ISSN No:-2456-2165

Adaptive Penetration Test Method

Volkan DORTKARDES
Computer Engineering Department
Gebze Technical University
Kocaeli, Turkey

İbrahim SOGUKPINAR
Computer Engineering Department
Gebze Technical University
Kocaeli, Turkey

Abstract:- The complexity of the Information systems is increasing day by day. This leads to more security vulnerabilities in IT Systems. Attackers use these vulnerabilities to penetrate for the target's system. It is better to find these vulnerabilities in advance before the attacker does. The power of vulnerability assessment is often overlooked. Penetration testing is a set of activities undertaken to identify and exploit vulnerabilities. It helps to verify the effectiveness of the security measures implemented. When the existing penetration test approaches and methods are examined, another issue that is overlooked is that not all steps are performed adequately in the penetration test process. In the environments where penetration tests will be performed, the number of attack vectors may be limited only if progress is made depending on the capabilities of the Penetration Test Specialist and / or some important parts may be missing due to forgetting or lack of ability. In this work, my studies on how to use the comprehensive penetration test software in cyber defense technology, how to apply more attack vectors and more results can be obtained during the tests with a new perspective and how to automate the detection of social engineering weaknesses with an automated application tool. is located together. Automated test results were compared with the results of the manual tests performed in 7 different environments and the superior parts are indicated in the results section. I also talked about which tools will be used for which purposes and at what stages, and I have automated several steps of penetration testing.

Keywords:- *Vulnerability Assessment, Adaptive Penetration Test, Automated Application Security, Computer Security And Computer Ethics.*

I. INTRODUCTION

Penetration testing and security assessments in general are critical for all companies based on an IT infrastructure. However, there are some problems. Penetration testing procedure may take several weeks or months, depending on the size and complexity of the targeted network and the level of detail the customer wants. A company can have up to twenty users, and all they want is a basic vulnerability scan to make sure they don't have a major problem with system configurations. In this case, the scan can be done in one day or several days with very little interaction other than entering the target systems and starting the scan. In another scenario, in a company with several thousand users and their infrastructure, unlike the previous scenario, the penetration testing process may take several weeks or months due to the complexity of network / application architectures and many different attack paths.

If we need to express it in detail, some weaknesses that can be easily detected using automated tools can become virtually impossible to detect manually, information security experts say. For example, inorder for Host Header injection to be detected, 4 requests must be sent separately for each URL. Or, when Sensitive data Leak is handled, automated scanning is needed because all pages of the web application need to be attacked by the bruteforce. In addition, some of the tests performed automatically can't reach the number of outputs that can be obtained automatically, even when performed manually. The number of outputs that will be obtained can both provide the information security expert with more clues, give him the chance to create more attack-vectors, and prevent possible output losses caused by human factors(forgetfulness, lack of information, etc.).

Basically, a company hires security experts to evaluate and hack their networks, servers and services so that malicious users can do the same. Penetration testing devices will provide a report on network, service and application vulnerabilities. The report may also include how penetration testers can access IT infrastructures and applications to gain access to specific accounts or systems. The report will also provide suggestions on how to correct these gaps. This allows the company to secure its networks, services and applications against various future attacks.

In this work, an adaptive penetration test method has been proposed for IT systems. We will cover adaptive pentesting tool. This tool will enable you to collaboratively conduct penetration tests efficiently and effectively against variable target environments. It allows us to create different attack vectors by providing more results and yields than manual pentest.

Rest of the paper are organized as follows. Fundamentals and related works are presented in the section 2. Proposed method is explained in the section 3. Experimental results are presented in the Section 4. Last Section is Conclusion and Future Works. Ease of Use

II. FUNDAMENTALS OF PENETRATION TESTING

The penetration test is a set of activities that involves examining a system for weaknesses, identifying these weaknesses as the impact of abuse, and finally preparing a report for the owners of the system. The life cycle of the penetration test consists of the following steps [2]:

- Scope
- Discovery
- Vulnerability detection
- Information analysis and planning
- Implementation of penetration test
- Privilege upgrade
- Results analysis
- Reporting
- Clean up the traces left

Penetration testing techniques are three types and each type presented as follows has its own characteristics.

- White box: the security expert who performs the test on this test model has complete knowledge of the network configuration of the test network and the test /system network. This test is usually performed from the internal network. White box testing requires an in-depth understanding of the test network or system and yields better results.
- Black Box Testing: in this technique, the expert performing the test has no prior knowledge of the network architecture or the systems of the test network. Black box testing is performed from external networks to internal networks. The person performing the test should use his or her expertise and skill.
- Gray box test: the person performing this test does not have detailed knowledge of network architecture, but knows some information about testing network and system configuration. It's actually a mixture of the previous two methods. It can be performed from both the internal network and the external network.

When performing these tests, the following operations are performed as an implementation:

- Information Gathering,
- Network Mapping,
- Vulnerability Scanning,
- Penetration into the system “Exploit performing ”,
- Authorization Upgrade,
- Penetration to Other Networks,
- Protecting Access,
- Attacks on Web-Based Applications,
- Social Engineering Attacks,
- Cleaning Footprints,
- Reporting.

Static analysis is also one of the common techniques used to find security errors in application code. The difference from penetration testing is that a system runs in a white box by analyzing the source code and identifying security vulnerabilities. Due to their different characteristics, both techniques report their findings differently [3]. In the paper of Mariano and Riccardo, a report based on a static analysis-based security tool was compared with a report based on penetration testing, and scenario-based studies were conducted on how reporting could be more useful.

A. Penetration Testing Methods And Related Works

Social engineering has emerged as a serious threat to social communities and is an effective tool for attacking information systems. The services used by today's information workers pave the way for sophisticated social engineering attacks. The increasing trend towards BYOD (bring your own device) policies and the use of online communication and social media in private / business environments make the problem worse. In globally operating companies, teams no longer work together geographically, but work full-time. Despite the decrease in personal interaction, the emergence of numerous tools used for communication (e-mail, IM, Skype, Dropbox, LinkedIn, Lync, etc.) creates new attack vectors for social engineering. Recent attacks on companies such as the New York Times and RSA have shown that targeted spearfishing attacks are an effective and evolutionary step in social engineering attacks. [5] Katharina and colleagues provide a comprehensive overview of advanced social engineering attacks on the knowledge worker, as well as a well-known taxonomy of social engineering attacks.

This study leveraged a long series of Applied Research studies aimed at automating and optimizing penetration testing processes and systems, in particular vulnerability assessment (vulnerability analysis) and penetration testing. [13.15]. Among the most important contributions to this topic, here we present a summary of previous research focusing on the approaches and contributions adopted. Initially, researchers were interested in the planning phase. Some studies have been applied in industrial penetration testing systems and frameworks, while others have remained research ideas [13,14].

Although people could not use social networking sites to communicate with each other, in fact the privacy of the user's information is confiscated and often neglected, although the services provided are considered to be an advantage [4]. The approach put forward by Markus and his colleagues in the study takes a step further with the automation of classical social engineering. They conducted two experiments to evaluate the proposed attack cycle and prototype applications (ASE bot). In their first attempt, they examine their boats' ability to gather information. In their second assessment, they performed a Turing test. The promising results of their assessments emphasize the possibility of effective and effective social engineering attacks using automated social engineering boots.

The attacks that Markus and his colleagues carried out with the automated social engineering tool they created in their work are very striking. The first is automated social engineering work on five successful large Swedish-based

multinational companies, where they conducted the penetration test study. These institutions:

- High-tech company
- IT company
- Scandinavian finance company
- Industrial engineering company
- It's a telecommunications company.

The following chart shows the cases of finding victims in these firms by means of social engineering.

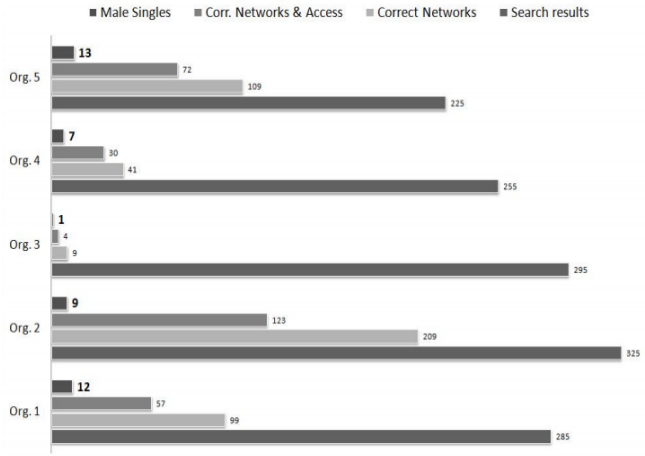

Fıg 1:- Finding targets with automated social engineering tool [4]

The information gathering process was observed to take 16 minutes (Organization 3), 65 minutes (organization 2) and an average of 44 minutes per organization. In the following two graphs, their messaging with Target groups is numerically expressed using different profiles created with this automated tool.

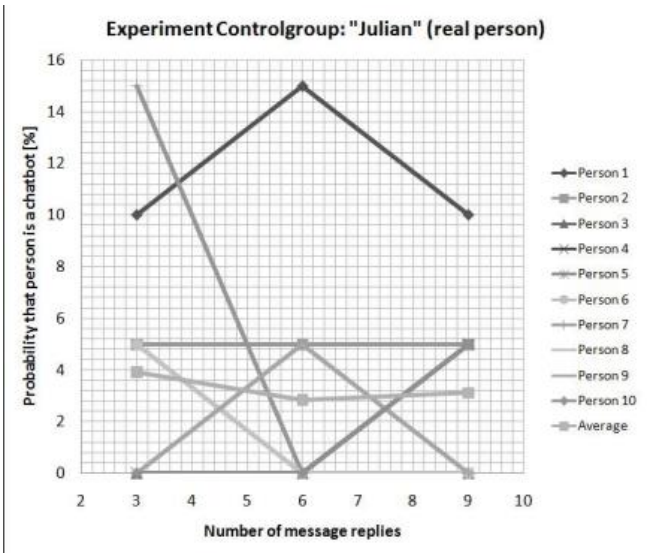

Fig 2:- the results of a conversation between the Julian character and the target's real people [4]

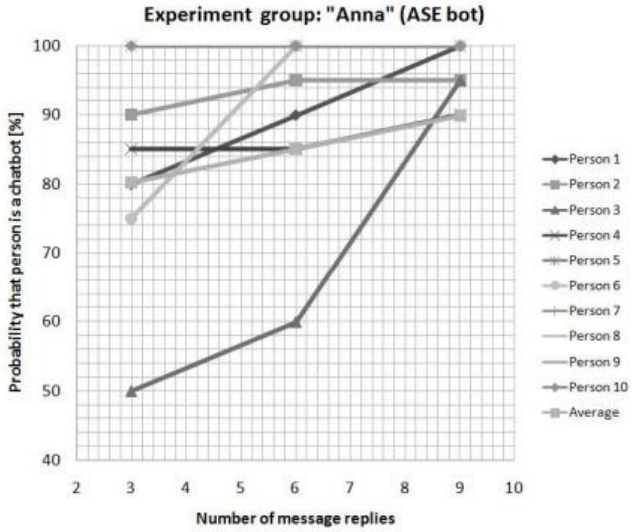

Fig 3:- messaging results for target users with the character Anna [4]

As can be seen from these studies, the attempts to be automated are often focused on automating social engineering attacks. In my study, the penetration test rather than a test which is only one step of Information Collection social engineering: Exploration: network scanning and Vulnerability Assessment results and make further automated by considering the weaknesses in OWASP 10 Steps The Biggest you could get , and would give more clues to the Information Security Professional think is going to gain benefit in terms of time; it supports the idea that the automated social engineering approach is an innovative and effective tool both in the presence of similar studies and in the automated penetration test tool, which I intend to establish success rates.

The automated systems require the permanent control of a human PT expert and often fail to produce acceptable results in medium and large assets context because of the significant number of operations required to cover the entire network [9,10,11,12]

Mohamed C. Ghanem and Thomas M. Chen mainly performed penetration test implementation related to the network perspective and focused on machine learning and especially the application of Reinforcement Learning techniques to make penetration test application intelligent and efficient.[9] In this research, the probabilistic output of the penetration Test action (screening, fingerprinting, abuse) was a crucial factor in which they considered allocating sufficient probabilities for transitions and observations to reflect real-world penetration test practice. Therefore, the NIST National Vulnerability Database known for everyone that creates a reliable online catalog (CVSS [16] and Common Vulnerabilities and exploits (CVE) [17] are two well-established source of using a standard and cross-validated the method they chose. Different types of operating systems, software, and applications associated with proven vulnerabilities: the use of such resources, rich content, easy accessibility and regular updates and the calculation of the scoring function

and associated probabilities, such as cvssv3 and is motivated by the existing mechanism. Each passage or observation is detailed in [16,17].

As can be seen from these studies , the attempts to be automated are often focused on automating social engineering attacks. In my study, the penetration test rather than a test which is only one step of Information Collection social engineering: Exploration: network scanning and Vulnerability Assessment results and make further automated by considering the weaknesses in OWASP 10 Steps The Biggest you could get , and would give more clues to the Information Security Professional think is going to gain benefit in terms of time; it supports the idea that the automated social engineering approach is an innovative and effective tool both in the presence of similar studies and in the automated penetration test tool, which I intend to establish success rates.

III. ARCHITECTURE OF ADAPTIVE PENTESTING TOOL

The ISO / IEC 25010 : 2013 quality standard defines a product pattern by separating software features into sub-features, consisting of eight important features.

This product has been developed by considering the weaknesses of OWASP 10, regardless of its methodology or framework. Here we investigated whether the tool in question was interested in the complete leak test and created a tool to support our approach to assess the Adaptive pentesting approach. The adaptive pentest tool has been created by keeping the standard in mind.

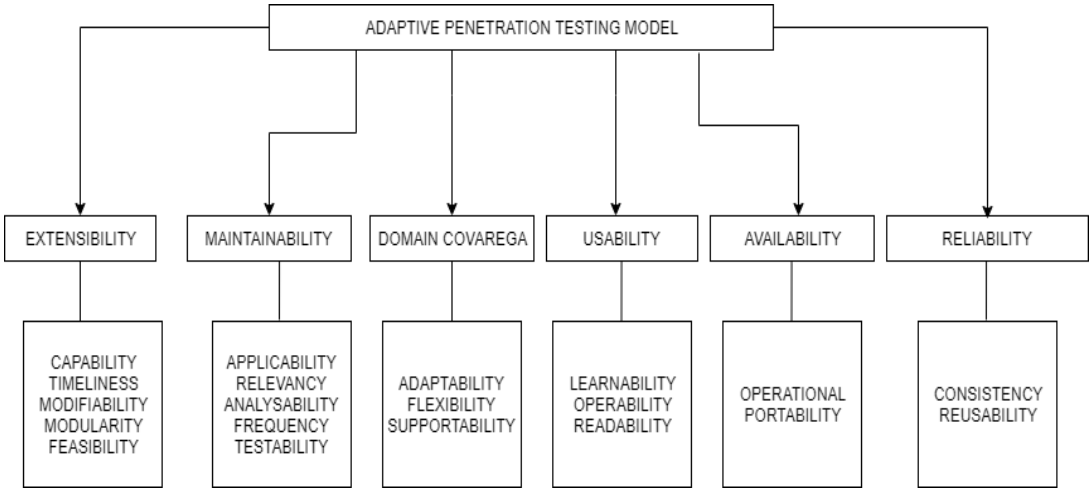

Fig 4:- Adaptive Penetration Testing Quality Model

The pentesting of a web application can be automated with a website in order to conserve time and cover a wider scope compared to manual pentestıng. This tool uses scans the wesite i.e starts visiting all the pages on the website one at a time and stores the address of all the unique url it encounter on every page and stores it in a deque to visit for next time. The tool is made to find the following vulnerabilities.

- XSS
- SQL Injection
- XXE
- SSRF
- Subdomain Takeover
- Missing security Headers
- Host Header Injection
- CORS
- Sensitive Data Leak

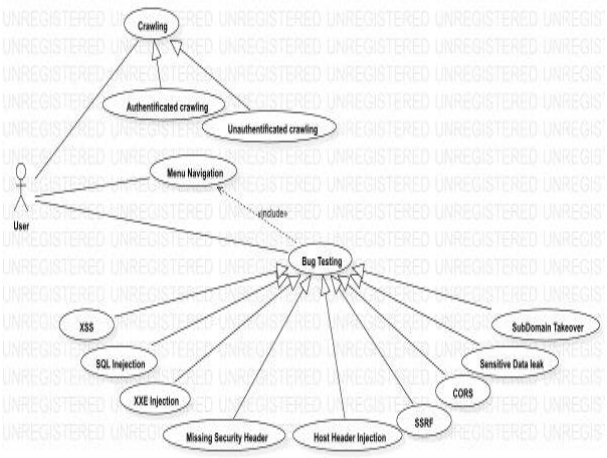

Fig 5:- Use Case Dıagram of the Adaptıve Pentest Tool

Crawling :The crawler function in scraping.py is called it takes two arguments the website name and the file in which the urls which are crawled have to be saved . There are two crawling function based on the authentication: as authenticated user and as unauthenticated user.

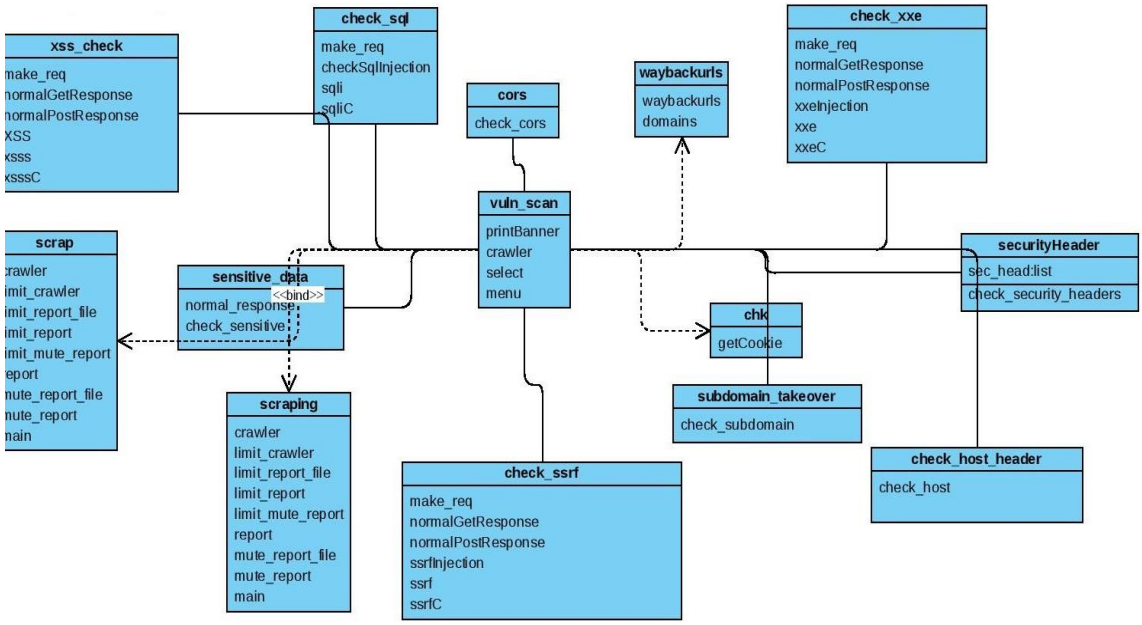

Fig 6:- Class Diagram of the Adaptive Pentest Tool

Information Flow: From vul_scan entity data is going to be scrapped entity. In there scrapping entity calls crawler module to crawling the website and generate a report file.The relation between VuL_scanner and scrapping is an optional means scrapping part depends on the user. If the user wants to crawl the website then only this module will be working otherwise not.

After scraping data is going to the chk and wayback_urls entities .In chk entity scanner gets cookies of the application . The relationship with Vul_scan entity is optional .

After that data is going from vul_scan entity to all other entities after one by one with the interaction of the user .Serial is depends on the user of the scanner.one can perform xxs check first on the other hand other can start with sql_Check .Its totally depends on the user .

Working procedure: Check_sql entity: (Relationship with Vul_scan entity is one to many.) Make a request to the server then Get a response from the server.then inject normal sql injection to the form input .If it does not work then the modules go for the sqli,sql blind injection mode.If the modules find any vulnerabilities while performing scan it will show a message to the user .

XSS check: one to many relation

First the module sends a request to the server to connect .Then check the normal get and post request .Then start the XSS injection attack to url ,or form input .If finds any vulnerable then shows an alert to the user.

check__ssrf: one to many relation First the module sends a request to the server to connect. Then check the normal get and post request. Then start the ssrf injection attack to url ,or form input .If finds any vulnerable then shows an alert to the user.

check__xxe: one to many relation First the module sends a request to the server to connect .Then check the normal get and post request .Then start the ssrf injection attack to url ,or form input .If finds any vulnerable then shows an alert to the user.

Sensitive_data: In this part this modules looking for robot.txt, password.txt files.

Subdomain takeover: find for subdomain under the main ip address. cors: check cors error by send a spoof request to server.

check_host header and security header: find vulnerabilities in the header by inspecting then and match with the signature databases.

A. Parts of Adaptıve Pentestıng Tool

- *Crawling :- is the first step and it is the most important step as soon as the script starts , the crawler function in scraping.py is called it takes two arguments the website name and the file in which the urls which are crawled have to be saved . There are two crawling function based on the authentication*
- *Crawling the website as unauthenticated user (Takes two arguments website name and the filename to save processed urls)*
- *Crawling the website as authenticated user (Takes three arguments website name , filename to save processed urls and the cookie for the authenticated user)*

The crawler starts visting the website from the index page and finds all the anchor and HTML tags with src attribute and the links in javascript it does this by using the python requests library and beautifulSoup library used to scrape data from the website.

After visiting all the links it saves it in the demofile.txt internally and it also takes care for authenticated user that it does not loose the sesssion so, it avoids visiting the urls which contain logout or signout keywords in them.

To make sure it visits all the url the it uses a script waybackurls.py it searches the web archives for the urls which belong to the target website which are mainly from the starting i.e when the website was created till now . It is a very rich source for urls which usually a manual pentest might miss it can contain the function that was used by the website in the starting and they forgot to remove it from the website and it vulnerable to some attack.

Fig 7:- Interface of Adaptive Pentest Tool

- *Menu :- it is the interface for the user , it helps the user with what is the tool does and makes the navigation for the user very easy it helps the user choose which bug they want to test in the website . There are many tools which do not provide the menu and take very long time to produce the result unlike them here you can check only particular bug which is very time saving and helps you conserve time and gives you a list to checklist of bugs . A manual tester not having a checklist might miss to test some bug . It uses figlet a linux utility to make the animated text in the linux terminal to give it an animated look .*

- *Testing for Specific Vulnerabilities*

As we stated at the beginning of the study, we focused on the most known weaknesses of OWASP and included 9 weaknesses in this section.A separate module has been created for each of these weaknesses and their working philosophies are listed below.

XSS : So first option to choose from the menu is the xss it stands for Cross-Site Scripting it is the most common bug in the mordern web application , very large enterprises are also vulnerable to this bug type . In XSS an attacker is able to execute the javascript on the victims browser . The script on entering the option 1 intiates the xss function in xxs_check.py file it takes two argument the website name and the cookie value if provided and then uses the form_input.py file to get all the urls with form tag in them and their input types names . After storing all the urls and input types in the list it uses the loop to get a single url and its and data and test if the url is vulnerable to xss or not . It cheks a single url is vulnerable by sending a unique such as batman in the input and checks if the response contains the same string in the response or not , if the string is present then it sends <>/() in the input and see if they are encoded or not . If not it has a list of xss polyglots which it submits in the request and checks if the string is reflected in the response if present in the string it print on the terminal the url is vulnerable to xss with the payload used in the attack .

*SQL Injection : It is the second option in the menu it the very old bug on found on the website and it is very less common now days because website using web frameworks now days and CMS such as Wordpress . It is very severe bug if a attacker is able to exploit this he can access data of any user and even change the data. In the script on choosing 2 option sqli function is called present in check_sql.py it takes two arguments the website name and the cookie if provided and gets input and urls as xss . After getting the links and the take for it sets up a difference between the normal request and the malicious request by ignoring the response if it contains terms such as error , exception , SELECT and many more and then uses the payload such as ' . " , * and many more to check if the what the response is if the response status code is like 500 502 etc then it flags the url and print on the terminal as sql injection found with the payload .*

XXE Injection : It is the third option in the menu it is a very new bug in the modern web found on the website which contain the xml file to fulfill store the data of the user or to use xml to respond to the user with the requested data . So to find XXE using the script we use call xxe function in check_xxe.py file it takes two arguments the website name and the cookie if provided . In this it changes content type of the request to application/xml for each request and check the difference between the length and status code of the response of the request with normal data and the request with the xxe payload and if the difference shows the possibility of the xxe attack it flags it as the xxe vulnerablility found the specific url with the specific payload . It uses xxe polyglots as the payloads .

SSRF : It is the fouth option in the menu it is also a very new bug types in this the attacker if able to exploit it can get access to the private file on the server and port scan the network on which the server is present . To find this bug in the script we call the function ssrf in the check_ssrf.py file it takes two argument as website and the cookie if provided . In this we develop the differnce between the normal request and malicious payload . If there is the differnce between the status code and response length then we flag it as the ssrf bug found the specific payload with specific payload . We uss ssrf polyglots as the payload .

ISSN No:-2456-2165

*CORS : It is fifth option in the menu it is also a fairly new bug type in this if the website allows different origin other than the domain it is having to send it a request and send response to that malicious origin than an attacker would change the origin header for the victim request and receive the response on his malicious server and make changes in the javascipt and compromise the user . To find this bug we change the Origin header of each request and and check if Allow-AccessControl header is * or null or the name of the website in origin and flag it and print on the terminal as CORS found the url .*

Missing Security Headers : It is the sixth option in the menu it check if all the security headers which a website response should have to make it secure against the attacks are present or not . To find this the script call the check_security_header function in security_header.py file . It check the X-Frame-Options ,X-XSS-Protection and many more header if present or not if not present flags it as vulnerable and prints on the terminal as the missing particular security header .

Subdomain takeover: It is the seventh option in the menu in this a attacker an attacker can claim a subdomain . To find this by the function check_subdomain is called in the subdomain_takeover.py file . The script uses the https://crt.sh/?q=%25 to search the subdomains of the target domain and then request all found subdomain and if the respose status code is 404 or the string Not Found are present then the script flags it as subdomain takeover found and prints it on the terminal .

Sensitive Data Leak : It is the eighth option in the menu in the attacker could get access to the sensitive file on the server which the developers forgot to remove or is by default present on the server . To find this bug we use a list of file which are by default present on the server and contain sensitive file . As we request for the specific file and its response status is 200 we request it as sensitive data leak on specific url and print it on the terminal .

Host Header Injection : It is the ninth option in the menu in this an attacker can manually divert the code to produce the hacker's desired output, simply by editing the host header. Most probably web servers are configured to pass the host header to the first virtual host in the list without proper re organisation. So It is possible to send the HTTP requests with arbitrary host headers to the first virtual host. In that case, if we specify an invalid Host means that the web server process it and pass the invalid host header to the first virtual host in the list. To find this bug we use function check_host in the check_host_header.py file it requests the url with Host and X-ForwardedHost header in the request and checks the response if the malicious host is found it flags it as the host header injection found with the url .

IV. REALIZATION AND EXPERIMENTAL RESULTS

A. Properties of the Test Environments

I tested the automatized application in 7 different environments with enough different parameters and compared the results with the tests I did manually. Test environments are included below:

i. http://testphp.vulnweb.com

This environment is an example PHP implementation that is deliberately weak against web attacks. Designed to help you test Acunetix. It also helps you understand how developer errors and poor configuration can allow someone to enter your website. It can also use other tools and manual safety testing to test your skills. Tip: we can look for potential SQL injections, cross-site scripting (XSS), and cross-site request spoofing (CSRF), and more.

ii. https://demo.testfire.net/

Altoroj is an example banking J2EE web application. It shows what happens when Web applications are written with application functionality in mind and not with application security in mind. It is a simple and orderly platform for learning and learning more about real-life application security issues. AltoroJ uses standard Java and JSP functionality without using any additional framework. While the most of real-life applications use frameworks, the same application security principles apply in both cases. Frameworks can be difficult to understand and learn for someone with no particular familiarity. There are many large and complex "old" Java web applications, not so-called, that are very similar to Altoroj(but of course many are almost repeatedly more complex).

iii. http://zero.webappsecurity.com

The Free Online Bank website was published by Micro Focus Fortify only to demonstrate the functionality and effectiveness of Micro Focus fortify's WebInspect products in detecting and reporting Web application vulnerabilities. This site is not a real banking site and the similarities to third party products and / or websites are purely coincidental. This site is offered "as is" without warranty of any kind, express or implied. Micro Focus Fortify does not accept any risk associated with your use of this website. Use of this website indicates that you have read and accepted Micro Focus fortify's terms of Use.

iv. http://testhtml5.vulnweb.com/

This is an HTML5 implementation with design-to-malicious design. This application was created so that you could test your Acunetix, other tools, or manual penetration testing skills. Application code is prone to attacks such as Cross-Site Scripting (XSS) and XML external presence (XXE). The links provided on this site are not linked to the site and are only available here as examples.

v. Typhoon Vulnerable Machine

Typhoon vulnerable VM is a virtual machine that comes with various vulnerabilities that provide a laboratory environment for researchers looking to improve their skills in cybersecurity. Typhoon VM v1 was developed by the Prisma CSI team to provide a mini-laboratory environment for practical Penetration Testing Training offered by the company. You can also download and install the virtual machine on your system, giving you a chance to gain some practical skills in this area.

vi. Damn Vulnerable Web Application

Damn malicious Web application (DVWA) is a PHP / MySQL web application that is vulnerable. Basic objectives of IT security experts to test the skills and tools in the legal environment, help web developers better understand the process of securing their web applications and help teachers / students in a classroom environment to teach you how to web application security / learn to help.

vii. Bwapp

bWAPP, or a buggy web application, is a free and open source deliberately insecure web application. Helps security enthusiasts, developers and students discover and prevent web vulnerabilities. bWAPP prepares to carry out successful penetration testing and ethical hacking projects. What makes bwapp unique is that it has 100 web vulnerabilities. OWASP covers all known major web bugs, including all risks from the top 10 project. bWAPP is a PHP application that uses a MySQL database. It can be hosted on Linux/Windows with Apache/IIS and MySQL. It can also be installed with WAMP or XAMPP. Another possibility is to download bee-box, a custom Linux VM pre-installed with bwapp.

B. Performed Tests

➢ http://testphp.vulnweb.com

Manual tests performed in the environment identified the following weaknesses : XSS, SQL Injection, Missing Security Headers, Subdomain takeover, Sensitive Data Leak, Host Header Injection.

Automated tests with scripts performed in the same environment were much shorter than manual tests and identified the same weaknesses.

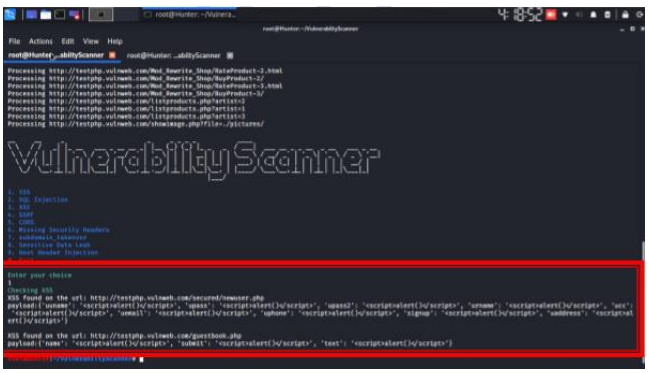

Fig 8:- Execution of CSS scripts over automatic tool

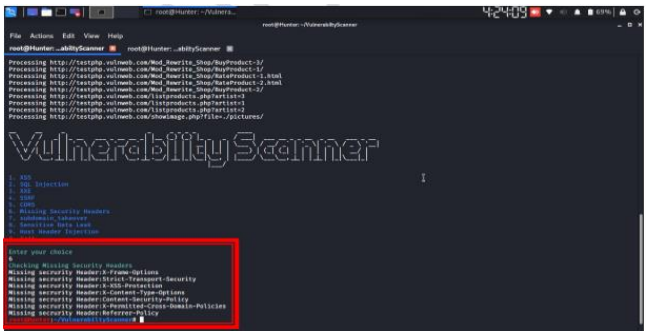

Fig 9:- Running Security Header scripts and results

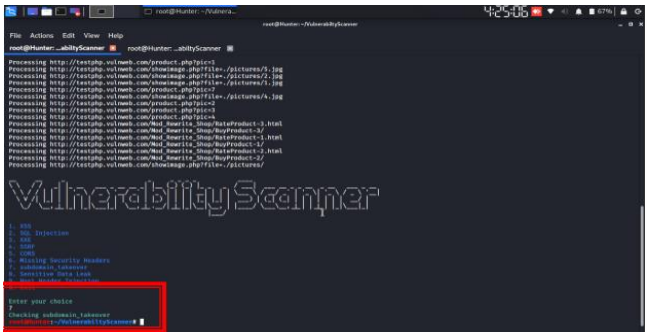

Fig 10:- Detection of subdomain takeover weakness with automatic scripts

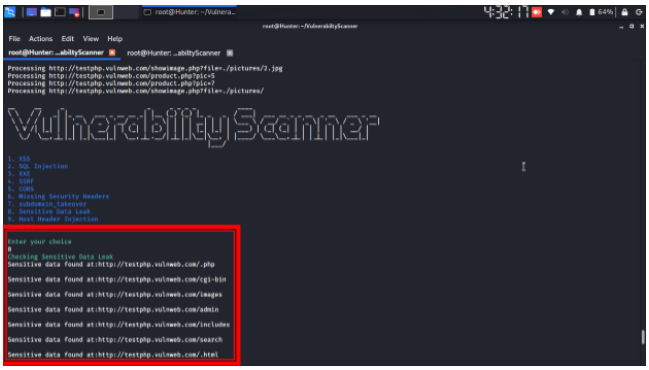

Fig 11:- Running Sensitive data Leak scripts and detection of weakness

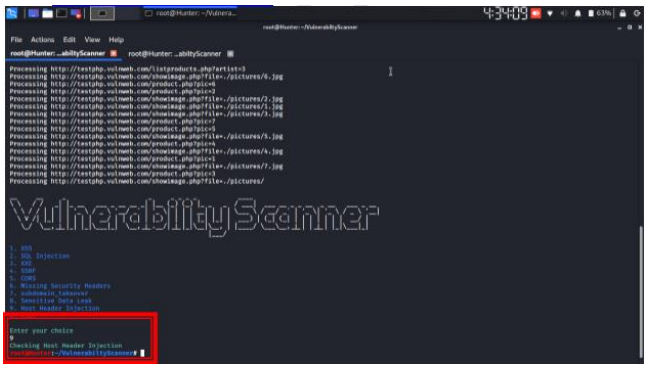

Fig 12:- Running and detecting Host Header Injection scripts

➢ https://demo.testfire.net/

In this environment, manually and automatically detected vulnerability numbers are equal, but automated detection time has been reduced. Weaknesses in both types are : XSS, SQL Injection, Missing Security Headers,

Subdomain Takeover, Sensitive Data Leak, and Host Header injection. Detailed tests on the environment are included in the Annex.Form

- http://zero.webappsecurity.com

The weaknesses detected by automating in this environment are more than those detected manually. Manually detectable weaknesses: XSS, SQL Injection, CORS and Missing Security header.

The weaknesses detected by automating are xxe and Sensitive Data Leak weaknesses in addition to the weaknesses detected manually.

- http://testhtml5.vulnweb.com/

The only manually detected weaknesses are XSS,XXE,CORS and Missing Security headers. Automated scripts provide additional detection of weaknesses such as Host Header Injection and Sensitive Data Leak.

- Typhoon Vulnerable Machine

In this environment, missing Security Headers and Sensitive Data Leak vulnerabilities have been identified with automated scripts.

- Damn Vulnerable Web Application

In this environment, CSS, Sensitive Data Leak and Missing Security Headers and SQL Injection were successfully detected with high output.

- Bwapp

In this environment, as mentioned earlier, OWASP has most of the vulnerabilities in 10, so the number of vulnerabilities detected is higher and are: CORS, Host Header Injection , Sensitive Data Leak, Missing Security Headers, SQL Injection and CSS. Output values are high, as can be seen in the test screenshots in the annex.

C. Dıscussions

In view of the diversity of the environments in which the Test takes place, what we can refer to as openings of urgent importance are openings that result in attacks carried out remotely by unqualified attackers and which result in the complete seizure of the system. For example, in banking applications such as XSS, SQL injection, openness vectors that can lead to customer information disclosure fall into this category. And in particular, these openings have been identified with high output in every environment.

In the information panel, there is a weakness in the image replacement function. Since the loaded file has not passed the required control, it has been observed that a file containing malware can be loaded into the user's profile field.

The detected XSS vulnerabilities occur when applications do not have sufficient input and output controls for outside information, and a malicious user can execute javascript code to steal the session information of the target people, and redirect the browser of the target people at will. The captured victim can perform port scanning, media recording and video recording on the internal network using the browser.

For checking XSS automatic scanning is better than manual scanning. In manual scanning penetration tester has to be gone through all of the input field and then has to be inject the script manually which is time consuming. But automatic scanner do it all in a short period of time.

Manual testing cannot possibly cover everything from A to Z. This is harder due to the obvious reasons such as time and skills, whereas automated tools can do it with a little bit of human intervention.

Automated tools are the perfect fit for testing a target for more of attacks with large number of payloads as it can do it even with a thousand different payloads for one single test. Hence, automated tools can cover the breadth.

A vulnerable SQL Injection vulnerability is a vulnerability that could allow an attacker to append queries to a database sent in the background because the information sent through the application parameters is not properly checked.

V. CONCLUSION AND FUTURE WORKS

It has been observed that the studies and results of different researchers have been supported on the technologies I intend to use when developing automated pentest software. In addition, the existence of many security libraries belonging to the python language that I will use in architecture will have a positive effect on the implementation time of the software. In addition, many tools in the kali operating system should not be ignored. The next phase of my work will be to integrate all these tools with which I perform the test operations. The more output we get from the test results we showed throughout the study and the advantage of all the time we have gained provides the basis for the following statements:

- Cost-conscious information security principals who need to do “less and more” with manual security penetration tests.
- Application security teams that need to provide tiered security and verify results from multiple sources.
- DevOps teams that need App Security to reduce the number of false positives associated with traditional tools.
- "Red teams “that will benefit from a detailed” road map"of current highlighted vulnerabilities in applications.

Following areas can be considered as Future works

- Expending of the Adaptivity in Pentesting to all environments
- Increasing Attack Vector with Machine Learning Methods
- Artificial Intelligence in Adaptive Penetration Testing

- Performance of Module Updating in Penetration Testing Tools
- Automated Attack Vector Planning in Penetration Testing
- How can adaptive pentesting improve by learning and / or capturing its expertise during testing?

ACKNOWLEDGMENT

I would first like to thank my thesis advisor Prof.İbrahim SOGUKPINAR of the Computer Engineering Department at Gebze Technical University. The door to Prof. SOGUKPINAR office was always open whenever I ran into a trouble spot or had a question about my research or writing. He consistently allowed this paper to be my own work, but steered me in the right the direction whenever he thought I needed it.

REFERENCES

[1]. Lee Epling, Brandon Hinkel and Yi Hu , “Penetration Testing in a Box”, InfoSecCD 2015: 6:1-6:4
[2]. Jai Narayan Goel , BM Mehtreb, “Vulnerability Assessment & Penetration Testing as a Cyber Defence Technology”, 3rd International Conference on Recent Trends in Computing 2015 (ICRTC-2015)
[3]. Mariano Ceccato, Riccardo Scandarito, “Static Analysis and Penetration Testing from the Perspective of Maintenance Teams”, 2016
[4]. Markus Huber ,Stewart Kowalski, Marcus Nohlberg and Simon Tjoa, “Towards Automating Social Engineering Using Social Networking Sites”, 2009
[5]. Katharina Krombholz, Heidelinde Hobel,Markus Huber Edgar Weippl , “Advanced social engineering attacks ”, 2014
[6]. G. v. Rossum, "Python Documentation," [Online]. Available: https://docs.python.org/2/library/os.html. [Accessed 15 5 2015].
[7]. G. v. Rossum, "Python," [Online]. Available: https://www.python.org/. [Accessed 16 5 2015].
[8]. Mohamed C. Ghanem, Thomas M. Chen “Reinforcement Learning for Efficient Network Penetration Testing”, 2020
[9]. Qiu, X.; Jia, Q.; Wang, S.; Xia, C.; Shuang, L. Automatic generation algorithm of penetration graph in penetration testing. In Proceedings of the Ninth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, Guangdong, China, 8–10 November 2014.
[10]. Heinl, C. Artificial (intelligent) agents and active cyber defence: Policy implications. In Proceedings of the 6th International Conference On Cyber Conflict (CyCon 2014), Tallinn, Estonia, 3–6 June 2014.
[11]. Sarraute, C.; Buffet, O.; Hoffmann, J. POMDPs Make Better Hackers: Accounting for Uncertainty in Penetration Testing. Available online: https://arxiv.org/abs/1307.8182 (accessed on 20 December 2019).
[12]. Backes, M.; Hoffmann, J.; Kunnemann, R.; Speicher, P.; Steinmetz, M. Simulated Penetration Testing and Mitigation Analysis. arXiv 2017, arXiv:1705.05088. [Google Scholar]
[13]. Almubairik, N.; Wills, G. Automated penetration testing based on a threat model. In Proceedings of the 11th International Conference for Internet Technologies and Secured Transactions, ICITST, Barcelona, Spain, 5–7 December 2016. [Google Scholar]
[14]. Obes, J.; Richarte, G.; Sarraute, C. Attack planning in the real world. arXiv 2013, arXiv:1306.4044. [Google Scholar]
[15]. Backes, M.; Hoffmann, J.; Kunnemann, R.; Speicher, P.; Steinmetz, M. Simulated Penetration Testing and Mitigation Analysis. arXiv 2017, arXiv:1705.05088. [Google Scholar]
[16]. NIST. Computer Security Resource Center—NATIONAL VULNERABILITY DATABASE (CVSS). 2019. Available online: https://nvd.nist.gov (accessed on 18 December 2019).
[17]. MITRE. The MITRE Corporation—Common Vulnerabilities and Exposures (CVE) Database. 2019. Available online: https://cve.mitre.org (accessed on 18 December 2019).