Securing Web Applications Using SNARE and Tanner Honeypot

[1]Kruthika P S, [2]Dr Prabha R,
[1]Mtech Student, [2]Professor
[1]Department of ISE,
[1]Dr Ambedkar Institute of Technology, Bangalore, India

Abstract : ***In today's world there is a constant emergence in web attacks, thus there is huge requirement to secure every web application. One such mechanism is to use Intrusion Detection System, but this mechanism doesn't work well against zero day attacks. This paper proposes a solution to secure web applications against zero day attacks by making use of honeypots along with Intrusion Detection Systems. Honeypots are bait systems which are used to lure attackers and based on their activities, rules are generated and are updated in Intrusion Detection Systems. This method is also cost effective since honeypot is hosted on Orange pi.***

IndexTerms – Honeypots.

I. INTRODUCTION

In today's world web applications are prone to attacks because of one security loop hole or the other. The number of web based attacks is increasing day by day and has cost huge losses. Any web application, be it an online shopping, online booking, bank transaction, digital library or online media streaming, music stores and online education portal applications; process sensitive data like the user's bank account number or credit card details etc. It is very important to secure these websites from intruders. It requires lots of effort to ensure the security of web applications. One such way is to use Intrusion Detection systems.

The Intrusion Detection systems (IDS) are used to monitor the network and block out if the traffic is found to be malicious. Signature Based Intrusion Detection works very well if the signatures have been set up by manually. Thus this may lead to generation of False positives. It also doesn't work well against zero day attacks.

In case of Anomaly-based Intrusion Detection System, there are two phases; and they are training phase and testing phase [1].In the first phase, the behavior of the system when it has normal traffic is observed and a profile is made. In the next phase, the system is kept in production environment and the comparison is done between the profile and the current behavior of the system. And if there is any difference to be found it is considered as an attack and suitable actions are taken. This may seem as a better option but Anomaly-based Intrusion Detection system generates false negatives as even if the traffic is not malicious, and if there is a small deviation when it is compared to the stored profile, it considers it as an attack. The major drawback of Intrusion Detection System is that the rules are not automatically generated. It needs manual updation which consumes time.

Honeypots are systems which are designed to resemble the original systems in production environment and are created with the intent of being compromised [2]. They are made vulnerable on purpose so that when the attacker exploits the system, his activities can be recorded and can be studied later to secure the original systems.

II. RELATED WORK

In [3] Irwan Sembiring presented a technique to detect Distributed Denial of Service attack(DDOS) by using Honeypots. He made use of a low-interaction Honeypot called as 'honeyd' which emulates an operating system and vulnerable ports. He successfully managed to capture a few DDOS attacks from various attackers. In [4] the authors set up a honeypot which could trap the attackers and they also provided solutions to improve the attractiveness of the honeypot. They set up a 'SSH honeypot' under Ubuntu Operating System (OS).

Proxy servers are used by the users to execute any requests indirectly to other network services. These servers can be used to perform malicious attacks. The authors in [5] set up an open proxy honeypot to observe how the intruder takes advantage of it and logged in all his methods to analyze later.

Victor Clincy and Hossain Shahriar proposed an Intrusion Detection System [6] to mitigate injection attacks for web services. They applied a Generic Algorithm to generate signatures for the Intrusion Detection System. This system was effective in generating signatures for new attacks. Hussein Alnabulsi, Md Rafiqul Islam, Quazi Mamun proposed a solution to filter out the SQL injection attack by using Snort Intrusion Detection System [7]. They inflated number of additional Snort rules to already existing ones. The authors evaluated their proposed solution by comparing the methods suggested by them with several techniques which already existed. Thus they demonstrated that their proposed method outperformed other techniques which used the same data set.

III. HONEYPOTS

Honeypots are bait systems that are placed along with production systems to gain the attention of hackers and lure them to attack by making them believe that the system is real. Any attempts which are unauthorized to access the information will be logged by the Honeypot. Usually the Intrusion Detection System used to detect the attacks may generate false positives or negatives, thus these Honeypot systems may be used to resolve this conflict.

A. Snare and Tanner

'Super Next Advanced Reactive honEpot(SNARE)' is a web application based honeypot sensor which attracts all maliciousness from the internet. It has a central decision making authority which is called as tanner. Tanner also emulates few vulnerabilities so that the attacker exploits it well and his activities which are logged in the system can be used for future analysis and also to update the rules of Intrusion Detection System's database. Running tanner service is optional but if used, it provides many benefits. By emulating vulnerabilities it makes

the attacker believe that he is attacking a real production system. Tanner can also be hosted on virtual OS. It is always a good practice to run SNARE and tanner in different host systems.

The function of SNARE is to clone the original web pages. The clone depth can be decided by the user as per his needs, and once it clones; for every subsequent request, it communicates with tanner. Tanner does remote data analysis. It is a classification service which evaluates HTTP requests and providing subsequent responses which are then served to SNARE. Tanner can also adopt and change the responses and they serve dorks. To be on a safer side it is good to run tanner in Docker containers so that the underlying operating system does not get corrupted.

IV. SNORT INTRUSION DETECTION SYSTEM

Snort is an open source Intrusion Detection and Prevention system. It is used to monitor the network. Snort has a rule-set which can be downloaded from their website and along with that it also allows the users to write their own custom rules. Every rule has two parts: They are the rule option and rule header. The rule header has the action of rules (Activation, Dynamic, Alert, Pass, and Log), destination and source address and ports and protocols. The rule option consists of the content of the alert messages, and information regarding the part of each packet that will be later analyzed to decide if or not to carry out the action.

V. ORANGE PI

Orange PI is an open-source single-board computer. It can run Android 4.4, Ubuntu, Debian, and Raspbian Image. It uses the AllWinner H3 SoC, and has 1GB DDR3 SDRAM. Orange Pi is used to host our cloned website.

VI. WEB APPLICATION

A simple student attendance application is used in this proposal where the student logs in with his university sequence number (USN) and views his attendance record. Fig.1and Fig. 2 show the login page and the result of it when proper credentials are given.

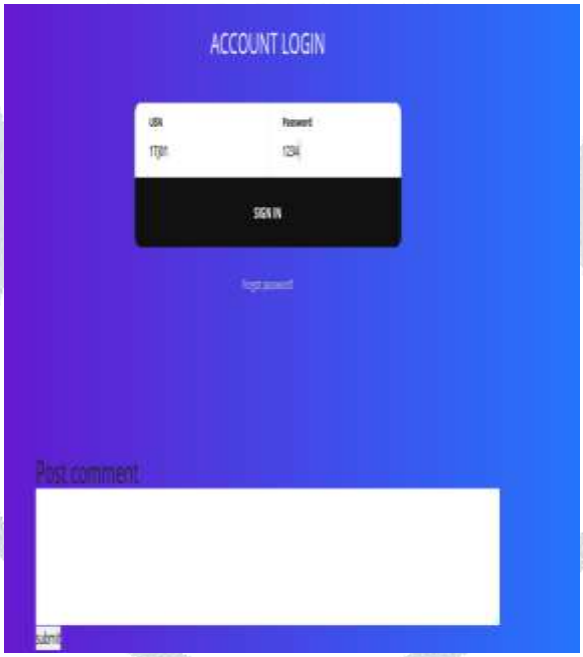

Figure 1: Login Page

Figure 2: Response to proper credentials

VII. TYPES OF ATTACKS

The attacks performed in this proposal are SQL Injection, command Injection and Cross-site Scripting.

A. SQL Injection

In SQL Injection (SQLi), the intruder executes spiteful SQL statements which control the database server of the web applications. Any web application which uses SQL database is vulnerable to this injection. For the SQL Injection attack to take place, user input should be included within SQL statements by the vulnerable website. The intruder then inserts payload which is included as part of SQL query and will run against the database server. For example, the SQL Injection payload can be of setting the password field to 'OR 1=1' which gets converted to SQL query as shown below;

SELECT usn FROM users WHERE username='uname' AND password=' OR 1=1'

Through this attack the intruder can impersonate a particular user and disclose all personal data and can also modify, update, alter or delete some information in the database.

B. Command Injection

Command injection is a type of attack where commands are executed on the host through vulnerable applications. Command injection attack is possible when the application passes the user input which is unsafe to the system. In this particular attack, the system commands are usually executed with the vulnerable application's privileges. Command injection attacks are possible when the input validation is not sufficient. For example, he can insert "whoami" command to the URL.

http://example1.com/?code=system('whoami');

C. Cross-site Scripting

Cross-site Scripting (XSS) is a client-side code injection attack where the intruder executes deceptive scripts into a website which is legitimate. XSS is one of the most epidemic web application vulnerabilities and it occurs when a website makes use of user input which is not validated or encoded within the output it generates. The attacker first injects payload into a web page that is visited by the victim to run the malicious code in his browser. He uses social engineering techniques to achieve this.

VIII. EXISTING SYSTEM

The main drawbacks of the existing system are that the Intrusion Detection Systems do not work well against zero day attacks because, for it to detect any attack, the attack's signature has to be updated in the rule file. The signatures in Intrusion Detection Systems must be manually updated and this consumes time. The existing system makes use of 'glastopf' honeypot. The honeypot used in this proposal is a successor of glastopf which has additional features like a decision maker called as tanner. The existing system extracts the Internet Protocol (IP) address from the log file of the honeypot and blocks the attacker IP address so that the next time when he tries to attack, his traffic gets dropped. This does not work well against IP Spoofing. The intruder can forge an internal IP and try performing his intended attacks. And also in the existing system the attacker is redirected to honeypot when he types the wrong credentials. This can be considered as a disadvantage as there are times when a normal user may type in wrong credentials by mistake.

IX. PROPOSED SYSTEM

As shown in Fig. 3, All the traffic in the network is made to pass through the firewall system and then to Snort Intrusion Detection System .When the requests arrive from the attacker and it matches with the signature of the Intrusion Detection System, it alerts the administrators and drops the request. If the signatures do not match, they get forwarded to the real web server. When the attacker tries to enter wrong credentials, the server checks its database. The database maintains a record of correct passwords and also few incorrect passwords which are most commonly used by attackers. If the credentials match with the correct record they are allowed to access the real web server and if they are matched with incorrect records they are redirected to SNARE. If no match is found with both the records then after the fifth login attempt; the credentials used by him will be added to the database to incorrect records and the attacker is redirected to the cloned website without his knowledge. There are times when the normal user may forget his password and types the credentials wrong, thus to prevent the normal user from being redirected, the server provides four login attempts. SNARE communicates with tanner. Tanner acts as the backend and emulates all the vulnerabilities and tries to respond as per to attacker's expectations.

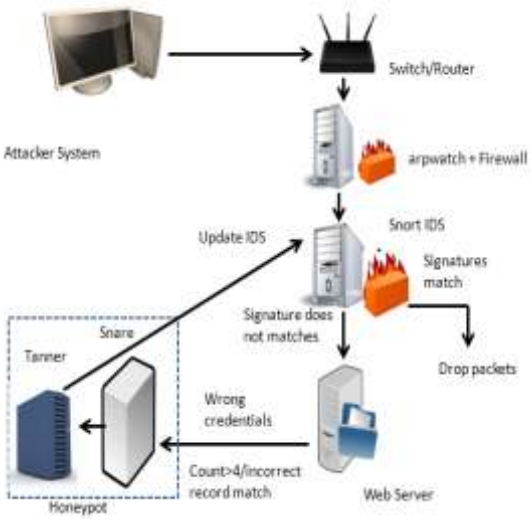

Figure 3: The Proposed Architecture

All these activities are saved in the log files of tanner and the paths accessed by the attacker is shown in SNARE log. Tanner log file is saved in opt/tanner/tanner.log. Here if the attacks are detected, then signatures are generated and are updated automatically in the Intrusion Detection System and the attacker IP is blacklisted, thus if any requests from outsider matches these signatures or the blacklisted IP then the requests are dropped. The attacker may spoof his IP and may try to perform new attacks and thus the log files of the firewall and the switch are monitored by a script, to check for multiple duplicate ACK packets and also if there are unusually large number of SYN packets to which there is no response then it is a clue that spoofing has appeared and those IP addresses are blocked by the iptable. Also if there is Internal Local Area Netword(LAN) IP address capturing data for an external interface in the log entry then it means that the attacker has spoofed internal IP of the LAN and this attempt is blocked. The firewall system also has utility called 'arpwatch' which keeps track of all the IP and Media Access Control Address (MAC) and if it senses any changes then it alerts the administrators about the possible intrusion. The flow chart to this proposed system is shown in Fig. 4.

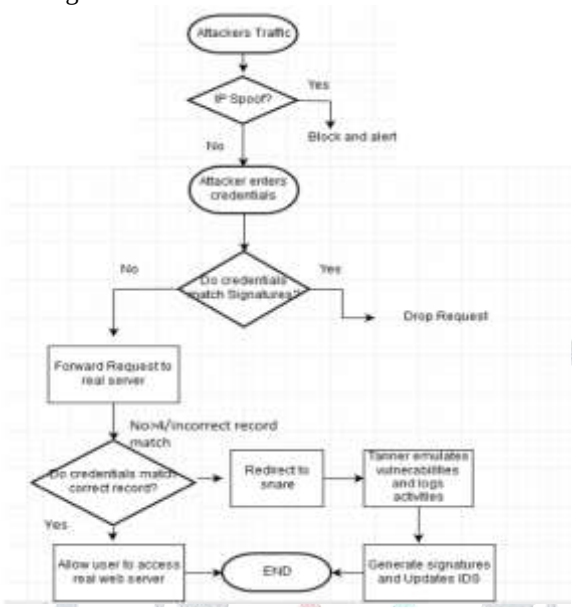

Figure 4: Flowchart

X. EXPIREMENTAL SETUP

This section has the implementation of the proposed work. We have made a student attendance web application which requires student's USN number and password as his credentials. The script to check the credentials is shown in Fig. 5.

Figure 5: Script to Check Credentials and Redirect

The above script checks the credentials typed by the user by comparing it with the credentials saved in the database. If there is a match with the correct record, then the user is allowed to access the real server. If the user types the wrong credentials for more than four times, he is redirected to the cloned website and the credentials used by him is stored in the incorrect record. If the credentials typed by the user matches with the incorrect record, then he is automatically redirected to SNARE. The page cloned by SNARE is shown in Fig. 6. SNARE has been hosted on Orange Pi. This is shown in Fig. 7.

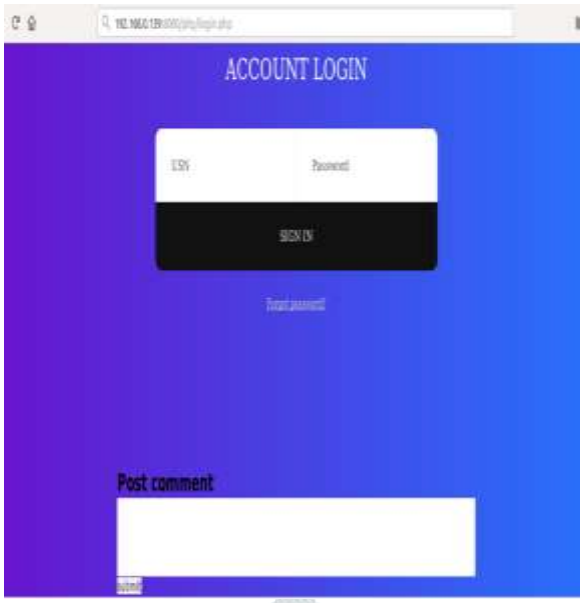

Figure 6: Cloned web page

The result of Command Injection attack emulated by tanner is shown in Fig. 8.This particular response is for 'cat /etc/passwd' command. Similarly when the honeypot was subjected to XSS attack the response given is shown in Fig. 9. Here, attempt to capture cookies were made and was successful. The code to capture it was posted in comment section. Attempt for SQL injection was also successful by giving 'Or'='1', but a dummy database had to be created in tanner.

Figure 7: Orange Pi

Apache/2.4.18 (Ubuntu) Server at 192.168.0.105 Port 80
root:x:0:0:root:/root:/bin/sh daemon:x:1:1:daemon:/usr/sbin:/bin/false bin:x:2:2:bin:/bin:/bin/false sys:x:3:3:sys:/dev:/bin/false sync:x:4:100:sync:/bin:/bin/sync
mail:x:8:8:mail:/var/spool/mail:/bin/false www-data:x:33:33:www-data:/var/www:/bin/false operator:x:37:37:Operator:/var:/bin/false
nobody:x:65534:65534:nobody:/home:/bin/false

Figure 8: Command Injection

All the paths accessed by the attacker can be seen through SNARE's log and this is shown in Fig. 10. This was captured during command injection attack.

Figure 9: XSS Attack

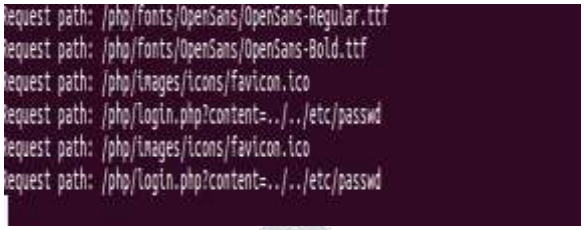

Figure 10: SNARE Log

The log file of the tanner has all the information about attacker including his IP address. This is shown in Fig. 11. Using this log file, signatures are generated and IDS is updated.

Figure 11: Tanner Log

From these log files, the required fields to make rules are extracted by using a script called splitall.py. This script splits the required fields and puts it into a text file which is then sent to the snort server. Also the IP address of the attacker is extracted from the file and is blacklisted in the snort server. When snort receives this file, it opens it and matches the rules in this file to the rules in its rule file. If there are no similar rules, then it will save the new rule in its rules file. The rule generated for SQL injection and XSS attack and command injection is shown in Fig. 12.

Figure 12: Snort Rule

The generated rules are updated in the directory by using the script shown in Fig. 13.

```
#!usr.bin/env python
with open("snort.conf","a") as myfile:
        for index in range(1,4):
myfile.write('include /etc/snort/rules/rules{}/rules\n'.format(index))
```

Figure 13: Script to Update Rules

A script is used to generate the rule automatically. This script is shown in Fig. 14.

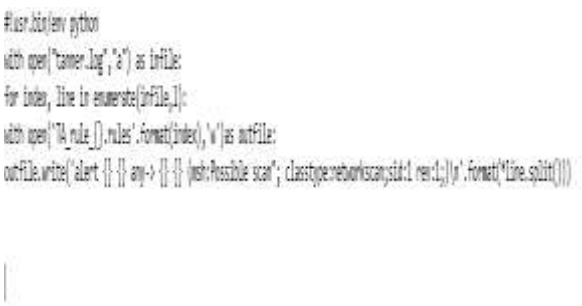

Figure 14: Script to Generate Rules

The next time when the traffic is passed through snort and it matches with the rules, it is dropped and if the IP is blacklisted then it is blocked.

XI. CONCLUSION

The proposed system was developed not only to secure a web application, but also to configure both the honeypot and Rules of Snort. The system was able to generate new Intrusion Detection System rules against attacks performed and blacklist IP address of the attacker with the help of honeypot. It also blocked out Spoofed IP addresses.

REFERENCES

[1] S. Khan and D. Motwani, "Implementation of IDS for web application attack using evolutionary algorithm," 2017 International Conference on Intelligent Computing and Control (I2C2), Coimbatore, 2017, pp. 1-5.

[2] D. K. Rahmatullah, S. M. Nasution and F. Azmi, "Implementation of low interaction web server honeypot using cubieboard," *2016* International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC), Bandung, 2016, pp. 127-131.

[3] I. Sembiring, "Implementation of honeypot to detect and prevent distributed denial of service attack," 2016 3rd International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE), Semarang, 2016, pp. 345-350.

[4] A. A. Egupov, S. V. Zareshin, I. M. Yadikin and D. S. Silnov, "Development and implementation of a Honeypot-trap," 2017 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), St. Petersburg, 2017, pp. 382-385.

[5] R. E. Mushtakov, D. S. Silnov, O. V. Tarakanov and V. A. Bukharov, "Investigation of modern attacks using proxy honeypot," 2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), Moscow, 2018, pp. 86-89.

[6] Varsha, Ravichandra Mouli & Kp, Jevitha. (2016), "Web Services Attacks and Security- A Systematic Literature Review. Procedia Computer Science," 93. 870-877. 10.1016/j.procs.2016.07.265.

[7] Alnabulsi, Hussein & Islam, Md Rafiqul & Mamun, Quazi. (2014), "Detecting SQL injection attacks using SNORT IDS," Asia-Pacific World Congress on Computer Science and Engineering,