



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

WEB APPLICATION THREAT SCANNER(WATS)

Project Supervisor: Dr.AROUL CANESSANE, M.E., Ph.D.,

Name of the Student: JUANITA MELOSHA.K

Register Number: 39110417

Presentation Outline

- Introduction
- Objectives
- System Architecture / Ideation Map
- Module Implementation
- Application Snapshots
- Results and Discussions
- Conclusion & Future work
- References

Introduction

PROBLEM STATEMENT

Security professionals usually checks for security vulnerabilities manually using request ,response shared by the web application and the server. This process consumes lots of time and efforts.

PROPOSED SOLUTION

To build a automated tool to detect web based security vulnerabilities which will save time and effort for security professionals/companies/organizations

Introduction

- Web Application Threat Scanner(WATS) is used to find the websites bug and after that it shows the types of bug on that websites.
- This project is developed in PYTHON.
- It is a quick check for potential security vulnerabilities, using automated approach.
- It saves time ,money because if the vulnerabilities is exploited by the malicious attackers, it affects the reputation of the company/organizations
- This scanner widely cover end-users such as security learners in cyber security and professionals.

Objectives

- It is a quick check for potential security vulnerabilities, using automated approach.
- It saves time ,money because if the vulnerabilities is exploited by the malicious attackers, it affects the reputation of the company/organizations
- This scanner widely cover end-users such as security learners in cyber security and also professionals.

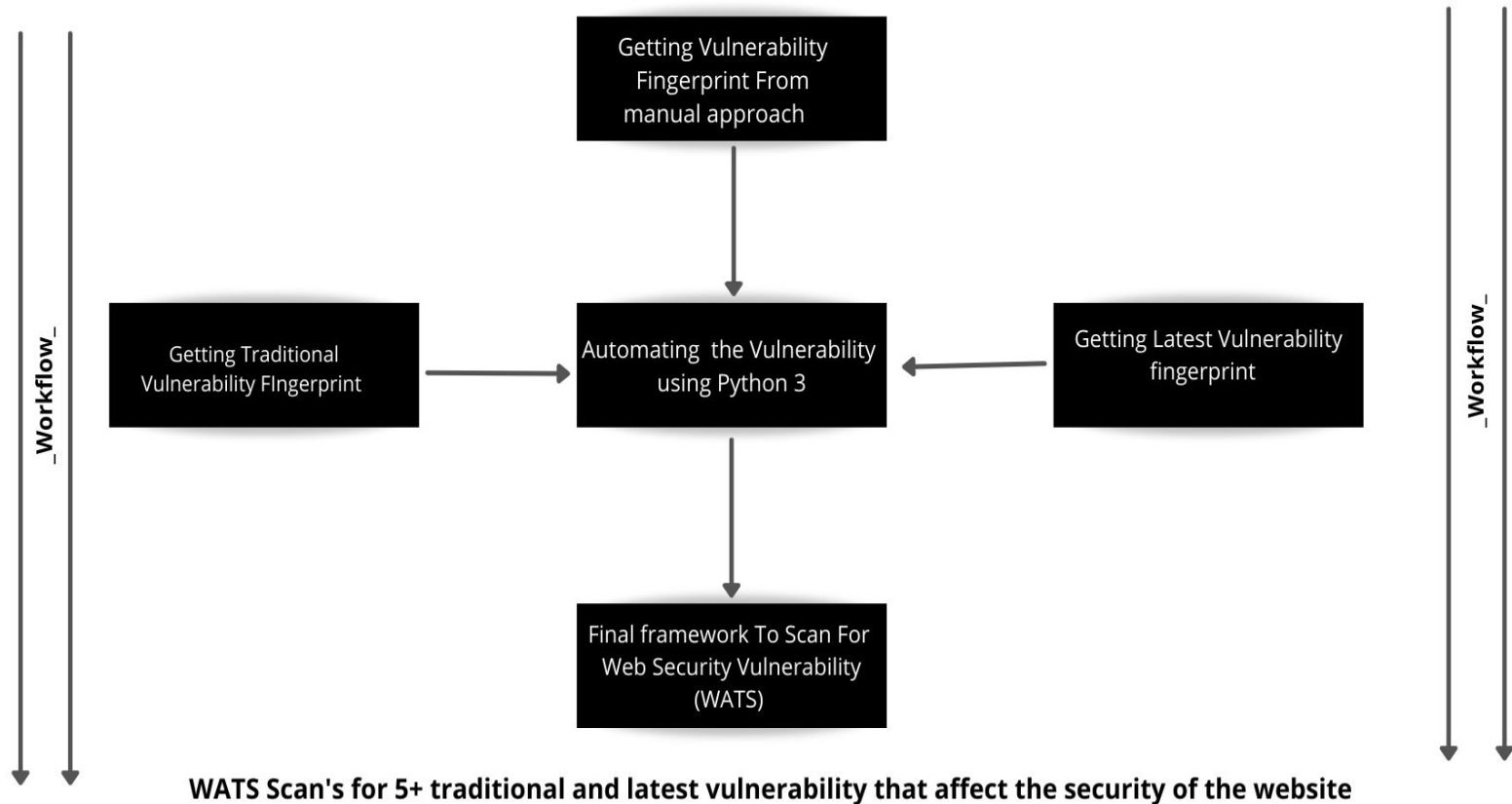
Scope

- Most of the online scanners, check for 1 or 2 vulnerabilities alone in large extend but WATS can scan more than 5 vulnerabilities in one go which makes this one stop solution and it will be open source , which can be further developed by the information security community.

Describe How?

To build an automated tool to detect web base security vulnerabilities , which will save time and effort for security professionals/companies/organizations , this can be developed into full fledged information security scanning framework

System Architecture / Ideation Map



Project Implementation

- First design your project either by Simulation or Calculation.
- Define Modules.
- Hardware and Software Requirements.
- Construction or Fabrication.
- Measurement and Analysis.
- Interpretation of Results.

Module Implementation

1. python-nmap
2. requests
3. urllib3
4. paramiko
5. wget
6. cryptography==2.4.2

Python-nmap Module

Python-nmap is a python library which helps in using nmap port scanner. It allows to easily manipulate nmap scan results and will be a perfect tool for systems administrators who want to automatize scanning task and reports. It also supports nmap script outputs. It can even be used asynchronously. Results are returned one host at a time to a callback function defined by the user.

Requests Module

Requests allows you to send HTTP/1.1 requests extremely easily. There's no need to manually add query strings to your URLs, or to form-encode your PUT & POST data — but nowadays, just use the `json` method.

Requests is one of the most downloaded Python packages today, pulling in around 30M downloads / week— according to GitHub, Requests is currently depended upon by 1,000,000+ repositories. You may certainly put your

urllib3 Module

urllib3 is a powerful, user-friendly HTTP client for Python. Much of the Python ecosystem already uses urllib3 and you should too. urllib3 brings many critical features that are missing from the Python standard libraries:

1. Thread safety.
2. Connection pooling.
3. Client-side SSL/TLS verification.
4. File uploads with multipart encoding.
5. Helpers for retrying requests and dealing with HTTP redirects.
6. Support for gzip, deflate, and brotli encoding.
7. Proxy support for HTTP and SOCKS.
8. 100% test coverage.

Paramiko Module

- Paramiko is a pure-Python [1] (2.7, 3.4+) implementation of the SSHv2 protocol [2], providing both client and server functionality. It provides the foundation for the high-level SSH library Fabric, which is what we recommend you use for common client use-cases such as running remote shell commands or transferring files.

wget Module

- Wget is a networking command-line tool that lets you download files and interact with REST APIs. It supports the HTTP , HTTPS , FTP , and FTPS internet protocols. Wget can deal with unstable and slow network connections. In the event of a download failure, Wget keeps trying until the entire file has been retrieved

Cryptography==2.4.2 Module

cryptography is a package which provides cryptographic recipes and primitives to Python developers. Our goal is for it to be your “cryptographic standard library”. It supports Python 3.6+ and PyPy3 7.2+.

cryptography includes both high level recipes and low level interfaces to common cryptographic algorithms such as symmetric ciphers, message digests, and key derivation functions. For example, to encrypt something with cryptography’s high level symmetric encryption recipe:

System Requirements

- 1.Windows / Linux / macOS Based Operating System
- 2.Python2/Python3 is installed
- 3.Required python Libraries which is given in the requirements.txt
- 4.Good computer/Laptop with good internet connection.

Methodology

- For this project, the literature review focused on the computer security, web application vulnerabilities, vulnerabilities scanning.
- Due to the vulnerabilities, detection model which can recognize the type of vulnerabilities is needed.
- In this project, only common vulnerabilities have been focused which are SQL Injection, ClickJacking, Nmap Port Scan, Host Header Injection, CORS, BruteForce, CMS Detection etc.

Results and Discussion

- Firstly, target host will be asked followed by the target port , then it will check whether the target is reachable or not.
- Once the Target has been alive then it will start checking following vulnerabilities check.

Results and Discussion

```
WATS — Python wats.py — 80x24
Last login: Tue Apr 12 11:47:48 on ttys000
[juanitamelosha@Juanitas-MacBook-Air ~ % cd desktop
[juanitamelosha@Juanitas-MacBook-Air desktop % cd WATS-main
[juanitamelosha@Juanitas-MacBook-Air WATS-main % cd WATS
[juanitamelosha@Juanitas-MacBook-Air WATS % python3 wats.py

dP  dP  dP  .d888888  d888888P .d88888b
88  88  88  d8'   88   88   88.   "'
88  .8P  .8P  88aaaaa88a  88   `Y88888b.
88  d8'  d8'  88   88   88   88   `8b
88.d8P8.d8P  88   88   88   d8'   .8P
8888' Y88'   88   88   88   Y88888P

Enter the Target Host : sathyabama.cognibot.in
Enter the Target port : 80

Starting WAVES

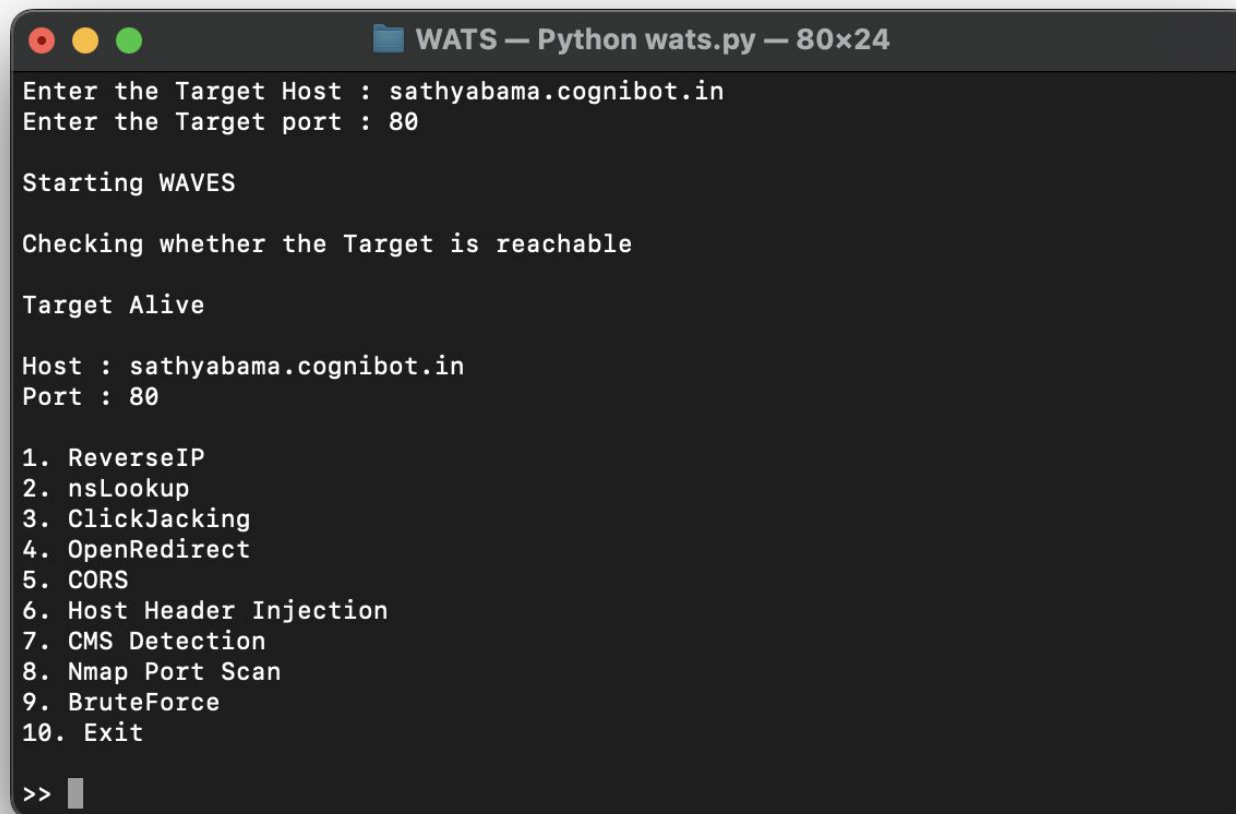
Checking whether the Target is reachable

Target Alive

Host : sathyabama.cognibot.in
Port : 80
```

Results and Discussion

The following are the vulnerabilities that are to be checked.



```
WATS — Python wats.py — 80x24
Enter the Target Host : sathyabama.cognibot.in
Enter the Target port : 80

Starting WAVES

Checking whether the Target is reachable

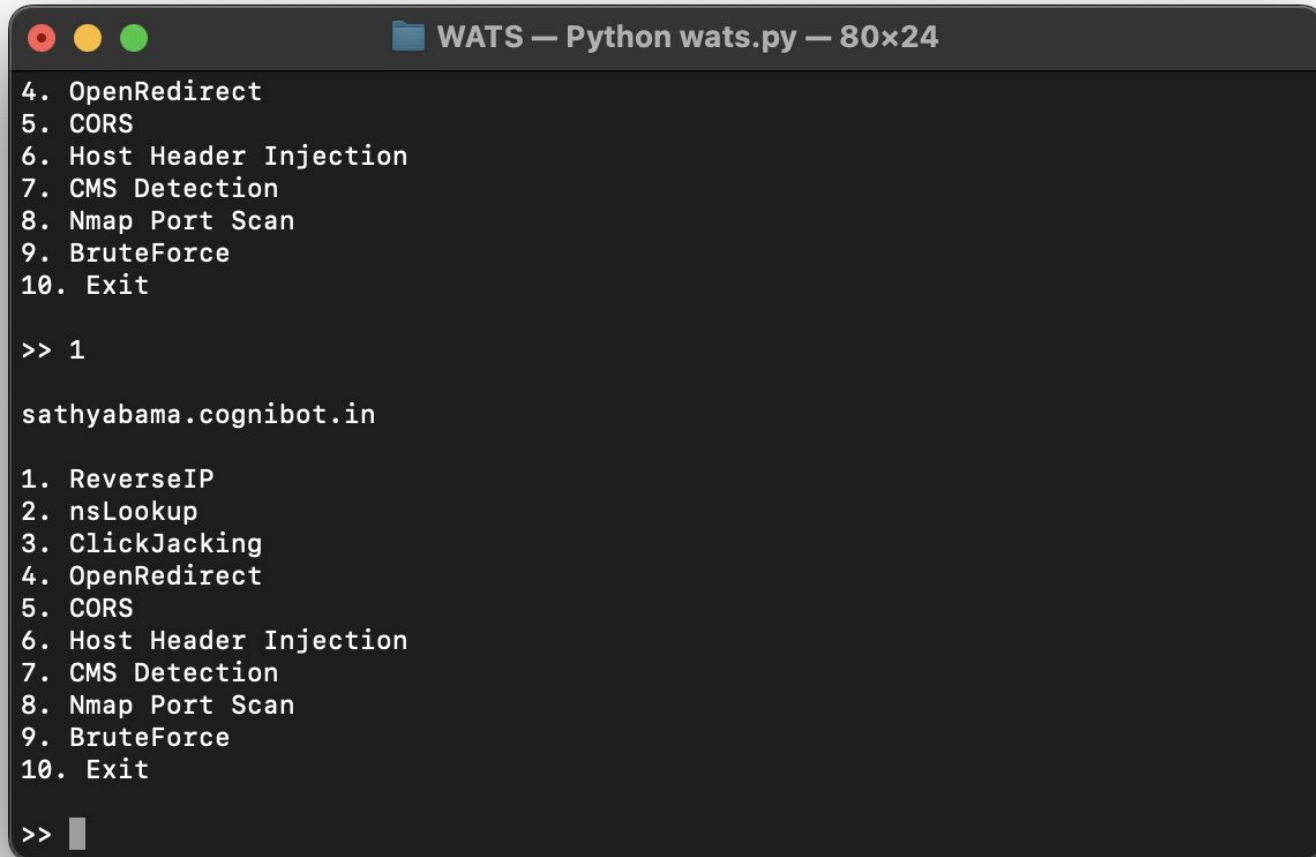
Target Alive

Host : sathyabama.cognibot.in
Port : 80

1. ReverseIP
2. nslookup
3. ClickJacking
4. OpenRedirect
5. CORS
6. Host Header Injection
7. CMS Detection
8. Nmap Port Scan
9. BruteForce
10. Exit

>> █
```

Results and Discussion



```
WATS — Python wats.py — 80x24
4. OpenRedirect
5. CORS
6. Host Header Injection
7. CMS Detection
8. Nmap Port Scan
9. BruteForce
10. Exit

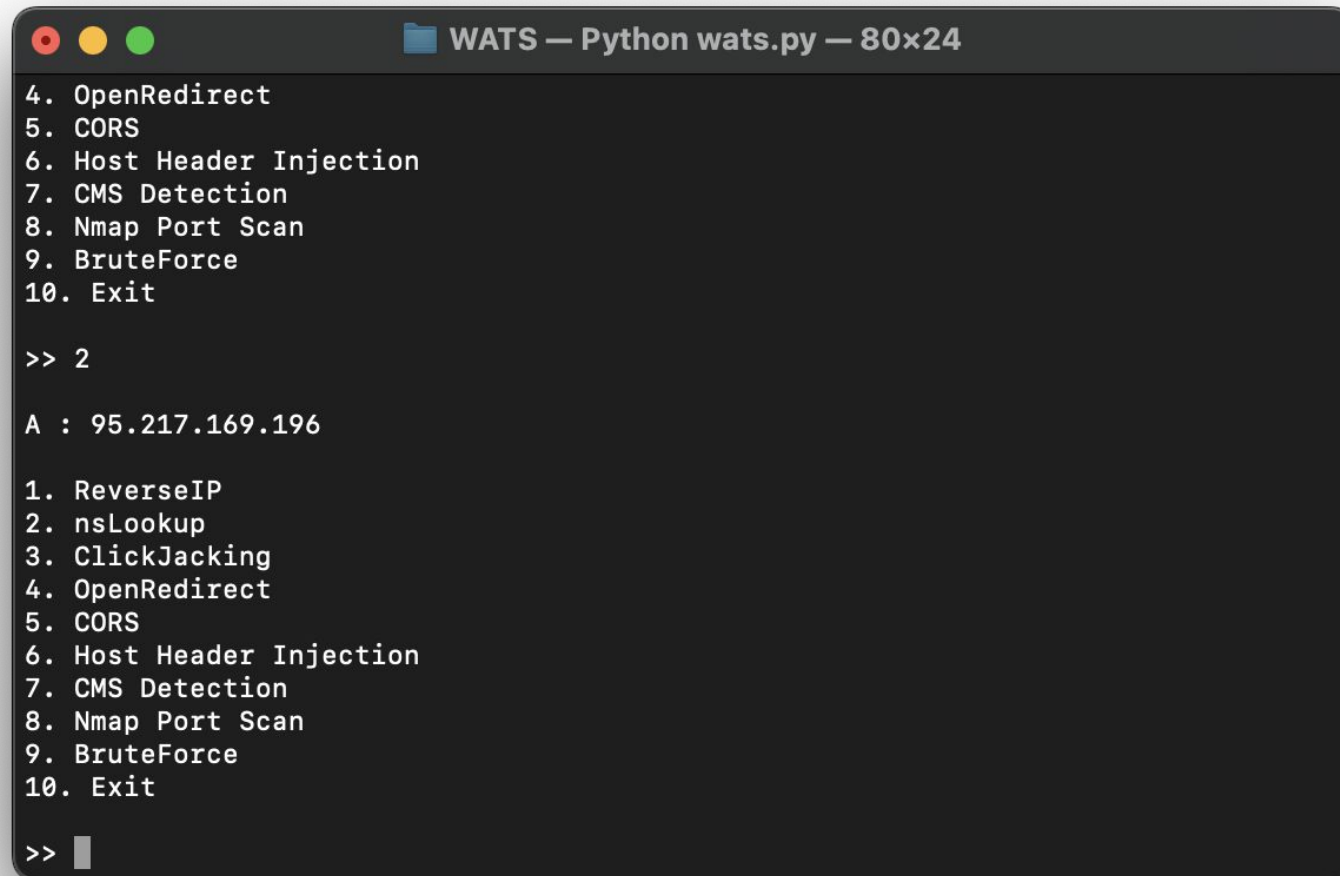
>> 1

sathyabama.cognibot.in

1. ReverseIP
2. nsLookup
3. ClickJacking
4. OpenRedirect
5. CORS
6. Host Header Injection
7. CMS Detection
8. Nmap Port Scan
9. BruteForce
10. Exit

>> █
```

Results and Discussion



```
WATS — Python wats.py — 80x24
4. OpenRedirect
5. CORS
6. Host Header Injection
7. CMS Detection
8. Nmap Port Scan
9. BruteForce
10. Exit

>> 2

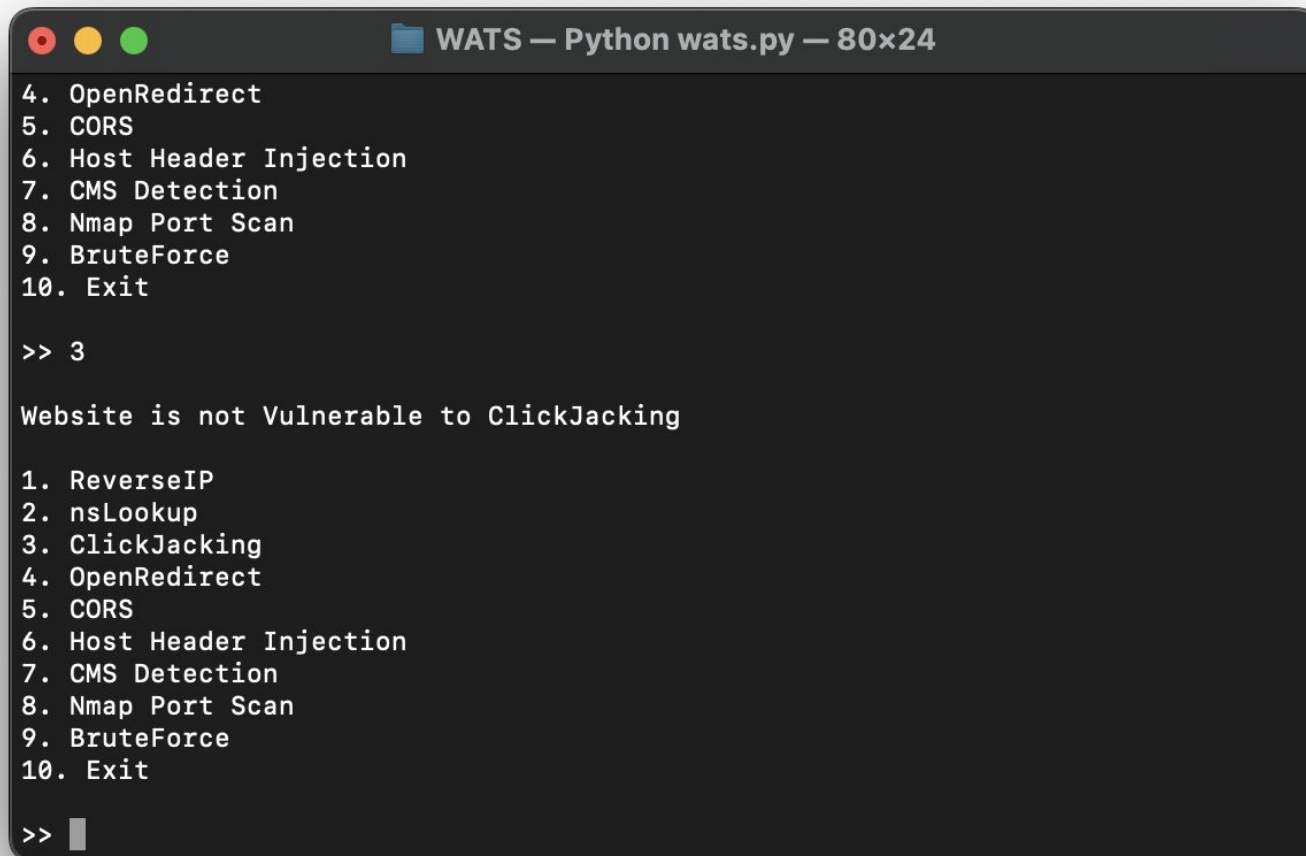
A : 95.217.169.196

1. ReverseIP
2. nsLookup
3. ClickJacking
4. OpenRedirect
5. CORS
6. Host Header Injection
7. CMS Detection
8. Nmap Port Scan
9. BruteForce
10. Exit

>> █
```

Results and Discussion

Websites will be checked for ClickJacking



```
WATS — Python wats.py — 80x24
4. OpenRedirect
5. CORS
6. Host Header Injection
7. CMS Detection
8. Nmap Port Scan
9. BruteForce
10. Exit

>> 3

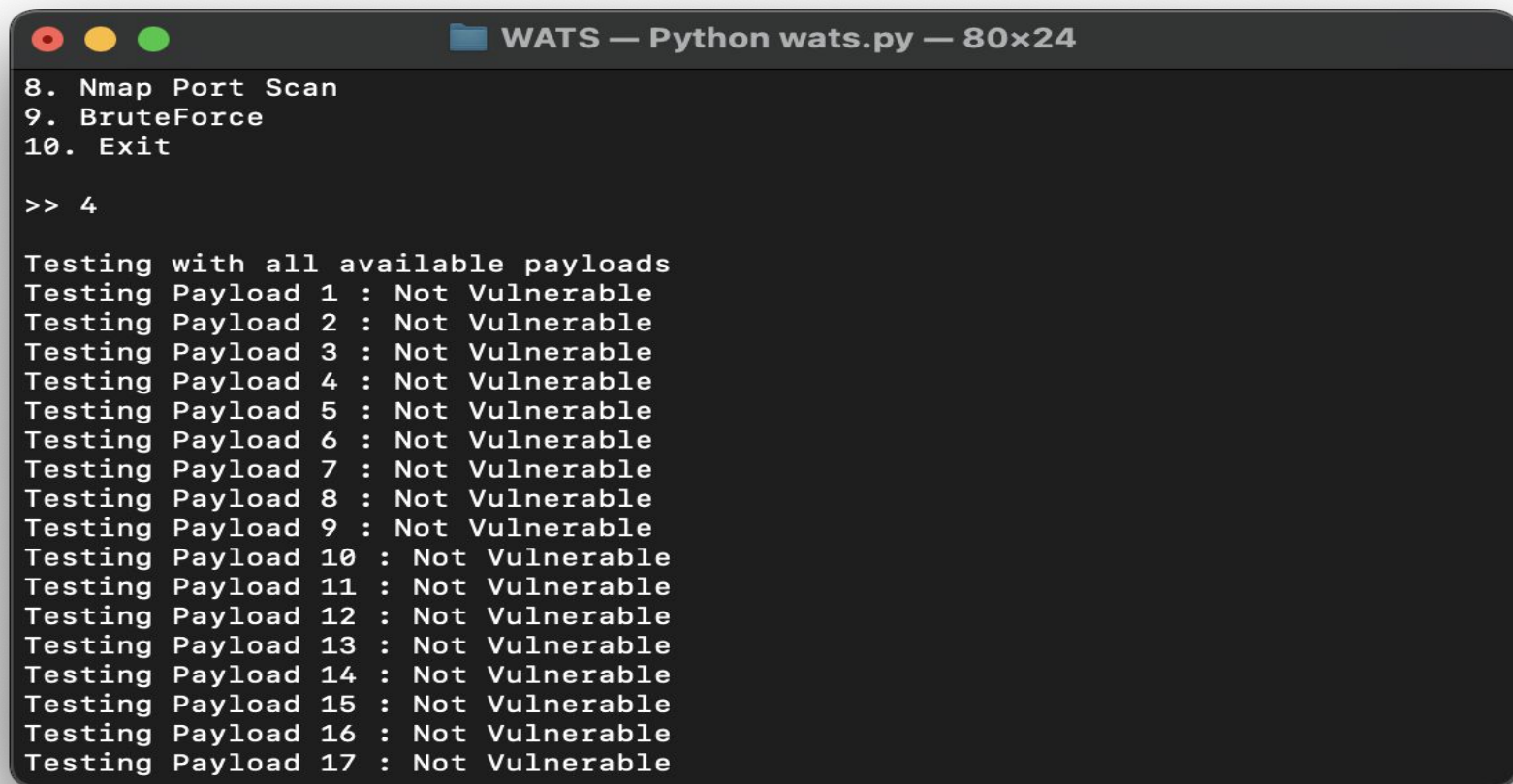
Website is not Vulnerable to ClickJacking

1. ReverseIP
2. nsLookup
3. ClickJacking
4. OpenRedirect
5. CORS
6. Host Header Injection
7. CMS Detection
8. Nmap Port Scan
9. BruteForce
10. Exit

>> █
```


Results and Discussion

Open redirect will check against all payloads available.



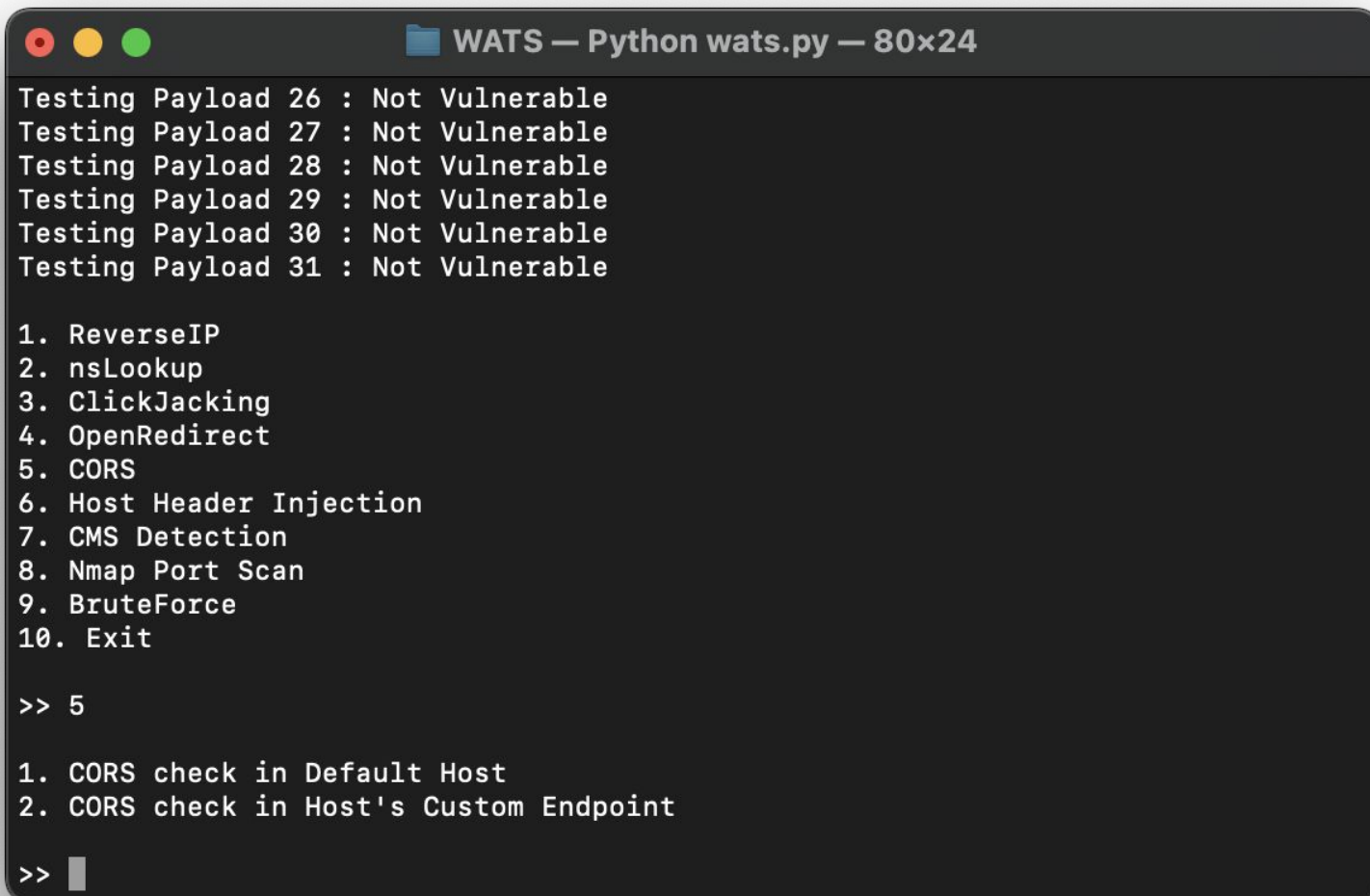
```
WATS — Python wats.py — 80x24
8. Nmap Port Scan
9. BruteForce
10. Exit

>> 4

Testing with all available payloads
Testing Payload 1 : Not Vulnerable
Testing Payload 2 : Not Vulnerable
Testing Payload 3 : Not Vulnerable
Testing Payload 4 : Not Vulnerable
Testing Payload 5 : Not Vulnerable
Testing Payload 6 : Not Vulnerable
Testing Payload 7 : Not Vulnerable
Testing Payload 8 : Not Vulnerable
Testing Payload 9 : Not Vulnerable
Testing Payload 10 : Not Vulnerable
Testing Payload 11 : Not Vulnerable
Testing Payload 12 : Not Vulnerable
Testing Payload 13 : Not Vulnerable
Testing Payload 14 : Not Vulnerable
Testing Payload 15 : Not Vulnerable
Testing Payload 16 : Not Vulnerable
Testing Payload 17 : Not Vulnerable
```

Results and Discussion

- In CORS we can check either through custom host or default host and it will check for cross origin resource sharing



```
WATS — Python wats.py — 80x24
Testing Payload 26 : Not Vulnerable
Testing Payload 27 : Not Vulnerable
Testing Payload 28 : Not Vulnerable
Testing Payload 29 : Not Vulnerable
Testing Payload 30 : Not Vulnerable
Testing Payload 31 : Not Vulnerable

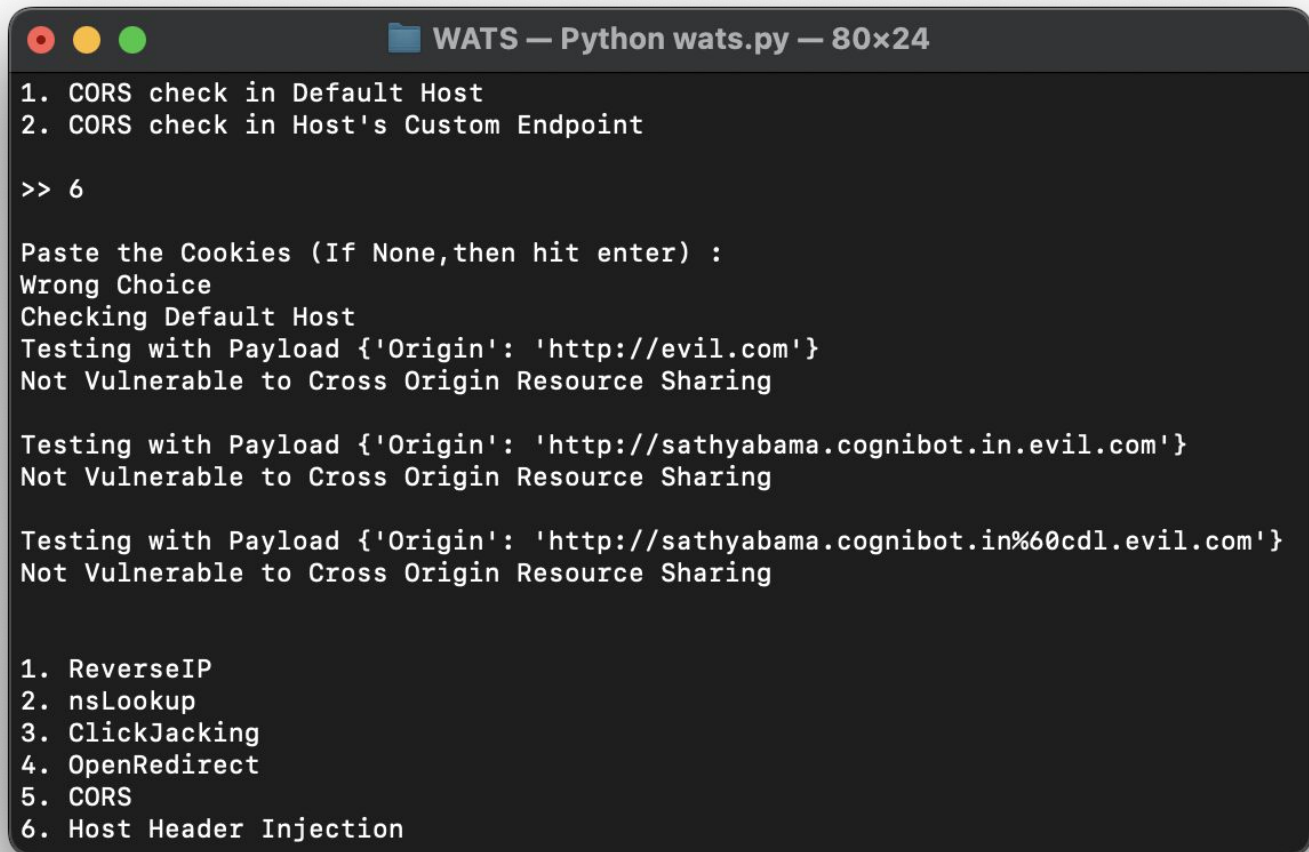
1. ReverseIP
2. nsLookup
3. ClickJacking
4. OpenRedirect
5. CORS
6. Host Header Injection
7. CMS Detection
8. Nmap Port Scan
9. BruteForce
10. Exit

>> 5

1. CORS check in Default Host
2. CORS check in Host's Custom Endpoint

>> █
```

Results and Discussion



```
WATS — Python wats.py — 80x24
1. CORS check in Default Host
2. CORS check in Host's Custom Endpoint

>> 6

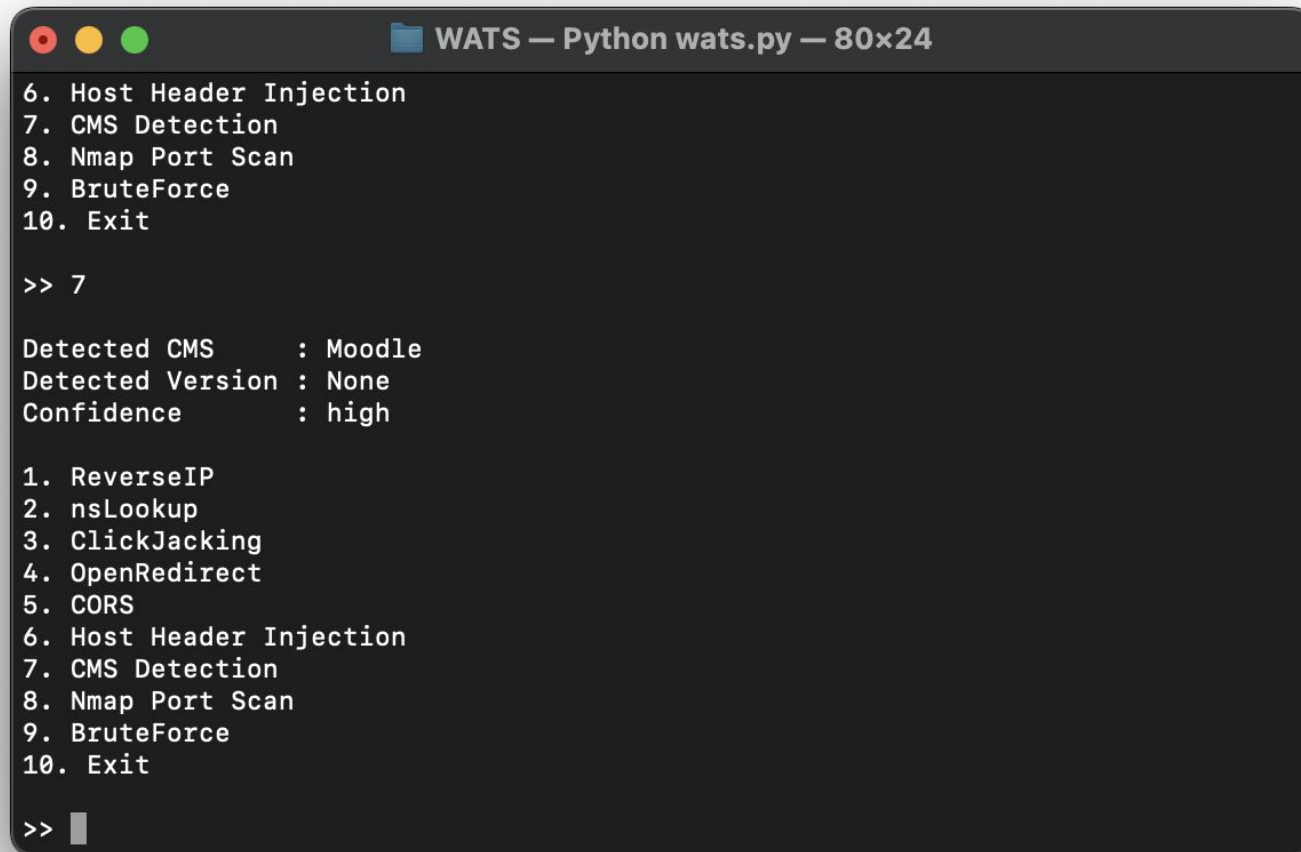
Paste the Cookies (If None,then hit enter) :
Wrong Choice
Checking Default Host
Testing with Payload {'Origin': 'http://evil.com'}
Not Vulnerable to Cross Origin Resource Sharing

Testing with Payload {'Origin': 'http://sathyabama.cognibot.in.evil.com'}
Not Vulnerable to Cross Origin Resource Sharing

Testing with Payload {'Origin': 'http://sathyabama.cognibot.in%60cdl.evil.com'}
Not Vulnerable to Cross Origin Resource Sharing

1. ReverseIP
2. nsLookup
3. ClickJacking
4. OpenRedirect
5. CORS
6. Host Header Injection
```

Results and Discussion



```
WATS — Python wats.py — 80x24
6. Host Header Injection
7. CMS Detection
8. Nmap Port Scan
9. BruteForce
10. Exit

>> 7

Detected CMS      : Moodle
Detected Version  : None
Confidence        : high

1. ReverseIP
2. nsLookup
3. ClickJacking
4. OpenRedirect
5. CORS
6. Host Header Injection
7. CMS Detection
8. Nmap Port Scan
9. BruteForce
10. Exit

>> █
```

Results and Discussion

- NMAP port scans also can be custom or default(22-443).



```
>> 7

Detected CMS      : Moodle
Detected Version  : None
Confidence       : high

1. ReverseIP
2. nsLookup
3. ClickJacking
4. OpenRedirect
5. CORS
6. Host Header Injection
7. CMS Detection
8. Nmap Port Scan
9. BruteForce
10. Exit

>> 8

1. Scan Default Ports (22-443)
2. Enter Custom Range
3. Back to Main Menu

>> █
```

Results and Discussion



```
WATS — Python wats.py — 80x24
1. ReverseIP
2. nsLookup
3. ClickJacking
4. OpenRedirect
5. CORS
6. Host Header Injection
7. CMS Detection
8. Nmap Port Scan
9. BruteForce
10. Exit

>> 8

1. Scan Default Ports (22-443)
2. Enter Custom Range
3. Back to Main Menu

>> 9
Please choose an Appropriate option
1. Scan Default Ports (22-443)
2. Enter Custom Range
3. Back to Main Menu

>> █
```

Conclusion

- Web Application Threat Scanner (WATS) work by automating several processes.
- Used to find the websites bug and after that it shows the types of bug on that websites. It is a quick check for potential security vulnerabilities, using automated approach.
- It saves time ,money because if the vulnerabilities is exploited by the malicious attackers, it affects the reputation of the company/organizations
- This scanner widely cover end-users such as security learners in cyber security and professionals

References

- [1] Harrell, Christopher R., et al. "Vulnerability Assessment, Remediation, and Automated Reporting: Case Studies of Higher Education Institutions." 2018 IEEE International Conference on Intelligence and Security Informatics (ISI). IEEE, 2018.
- [2] Wang, Yien, and Jianhua Yang. "Ethical hacking and network defense: Choose your best network vulnerability scanning tool." 2017 31st International Conference on Advanced Information Networking and Applications Workshops (WAINA). IEEE,

References

- [3] Holm, Hannes, and Teodor Sommestad. "Sved: Scanning, vulnerabilities, exploits and detection." MILCOM 2016-2016 IEEE
- Military Communications Conference. IEEE, 2016.
- [4] Appiah, Vincent, et al. "Survey of Websites and Web Application Security Threats Using Vulnerability Assessment." (2018)

References

- [5] Aarya, P. S., et al. "Web Scanning: Existing Techniques and Future." 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS). IEEE, 2018.
- [6] Gorbenko, Anatoliy, et al. "Experience report: Study of vulnerabilities of enterprise operating systems." 2017 IEEE 28th International Symposium on Software Reliability Engineering (ISSRE). IEEE, 2017.
- [7] Yadav, Ravinder, and Aakash Goyal. "Web Application Security." International Journal of Computer Science and Mobile
- romnlltln-Vol1 Tccru 10 (2014): 349-355.

References

- [8] Kushe, R. "Comparative Study of Vulnerability Scanning Tools: Nessus Vs Retina." Security & Future 1.2 (2017): 69-71.
- [9] Im, Sun-young, et al. "Performance evaluation of network scanning tools with operation of firewall." 2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN). IEEE, 2016.
- [10] Stephenson, P. "Tenable Network Security Nessus." (2015).
- [11] Bairwa, Sheetal, Bhawna Mewara, and Jyoti Gajrani. "Vulnerability Scanners-A Proactive Approach To Assess Web Application Security." arXiv preprint arXiv:1403.6955 (2014)