The investigation on cowrie honeypot logs in establishing rule signature snort

E Satria¹*, T P S Huda², M Iqbal², F W Sarjana²

¹South Aceh Polytechnic, Indonesia
²Informatics Department, Syiah Kuala University, Indonesia

Email: eri89rz@gmail.com

Abstract. The attack of brute force is still one of the popular attacks used to hack into your account unauthorized by a computer system. Brute force is also the most crucial attack and has a high risk of the system being taken over. Investigating brute force attacks is useful for building strong computer network defense systems. In this study, Snort acts as an intrusion prevention system and Cowrie Honeypot as a tool to investigate anomalous behavior that occurs when a brute force attack happened. The aim of this research is to improve Snort's rule signature performance from brute force attacks by relying on the results of the Cowrie Honeypot log investigation. The results obtained, namely Snort rule signature successfully improved detection capabilities with performance in matching the same packet only requires a short processing time, respectively: 3.5 microsecs in Hydra attacks, 3.8 microsecs in Medusa attacks and 2.3 microsecs in Ncrack attacks.

1. Introduction

Brute force attack, one of the very dangerous threats to a computer network system. The impact that may result from brute force attack a target computer system include piracy, computer system resources exploitation and even steal sensitive data stored in computer systems. But the brute force attack method evolved from traditional to modern ways along with the rapid development of the world of computer testing. Various variations of brute force attack methods are designed by hackers to avoid checks from computer security systems. The most well-known method among hackers is the word list method, which is the method of guessing a valid user account password from a computer system based on a file containing a combination of passwords to be tried for access [1]. Hackers generally directly target administrative accounts or super accounts (root) of the target computer system instead of targeting ordinary user accounts [2].

Secure Shell (SSH), remote access service encrypted path to a computer or server system [3][4]. Today's computer or server systems provide remote control SSH services to make it easier for administrators to complete their work without having to visit the physical system directly. The SSH service uses the authentication method to verify the authenticity of the account before the user (administrator) enters the system, username and password are conditions that must be owned by an administrator to access computer system resources that are controlled [4].
Computer systems or servers connected to public network systems (the Internet) are more easily controlled remotely than computer systems or servers that are on a local network. Connecting to a public network system means that the computer system can be connected to anyone, this will certainly have the potential to be hijacked if the computer does not have access rights and strict network security in the computer network system environment. Brute force attacks target computer systems or servers running SSH services on them on public networks [5].

The password creation policy is set to mitigate hackers breaking into brute force attacks, but it is not uncommon for computer or server systems to be given passwords in weak combinations instead of using strong combinations to make it easier to remember. User accounts given a weak password combination will be easily hijacked by hackers with brute force attacks [6]. Installation of firewalls also be the right choice to block brute force attack, but a firewall works with IP address detection system and Port makes a perfect weakness for hackers brute force attacks passed through the firewall check by anticipating the IP address and port permitted by the firewall. Because of the limitations of the firewall in detecting computer network security threats so that gave birth to the intrusion detection system method that is able to carry out inspections up to the payload level of network packets [7][8].

Intrusional Prevention system comes after the creation of intrusion detection system, in contrast to intrusion detection system that will only notify administrators in the form of warnings of potential threats, as the development of intrusion prevention system is designed to block the detected threats [8][9]. How the Intrusion prevention system works in detecting threats to attack a computer network system refers to matching the signature defined by the network administrator.

The process of making signatures in the intrusion prevention system involves an in-depth investigation of threats from hackers by learning the uniqueness of the various attacks carried out. Especially with threats from brute force attacks, network administrators learn how many attempts to enter the system are done per second.

Cowrie Honeypot is a fake SSH service system that is specially emulated targeting brute force attacks from hackers to get caught in fake computer systems or servers [10]. The Cowrie Honeypot system acts as a trap for the barrier to be trapped in a computer system environment that is monitored by producing log files from a series of activities that are done by hackers in the Honeypot trap system [11][12]. This log will later be useful for investigations in revealing hacking behavior patterns as well as calculations of the time of performance in trying to break into verification of a valid SSH authentication account.

In this study, we propose how to build a rule signature Snort in the face of brute force attacks based on the results of the log Cowrie Honeypot investigation. In this research we will describe each phase in several sessions namely in session 2 we explore the literature related to this research, session 3 discusses the research methodology that we use to obtain research results, session 4 implements and discusses the test results obtained and the final session we draw conclusions from a series of tests that have been carried out.

2. Related Literature

2.1. Cowrie honeypot
According to Michel Oosterhof, Cowrie is a honeypot that emulates SSH and Telnet services with medium to high interaction to record brute force attacks and shell operations carried out by hackers [10]. Log Cowrie honeypot collected to study the pattern or hacker suspicious behavior or new discoveries related to how brute force attacks launched by hackers, so make the investigators more renewable explore information in a brute force attack that occurred. The output format of the log cowrie JSON with JSON log cowrie writing structure can be seen in Figure 1.
Snort is a program of network-based intrusion detection system that can operate as an intrusion prevention system and rule-based in its detection [13]. Snort has 5 series of components that have specific functions and tasks to process packet flow in network traffic [14] while the Snort component framework is shown in Figure 2.

Snort relies on the rule in its expansion of packet flow in network traffic. Rule plays an important role in the success of the Snort program in detecting threat attacks in network traffic. Therefore, a strong rule will make the detection system to be reliable in dealing with various threats or attacks in the network system. The structure of the signature sign nort consists of 2 parts namely the rule header and the rule options, which in the rule header section contains several parameters, namely: rule action, protocol, source and destination IP address, netmask, source and destination port. There are three action actions available namely alert, log and pass. Additionally, when Snort is run in the intrusion prevention system mode, the additional options include drop, reject and sdrop. Whereas rule options are an important part of Snort detection, where there are four main categories of rule options, namely: general, payload, non-payload and post-detection. In the detection of brute force attacks, the rule options used are post-detection on the parameter “detection filter” [13]. Where the important role in detecting failed logins is in the options section of the “detection_filter” parameter, namely the “count” and “seconds” options.

Snort provides the rule profiling feature in its program which aims to measure the performance of the rules made [13]. Activating this feature is done in the Snort configuration file. Snort will automatically provide statistics about the rule analyzed its performance in a log file with the format as shown in Figure 3.

Several previous studies that have been carried out relating to this research or involve Cowrie Honeypot, as in the script [15] Tripathi and Kumar, implementing Snort, Cowrie Honeypot and Tshark packet analyzer on Raspberry Pi devices in securing Internet based home computer network systems of Internet of Things based. In testing the system that was built, simulated ICMP attacks, Brute Force and Port Scanning. The test results obtained indicate that Snort, Cowrie Honeypot and
Tshark managed to record the attacks that occurred and the three programs successfully run in the Rasberry Pi device environment.

The analysis of the Honeypot fraud technique in the manuscript [16] Qassrawi and Hongli shows that the fraud technique built on the Honeypot has different variations with the same goal of making the system like the original system. With this Honeypot fraud technique makes it easier to investigate threats or attacks on computer networks and increase the ability of signature intrusion detection.

The research that conducted by Lingenfelter et. al., analyzed variations of IoT Botnet using the high interaction medium honeypot, Cowrie Honeypot. The results of the implementation of the Cowrie Honeypot showed that the Cowrie Honeypot managed to capture the Mirai worm, a botnet that attacks an IoT device using brute force attack techniques [17].

3. Methodology

In this study, we propose a method for making Snort rule signatures, especially on the threat of brute force attacks, based on the results of investigations of the Cowrie Honeypot log records. The accuracy and success in establishing the detection of Snort rule signatures from brute force threats is greatly influenced by two main parameters namely: “count” and “seconds”. Therefore, by relying on the results of investigating the events of the “timestamp” brute force attack recorded by Cowrie Honeypot it will be easier to determine the value of these two parameters. In this study, brute force testing is carried out in a virtual system environment that has been designed as shown in Figure 4.

![Figure 4. The physical topology of system testing.](image)

The physical topology test shown in Figure 1 explains that there are two subnets in area B, where the two subnets are not interconnected between each other configured in the routing table on router B, so these two subnets cannot communicate with each other, while the subnets in the area A can communicate to two subnets in area B. In this study, there are two test scenarios arranged to simulate the brute force attack that occurs in accordance with the original environment. The explanation of the two test scenarios is explained in full as follows:

The first scenario, in this scenario the dropper computer will attack the SSH server service which is simulated by Cowrie Honeypot in the subnet area 192.168.12.0/24. Hackers are circumvented as if they were targeting the original SSH service system instead of the hacker being trapped in a fake SSH service system. In this study, brute force was carried out using 3 popular open source brute force programs in the Internet, namely Hydra, Medusa and Ncrack. The third program will be performed in-depth investigation of many packets that are sent right per second to the target of the attack based on the results of logging Cowrie Honeypot. In summary the first test scenario and its investigation are shown in Figure 5.
The second scenario, in this scenario begins by building a Snort rule signature based on the results of an investigation conducted in the first stage. The built-in signature will be tested in the physical topology environment as shown in Figure 1. Hackers launch a brute force attack using 3 previous brute force programs namely Hydra, Medusa and Ncrack against the original SSH server service that was in the 192.168.13.0/24 subnet area. Unlike the first scenario, in this second scenario the packet sent by the hacker will be checked first by Snort. The packet snort checking process involves matching packets with the collection of rule rule databases it has. If the packet does not match the specified rule signature, the packet will be allowed to the target, whereas if the packet is found to match the existing signature, the packet will be blocked to the target. In this second scenario data is also obtained regarding the performance of Snort's rule signature that has been built based on the recording of Snort's “rule profiling” log. In brief, this second scenario is illustrated in Figure 6.

4. Results and discussion

4.1. Cowrie honeypot
The log investigation method is done by finding the average packet per second of each brute force program based on the Cowrie Honeypot log records produced for the brute force attack that has been described in the first scenario. The results of the investigation log Cowrie are shown in Table 1, it was revealed that the pattern of delivery of packets per second, brute force every program tested in this study is different, where the program hydra send packet attacks per second as many as 16 packets, the program Medusa average send 1 packets per second and the Ncrack program averages 111 packets per second.
| Tool of brute force | Time of attack (hour:minute:second) | Packets on average per second (packet) |
|--------------------|-----------------------------------|---------------------------------------|
| Hydra              | Start 12:08:18, Finish 12:34:51   | 16                                    |
| Medusa             | Start 12:56:53, Finish 15:45:05   | 1                                     |
| Ncrack             | Start 15:49:12, Finish 15:50:42   | 111                                   |

4.2. Establishing models
As was mentioned at the beginning of the methodology section, that the process of establishing Snort's rule signature is driven by the results of the Cowrie Honeypot log investigation. Based on the details shown in Table 1 before, that the investigation results from the calculation of the number of packets per second recorded in the Cowrie Honeypot log, then the rule signature brute force was built by filling the average packet per second obtained from the investigation into the Snort rule brute force parameter namely “count” and “seconds”. Each rule signature in Table 2 represents one brute force program that will be inspected for its attack. Inspection of packets from brute force attacks, Snort rule signatures are built with the parameter “count” of successive values of 3 rules namely 16, 1 and 111 with the value “seconds” given 1. It means that each packet from brute force attacks that pass through the Snort system will be checked the number of packets per second it receives. If a packet that enters the Snort system matches one of the 3 declared rules, the Snort system will block the packet that has passed the threshold of the packet that may pass per second.

4.3. Testing
Rule testing is done to test how high the rule performance is in checking packets that are launched on brute force attacks. The results of rule profiling analysis on the rule signature test that has been built from Table 2 shows that the three rule signatures work productively, where the number of matching packets is almost half of the packets examined. The time needed for the rule in evaluating packet matching looks a little, namely the average checking of the three rules is 1.9 microseconds, 2.4 microseconds and 1.3 microseconds, while the time needed to evaluate the corresponding packet is 3.5 microseconds, 3.8 microseconds and 2.3 microseconds. It can be seen that the average time required for evaluating a suitable packet is higher than the average checking time because Snort needs more time to assign a packet to the rule parameters created whether the packet matches the rule or not. However, unlike the case when the packet is declared incompatible with the rule, the average time needed to evaluate a packet that does not match the lower the consecutive 0.4 microseconds, 1.2 microseconds and 0.3 microseconds, this is because the packet has been considered by Snort does not in accordance with the criteria specified in the database of the declared signature brute force rule.

| SID    | Checks (packet) | Matches (packet) | Avg/Check (microsecs) | Avg/Match (microsecs) | Avg/Nonmatch (microsecs) |
|--------|-----------------|------------------|-----------------------|-----------------------|--------------------------|
| 1000001| 19899           | 9933             | 1.9                   | 3.5                   | 0.4                      |
| 1000002| 17              | 8                | 2.4                   | 3.8                   | 1.2                      |
| 1000003| 60684           | 30264            | 1.3                   | 2.3                   | 0.3                      |

5. Conclusion
In this study, the proposal to make rule brute force based on log cowrie investigation to find the right value in filling the parameters “count” and “seconds” in the Snort brute force rule signature format has been successfully carried out. This is explained by the results of rule profiling analysis showing
half the packets examined successfully matched the rule signature created and the time required for matching the same packet is still low and matching packets that are not the same is also very low. Thus, it can be concluded that the rule is built according to investigations by the cowrie logs very productive and can improve Snort intrusion prevention system especially in inspecting packet on brute force attacks.

References
[1] Bošnjak L, Sreš J and Brumen B 2018 Brute-force and dictionary attack on hashed real-world passwords 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO) pp 1161–6
[2] Grasdal M, Hunter L E, Cross M, Hunter L, Shinder D L and Shinder T W 2003 Chapter 2 - MCSE 70-293: Planning Server Roles and Server Security MCSE (Exam 70-293) Study Guide ed M Grasdal, L E Hunter, M Cross, L Hunter, D L Shinder and T W Shinder (Rockland: Syngress) pp 53–146
[3] Kälkäinen J 2018 Collection and analysis of malicious SSH traffic in Oulu University network (Oulu: Oulu University Library)
[4] Barrett D J and Silverman R E 2001 SSH, The Secure Shell: The Definitive Guide (USA: O’Reilly & Associates, Inc.)
[5] Owens J and Matthews J 2007 A Study of Passwords and Methods Used in Brute-Force SSH Attacks
[6] Sentanoe S, Taubmann B and Reiser H P 2018 Sarracenia: Enhancing the Performance and Stealthiness of SSH Honeypots Using Virtual Machine Introspection Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) vol 11252 LNCS (Springer Verlag) pp 255–71
[7] Roberto D Pietro, V M L and Sushil J 2008 Intrusion Detection Systems (Springer {US})
[8] Scarfone K, Mell P and Mell P 2007 Guide to Intrusion Detection and Prevention Systems (IDPS) Recommendations of the National Institute of Standards and Technology (USA)
[9] Scarfone K and Mell P 2010 Intrusion Detection and Prevention Systems Handbook of Information and Communication Security (Springer Berlin Heidelberg) pp 177–92
[10] Michel O 2020 Cowrie Readthedocs io en latest
[11] Provos N 2004 A Virtual Honeypot Framework
[12] Spitzner L 2003 Honeypots: tracking hackers (Addison-Wesley)
[13] Snort Project Team 2020 SNORT Users Manual 2.9.16
[14] Charlie S, Hayes B and Wolfe P 2004 Snort™ For Dummies (Hoboken: Wiley Publishing, Inc)
[15] Tripathi S 2018 Raspberry Pi as an Intrusion Detection System, a Honeypot and a Packet Analyzer 2018 Int. Conf. Comput. Tech. Electron. Mech. Syst. 80–5
[16] Qassrawi M T and Hongli Z 2010 Deception Methodology in Virtual Honeypots 2010 Second International Conference on Networks Security, Wireless Communications and Trusted Computing vol 2 pp 462–7
[17] Lingenfelter B, Vakilinia I and Sengupta S 2020 Analyzing Variation Among IoT Botnets Using Medium Interaction Honeypots 2020 10th Annual Computing and Communication Workshop and Conference (CCWC) pp 761–7