P, pursuing B.tech information technology at Sri Krishna College of engineering and technology.
Bala Sri Varshini K B, pursuing Third year B.Tech. Information Technology at Sri Krishna College Of Engineering and Technology.

The most common attacks on client side include Cross Site Request Forgery (CSRF), Cross Origin Resource Sharing (CORS), Cross Site Scripting (XSS), Clickjacking, HTML Injection, etc.

B. Server Side Attacks

On the contrary, Server side attacks are deployed against the web server. In server side attacks, the attacker targets a vulnerable end-point of the web app and sends a malicious payload to the server. After the successful execution of payload in the server, it responds to the attacker with the confidential data he requested with the payload. The confidential data include Server information, Service running on the server and their version related info, user information, passwords etc.

III. PENETRATION TESTING

Penetration testing and vulnerability assessment are two different terms. The latter includes uncovering the security flaws and reporting it to the concerned security team whereas the former includes exploiting the discovered flaw and attempting data ex-filtration or privilege escalation or any other possible malignant action on the target host. Penetration testing helps the developers to find security flaws in their application and maintain their application secure. Performing realtime tests on web applications has proven to be helpful in hardening the security of the website[3]. Regular penetration testing is mandatory after making the application online to avoid potential risks. Because, new zero-day vulnerabilities are discovered day to day and its developer’s primary responsibility to have a keen eye on what kind of third party services they are relying on. Penetration testing is not only limited to web apps, but also performed on IoT Devices, Networks, Computer Systems, Mobile Applications etc. But in this paper, we will be discussing about the techniques used for testing web applications.

IV. MANUAL TESTING VS AUTOMATED TOOLS

Manual penetration testing needs lot of expertise in playing with HTTP requests and response. An Expert penetration tester would know the possible attacks that can be performed on a particular end point by fuzzing the HTTP requests. The main drawback of using automated tools are false positives. The automated tools work based on the use cases coded by the developer. And every developer has their own testing technique. Some of them may be effective while some may not. So not all the automated tools result in success. It’s better to follow own strategy when it comes for penetration testing. But automated tools play a vital role in content discovery and reconnaissance and it helps in saving lot if time. Within few years, the whole Pentesting process would be automated with integrated Penetration testing can be broadly classified into five phases[4].

1. Reconnaissance
2. Scanning
3. Exploitation
4. Maintaining Access and Privilege Escalation
5. Clearing Tracks and Reporting

V. RECONNAISSANCE

Reconnaissance is the foremost phase in Penetration Testing where the attacker gains necessary information about the target. This helps the attacker to gain foothold on what are the technologies the application is using which further helps him to identify the security vulnerabilities. For example, in order to hack something, the attacker doesn't always hack to find the direct way to break in. He can even compromise the hosts on which the target is relying on and then pivot into the target. There are lot of websites as well as frameworks available for performing reconnaissance.

A. Online Apps for Recon

- pentest-tools.com – provide detailed information about the web server, frameworks, hosting panels, font scripts, JavaScript frameworks used in the application along with their version info.
- dnsdumpster.com – provides information about DNS servers, MX records, TXT records, Host records and domain map
- virustotal.com – checks for malicious files in the website and supplies the DNS information and sub-domain info.
- Hackertarget.com – offers basic functions like reverse DNS lookup, TCP UDP port scan, reverse IP lookup, and finding shared DNS servers.
- Shodan.io – helps the attackers to find internal infrastructure of an organization which are exposed to the internet. Also, Shodan makes the job easier by making a port scan on the target IP address.
- Censys.io – similar to shodan but censys helps in asset discovery buy analyzing the SSL certificate
- Wayback.com – great place to find sensitive information. Sometime, when you get a 403 forbidden error, there are possibilities that back then, the page may have been left accessible publicly. Similarly, when you find a 404 page not found error, there are possibilities that back then, there may be sensitive information left publicly on that page.
- Github.com – helps the attacker to find API keys and other sensitive infos of an Organization and employee email IDs

B. Recon Frameworks

- Recon-ng – a swiss army knife for web application reconnaissance. It has various modules to perform Open Source Intelligence (OSINT) gathering as well as reconnaissance. You've to configure the tool with API of respective source that the tool uses.
- TIDoS framework – consists of 48 inbuilt modules for OSINT and recon process. In addition to this, it also contains modules for scanning and enumeration, vulnerability analysis, Exploitation and other auxiliary modules.
- Wappalyzer – a firefox plugin which helps attackers to know about the technologies used by the target server along with their version info. Handy for finding the CMS (incase of any) used by the target domain and their plugins.
- Recon Dog – a python based recon framework which offers limited yet effective functions like DNS lookup, Honeypot detection, Censys lookup, and filtering technologies used by the target web server.
- DNSRecon – scans all type of domain records

VI. SCANNING

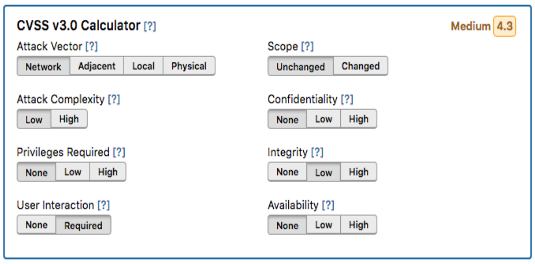
Scanning is the second phase of penetration testing which uses the information gathered from recon and digg deep into the services and contents. It involves host discovery, content discovery, scanning ports and services, vulnerabilities, OS fingerprinting etc. The data gathered from the Scanning phase would give the attacker, enough knowledge select the right end point to begin carrying out his exploitation phase. Similar to recon, both web apps and frameworks are available to do this yet, Frameworks do the best job in case of scanning. Some best frameworks for scanning include

- Nmap - most prominent and traditional tool for port scanning and OS fingerprinting. Nmap has its own scripting engine called Nmap Scripting Engine (NSE) which allows users to write their own script and automated their tasks. NSE scripts such as nmap-vulners,vulscan retrieves the CVE IDs associated with the target port, operating system etc. Nmap supports different types of scan to detect and evade various types of IDS and Firewall. NSE scripts like smtp-strangeport, dns-blacklist, http-enum are most commonly used for website scanning purposes.
- Nikto - Open Source Web Vulnerability scanner which scans for server misconfigurations and insecure files. It supports features like URI Encoding, premature URL ending, fake parameters. Also, it scans for common web vulnerabilities like Clickjacking and looks out for interesting files on the web server.
- W3af - Automated open source scanner which tests for 200+ vulnerabilities written in python.
- Acunetix - Most common vulnerability scanner which tests for more than 4500 vulnerabilities. The best part of acunetix is that it doesn't require any expertise to handle the tools. It's a GUI tool which comes handy with all the features and all you have to do is just to enter the URL of your target website. No automated scanner can assure you 0% false positive rate and acunetix no different.

There are lot of enterprise tools and other open source scanning tools used for scanning web vulnerabilities. Some tools out there are specifically used to scan a particular vulnerability, like Knoxss which hunts only for Cross-Site-Scripting attacks.

VII. EXPLOITATION

Although automated tools do a decent work over recon and scanning, they can't compete with manual exploitation techniques. A vulnerability can be exploited in thousand ways but manual testing finds the most appropriate way (i.e. the way which has the highest impact & severity). The severity of the vulnerabilities is represented in different terms depending upon the organisation. OWASP summarizes the vulnerabilities from A0 to A10 according to their severity. Similarly, Online Crowdsourcing platform like BugCrowd has their own vulnerability rating taxonomy where each vulnerability is assigned a value from P1 - P5 depending upon the Vulnerability severity and impact. HackerOne makes use of Industrial standardized severity calculator like Common Vulnerability Scoring System (CVSS).


CVSS v3.0 Calculator [?] Medium 4.3

Attack Vector [?]: [✓ Network] Adjacent Local Physical
Scope [?]: [✓ Unchanged] Changed
Attack Complexity [?]: [✓ Low] High
Confidentiality [?]: [✓ None] Low High
Privileges Required [?]: [✓ None] Low High
Integrity [?]: None [✓ Low] High
User Interaction [?]: None [✓ Required]
Availability [?]: [✓ None] Low High

Let's begin from most common vulnerabilities that are exploited widely all around the globe.

A. SQL Injection

SQL database are used by almost 70% of the web applications that has the need of storing data. Injecting malicious queries via data input fields and accessing the unauthorized data is referred to as SQL injection. SQL database has roles and permissions for each user. But if an attacker executes a malicious query and the server performs the attacker requested action due to improper query validation[6], the attacker may leverage this to attain administrative rights which would then let him to have complete access over the database[7]. He can even shut down the database after attaining the administrative privileges. The attacker vector deviates with the type of the SQL database the target application is using. The below testing techniques are used for testing MySQL database. The sql injections are classified into the following types:

- Union based – the UNION statement in SQL allows to combine two queries which has the same structure. By leveraging this, the attacker can insert the UNION statement in an input field of the application and execute his own query. For example, consider an e-commerce web application that lists Products based on category. The URL would look something like www.redacted.com/products.php?category=10
  The SQL query would look like

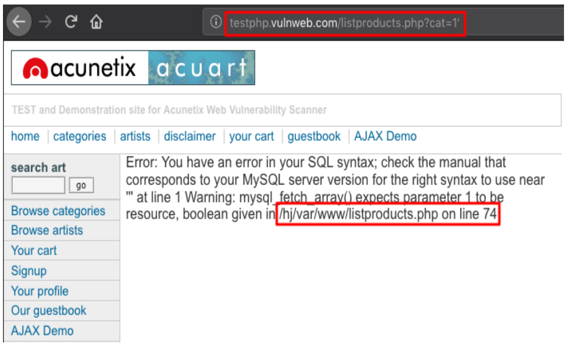
```
select item_name, item_info, item_cost from products where category=1
```

  After injection malicious union statements in the URL, the query sent by the application to the database would look like

```
select item_name, item_info, item_cost from products where category=1 union select 1 from information_schema.tables
```

  Now this query after execution displays the contents from the table information_schema

- Error based – Error based SQL injection allows the attackers to exfiltrate data from the error messages thrown by the application. Errors are triggered when the attacker sends an input that affects the SQL query. E.g.: Semicolon, single quote, Double quote and comment delimiters.


testphp.vulnweb.com/listproducts.php?cat=1'

Error: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ''' at line 1 Warning: mysql_fetch_array() expects parameter 1 to be resource, boolean given in /hj/var/www/listproducts.php on line 74

acunetix acuart
TEST and Demonstration site for Acunetix Web Vulnerability Scanner
home | categories | artists | disclaimer | your cart | guestbook | AJAX Demo
search art
go
Browse categories
Browse artists
Your cart
Signup
Your profile
Our guestbook
AJAX Demo

- Blind Injection – can be divided into two types 1) Time based 2) Boolean based. Both of these techniques exploit the application by asking Yes/No question. For example, consider a query

```
select exp_date when cvv='cvv' then pg_sleep(10) else NULL end from Credit_Card where user_id=201
```
[10]

The above query on execution returns quick if the condition is false and takes 10 seconds if the condition is right. Using this attack, the attacker can exfiltrate binary answers from the response.

SQL injection can also be used to bypass the Login forms if user input is not sanitized properly. For example, let's consider that a web application has a login page which requires username and password for authenticating the user. The username and password are requested from the user as inputs and the application generates genuine SQL queries to the backend DBMS. Now, the attacker injects a fragment of a SQL instruction in the username or password field and passes it to the application. The application then combines the malicious SQL statement given as input by the attacker with a Valid SQL statement and queries the Database.

Retrieval Number: J91730881019/19©BEIESP
DOI: 10.35940/ijitee.J9173.0881019
Journal Website: www.ijitee.org

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

The database then responds to the input query. For example, consider a PHP query which the application requests to the database after receiving the input from the user.

```
$query = “select CreditCard, CVV, Exp from Users where username=‘”.$_POST[“username”].”‘ and password=‘” $_POST[“password”].”’”
```

Assume that the Application takes and places the input directly in the query without any validation. If user provides the credentials Username = REDACTED & Password = PASSWORD, the PHP query would look like this

```
$query = “select CreditCard, CVV, Exp from Users where username='REDACTED' and password='PASSWORD'
```

Now, the attacker after understanding the flow of the querying process of the application would start testing his payloads one by one. For example, let’s assume that the attacker fills the input with Username = **Jhon'; --** and Password = WhoCares. The Query would now look like

```
$query = “select CreditCard, CVV, Exp from Users where username='Jhon'; -- and password='WhoCares'
```

When the above query is executed, the attacker will be presented with the Credit card information of the user **Jhon**. Let’s analyse the input of the attacker. The **Jhon** part of the input completes the username part because the colon gets closed after Jhon. The semicolon ‘**;**’ ends the query and **--** comments all other lines. So, only username goes to the application and the password is commented out. Due to improper validation by the application, the Database responds with the requested information. This is how SQL Bypass techniques work. The most common payloads to bypass such type of insecure login forms are

- ' OR 1=1 -- 1
- ' OR '1'='1
- ' or 1=1 LIMIT 1;--
- admin';--
- " or 1--

It’s not obvious that this payload will give 100% result and it depends completely upon how the application handles the input and forms the query. We have to build our own payload by understanding the querying process of the application.

B. Cross Site Scripting

Cross site scripting (XSS) allows the attacker to inject a malicious script (often javascript) in the target website. They allow the attacker to execute undesired function’s in other user’s browser who visits the injected website. XSS attacks can be leveraged to various high severe impacts such as account takeover, credential stealing, data exfiltration, cryptomining, keylogging, fingerprinting, tab-napping, screenshot capture and so on. XSS can be combined with several other vulnerabilities to increase the impact level. During 2005, Samy Kamkar’s Samy worm exploited a Cross site scripting vulnerability in MySpace and affected more than one million users. XSS attacks can be broadly classified into 4 types

1. Reflected XSS – reflects the attacker’s input in the target web application. The attack can be abused to deceive the user and exploit his credentials or cookies. The attacker tests for all the input fields in a web application. He learns how the application handles the input. After checking whether his input is sanitized properly or not, the attacker tries to break the HTML tags and execute his input if there is improper sanitization. In most of the cases, the attacker uses the <script> tag of HTML to break the application’s source code and bring his output out of it which is then executed by the browser. These reflected xss are often tested by popping up an alert box.However, this attack works only when the victim visits the attacker’s malformed link. For example, take a Library application which has a search function where a user can input his Search keyword such as Book name, Author name etc. The attacker first inserts a search query and tests for reflections form the site. If his input is reflected somewhere in the response, the attacker tries to escape his input outside the tag which is reflecting the input and eventually gets his payload executed by the browser.
2. Stored XSS – It’s the most powerful and persistent attack. Here, the attacker’s payload gets stored in the server and served to all the users who visit the application. This is how MySpace worm affected a million+ users. This attack takes place when the user supplied input is stored and displayed without any HTML encoding. Since the payload is served from the server, the Browser’s XSS filter wouldn't find it malicious.
3. DOM based XSS – It’s a subclass of reflected XSS where the attacker’s payload instead of touching the web server abuses the javascript in the client side. The DOM based attacks aren’t visible in the response page’s source code.
4. mXSS – refers to Mutated XSS which works by abusing the incorrect reading of innerHTML by the application. For example, <listing>&lt;img src=1 onerror=alert(1)&gt;</listing>[9], here the normal script is mutated into an image element which then executes the alert function.

Browsers like Chrome has inbuilt XSS auditor which prevents the application form executing malicious codes. Most of the web application now implements Web Application Firewall(WAF) which prevents attackers from attaining XSS. Still, every day, researchers find new bypass techniques to bypass the WAF and execute their payloads. XSS may exist on almost all input insertion points. For example, the file upload functionality of an application can be leverage to attain XSS by renaming the file name with a XSS payload. There are few automated tools which can help to find the right XSS payload vectors. Few of them are

- Knoxss
- XSS Sniper
- XSS Strike
- XSS Hunter

C. Cross Site Request Forgery

Although CSRF can have severe impact on client side, they require victim's interaction. Generally, Cross Site Request Forgery attack, as the name refers to, marks the victim to make a forged request. This can be done by sending a link to the victim and obtaining a click. You must have noticed while browsing that, if you are logged into a website in one tab and open the website in other tab, you need not have to login again because the website shares the cookies with tabs if the domain is same. Also, the website can share access with other domain, if the application's Origin header is defined with the site where it has to share the account access. This is how websites communicate within themselves. For each click that the user performs in a website, a HTTP request is sent to the web server. CSRF is possible on both GET and POST requests. GET CSRF can be exploited so easily just by redirecting the user to vulnerable URL. For exploiting CSRF using POST, the attacker has to craft a HTML page forged request when the victim clicks on something. For example, a password reset request would look like this.

```
POST /password_reset HTTP/1.1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X x.y;
rv:10.0) Gecko/20100101 Firefox/10.0
Host: www.redacted.com
Content-Type: application/x-www-form-urlencoded
Content-Length: length
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive
password=123456&confirm_password=123456
```

In the above request, there is no header present that defines where the request is originated from which means that any website can make this request by placing a html page like the one below

```
<html>

<body>
<form action="https://www.redacted.com/password_reset"
method="POST">
<input type="hidden" name="password" value="attacker" />
<input type="hidden" name="confirm_password"
value="attacker" />
<input type="submit" value="Submit request" />
</form>
</body>
</html>
```

Now, the attacker hosts this page in the internet and sends the link of this page to the victim. When the user clicks the "Submit", his redacted.com's account password is changed to "attacker". This is how CSRF attacks work. However, these types of attacks can be mitigated by placing CSRF token which are generated for each request from the client side and validated form the server side. Again, the CSRF token can be abused if they are improperly validated on the server side.

D. Cross Origin Resource Sharing

This attack abuses the earlier mentioned Origin header which is used for sharing user access between tabs. Same Origin Policy (SOP) prevented domains from passing sensitive information between themselves. But when days passed, there came the need for passing sensitive information within domains. For example, within a domain and its sub domain. To achieve this, Cross Origin Resource Sharing mechanism was invented which allowed domains to exchange information. Consider a GET request below that fetches a User's secret from a website.

```
GET /secret HTTP/1.1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X x.y;
rv:10.0) Gecko/20100101 Firefox/10.0
Host: www.redacted.com
Origin: attacker.com
Content-Type: application/x-www-form-urlencoded
Content-Length: length
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive
```

And if the response has any of the following header, there may exist a CORS misconfiguration.

1. Access-Control-Allow-Origin : *
2. Access-Control-Allow-Origin : attacker.com
3. Access-Control-Allow-Origin: null

Sometimes, the server may only validate a part of the domain in the origin header. For example, the Server may allow access to any domain which has the part of Original domain in it.

VIII. PRIVILEGE ESCALATION

E. Insecure Direct Object Reference

As the name suggests, the vulnerability arises when a application directly indicates any object without any encoding or authentication. These attacks can result in a severe data breach and unauthorised access to other user's data. The following is a HTTP request that downloads the user's account information.

```
GET /Download.php?id=701 HTTP/1.1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X x.y;
rv:10.0) Gecko/20100101 Firefox/10.0
Host: www.redacted.com
Origin: attacker.com
Content-Type: application/x-www-form-urlencoded
Content-Length: length
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive
```

As seen in the above request, the id=701 is directly pointing the object which the user has requested to download and there is no authentication header is passed in the request. Therefore, by manipulating the id parameter, the attacker can download other user's information. For example, the attacker may change the id=701 to id=805 and forward the request which in turn download the account information associated with the user id 805.

The IDOR vulnerability allows the user to escalate even their user privileges. For example, Abusing IDOR in a social networking platform may allow the attacker to delete, edit other user’s comments, posts and captions.

VI. REMOTE CODE EXECUTION

The vulnerability allows the attacker to execute arbitrary codes in the application with the normal user privileges. After the successful execution of his code, the attacker would attempt to escalate his privilege from normal user to administrator. The vulnerability occurs when a parser executes the user’s input before sanitization. For example, Consider the below HTTP request.

```
GET /language=eng HTTP/1.1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X x.y;
rv:10.0) Gecko/20100101 Firefox/10.0
Host: www.redacted.com
Origin: attacker.com
Content-Type: application/x-www-form-urlencoded
Content-Length: length
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive
```

The above request will be parsed as $lan=’eng’; by PHP. Now the attacker tampers the language parameter and inserts the code **eng’;phpinfo();**

```
GET /language=eng’;phpinfo(); HTTP/1.1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X x.y;
rv:10.0) Gecko/20100101 Firefox/10.0
Host: www.redacted.com
Origin: attacker.com
Content-Type: application/x-www-form-urlencoded
Content-Length: length
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive
```

The ‘**;**’ ends the first query and the **phpinfo();** query is concatenated with the first query. Forwarding the above request will result in exposure of the PHP information of the web server. This is how remote code execution vulnerability occurs. Remote code execution flaw in Linux servers allows the attackers to get reverse shell from the server to their computer. After getting the reverse shell, they will look for privilege escalation exploits released for the target’s Kernel version and obtain superuser rights to execute administrator level commands. To prevent applications from escalating privileges, it’s best advised to use least privileged services, process, and user accounts[7]

There are several other vulnerabilities which may cause potential damage to both server and client side like Command Injection, LDAP injection, Information Disclosure, CRLF injection, Host header injection, Open redirection, Insecure access control, Improper captcha validation. 2FA bypass and so on.

IX. CLEARING TRACKS AND REPORTING

All the incoming requests to a web server will be saved in a log file. If the attacker attains superuser permissions in the web server, he can delete the log file leaving no trace for him. But attaining superuser permission is not that easy and it depends upon what kind of kernel version and other vulnerable softwares, the server is using. So, rather than clearing logs, it’s better to use proxy mechanisms to penetration test a website. Reporting is the final process in the penetration testing. Write a detailed report which includes

- Name of the vulnerability
- Vulnerable Endpoint
- Technical Description of the vulnerability
- Business Impact and severity
- Proof of Concept (PoC) video or image

These are the minimum requirements to write a complete report.

REFERENCE

1. M. Howard And D.E. Leblanc, Writing Secure Code, Micro- Soft Press, 2002.
2. M. Khari, Sonam, Vaishali And M. Kumar, "Comprehensive Study Of Web Application Attacks And Classification," 2016 3rd International Conference On Computing For Sustainable Global Development (Indiacom), New Delhi, 2016, Pp. 2159-2164.
3. Jose Fonseca, Marco Vieira, And Henrique Madeira, "Evaluation Of Web Security Mechanisms Using Vulnerability & Attack Injection", Dependable And Secure Computing, Ieee Transactions (Volume:11, Issue: 5)
4. Https://Simplysecure.Blog/2017/07/05/Five-Phases-Of-Penetration-Testing/
5. K. Nirmal, B. Janet And R. Kumar, "Web Application Vulnerabilities - The Hacker's Treasure," 2018 International Conference On Inventive Research In Computing Applications (Icirca), Coimbatore, India, 2018, Pp. 58-62.
6. Padmaja K,"A Study On Web Application And Protection Against Vulnerability", In International Journal Of Engineering Research And Application, (Ijera),2012, Pp.001-006.
7. "Security Code Review-Identifying Web Vulnerabilities", By Kiran Maraju.
8. M.Khari And N.Kumar, "User Authentication Method Against Sql Injection Attack", International Journal Of Scientific And Engineering Research,2013, Pp. 1649-1653.
9. Http://Www.Thespanner.Co.Uk/2014/05/06/Mxss/
10. Https://Hackernoon.Com/Timing-Based-Blind-Sql-Attacks-Bd276dc618dd

AUTHORS PROFILE

Nagendran K has academic experience of 13 years. He is working as Assistant professor in IT Department at Sri Krishna College of Engineering and Technology. His educational qualification is M.E.,(Ph.D). He has published 8 research papers in International journals. His area of Interest include Computer networks, Network security, Data mining, Big Data analytics.

Adithyan A is an avid Security researcher with expertise in Cyber Security. He owns 4 CVEs and has published more than 5 research papers in the field of Cyber security. He had delivered his guest lectures at various workshops and Universities including Defcon Trivandrum and IITM Research Park. He has been listed in Microsoft and Apple security advisories for reporting critical vulnerabilities in their products. He is Oppo’s Security Leaderboard in Top 5th position for finding vulnerabilities.

Chethana ravichandran currently pursuing third year in the branch of information technology. Being a novice she still proves to be insightful and diligent. Her collaborative skills has always been her strength.she is still continuing to explore untravelled world of programming languages which includes c and python.Area of interests cover artificial intelligence and animation. She is intuitive and has quality communication skills.

Bala Sri Varshini K B pursuing Third year B.Tech. Information Technology at Sri Krishna College Of Engineering and Technology.She has interest in Machine learning and coding. Also has interest in python and programming in Java. She has presented a paper in International Conference and selected as Best Paper presentation award. Made achievements in Competitive programming and technical levels.

He is Camillus P pursuing B.tech information technology at Sri Krishna College of engineering and technology. His field of interest is networking and machine learning. Expertise in python and c language. Worked in many freelancing projects based on machine learning with python. Made achievements in competitive programming and hackathons. Good in developing websites with html back-end and have a good knowledge on PHP .