Hindawi
Journal of Electrical and Computer Engineering
Volume 2019, Article ID 4568368, 13 pages
https://doi.org/10.1155/2019/4568368

Research Article

Investigating Brute Force Attack Patterns in IoT Network

Deris Stiawan,[1] **Mohd. Yazid Idris**,[2] **Reza Firsandaya Malik**,[1] **Siti Nurmaini**,[1] **Nizar Alsharif**,[3] **and Rahmat Budiarto**[3]

[1]*Computer Engineering Department, Faculty of Computer Science, Universitas Sriwijaya, Palembang, Indonesia*
[2]*School of Computing, Faculty of Engineering, Universiti Teknologi Malaysia, Johor Bahru, Malaysia*
[3]*College of Computer Science & Information Technology, Albaha University, Albaha, Saudi Arabia*

Correspondence should be addressed to Deris Stiawan; deris@unsri.ac.id

Received 19 November 2018; Revised 4 February 2019; Accepted 27 February 2019; Published 1 April 2019

Academic Editor: Maurizio Martina

Internet of Things (IoT) devices may transfer data to the gateway/application server through File Transfer Protocol (FTP) transaction. Unfortunately, in terms of security, the FTP server at a gateway or data sink very often is improperly set up. At the same time, password matching/theft holding is among the popular attacks as the intruders attack the IoT network. Thus, this paper attempts to provide an insight of this type of attack with the main aim of coming up with attack patterns that may help the IoT system administrator to analyze any similar attacks. This paper investigates brute force attack (BFA) on the FTP server of the IoT network by using a time-sensitive statistical relationship approach and visualizing the attack patterns that identify its configurations. The investigation focuses on attacks launched from the internal network, due to the assumption that the IoT network has already installed a firewall. An insider/internal attack launched from an internal network endangers more the entire IoT security system. The experiments use the IoT network testbed that mimic the internal attack scenario with three major goals: (i) to provide a topological description on how an insider attack occurs; (ii) to achieve attack pattern extraction from raw sniffed data; and (iii) to establish attack pattern identification as a parameter to visualize real-time attacks. Experimental results validate the investigation.

1. Introduction

Earlier security protocols should be pertinent to IoT to assure basic security services including authentication, confidentiality, integrity, nonrepudiation, access control, and availability. The reason is that IoT is as an extension of the classical Internet framework and technology. Nevertheless, the IoT network is constrained by several new factors such as huge numbers of devices and objects that may interact together in a complex manner, using different security techniques. Moreover, the evolution from limited access and closed networks to open ones increased the requirement for security alerts to protect all the devices in an IoT network from intrusions [1].

End nodes (sensors/devices) are attached to IoT networks and communicate with a data/application server through a gateway. Collected data are usually transmitted from the gateway to a data/application server using the FTP protocol. Unfortunately, in terms of security, the FTP server at a gateway or data sink very often is improperly set up. At the same time, password matching/theft holding is among the popular attacks as the intruders attack the IoT network.

The novelty of this paper is the use of a time-sensitive statistical relationship approach and visualizing the attack patterns that identify its configurations in brute force attack (BFA) on an FTP service investigation. The investigation focuses on attacks launched from the internal network, due to the assumption that the IoT network has already installed a firewall. The investigation provides a new insight of this type of attack with the main aim of coming up with attack patterns visualization that may help the IoT system administrator to analyze easily any similar attacks.

An insider/internal attack launched from an internal network endangers more the entire IoT security system. Thus, securing FTP connections from botnet attacks on IoT networks is crucial. To understand how to protect against

such attacks, it is best to examine the attack from the attacker's perspective with regard to the used methods, desired goals, and the manner of launching the attacks. The authors undertake experiments to investigate several attack types, in particular in intrusions such as (i) *probes* that aim to obtain detailed information and (ii) *brute force attacks* (BFA) geared towards guessing passwords and/or gaining privileged access.

Several malware variants, as discussed by [2, 3], infect hardware, software, and networks and, in some cases, can also infiltrate via *spam, phishing, and drive-by download*.

This paper describes *brute force* malware attacks on the FTP server of an IoT network to gain escalating privileged access in the IoT environment. Steiner [4] identified weaknesses in FTP service provision and strengthened by the results of research in [5]. Meanwhile, Joshi et al. [6] clearly described a BFA to break FTP's encrypted password. Nevertheless, FTP remains a major alternative for the provision of data transfer services despite its vulnerability, due to the use of plain text authentication procedure.

Having done experiments and investigation, the authors describe the following relevant matters:

(i) How to extract important features of data packages related to potential attack packages

(ii) How to detect BFAs on FTP services on IoT networks

(iii) How to visualize FTP attacks by using a time-sensitive statistical relationship

(iv) How to display patterns of known attacks by computing the number of alerts.

The paper is divided and arranged as follows: Section 2 discusses literature review of related works. The research methodology is in Section 3 that consists of the scenario, the stages, and the groove in the investigation including scanning, brute force, and gaining privileges are considered. Presentations and discussions of the results are described in Section 4. Section 5 provides a conclusion.

2. Related Works

Previous researches [7, 8] have tested the penetration of Internet service operating systems to analyze vulnerability and exploitable security lapses. Their results can be summarized as follows: (i) an advanced system is affected by many factors including its kernel engine, active services, degree of expired service engines, and time period for updates; (ii) each attack contains a unique payload that serves as a flag attack pattern; and (iii) all operating systems tested (FreeBSD, Linux, and Windows servers) have levels of vulnerability and were given risk ratings. Similar researches are also being carried out by Austin et al. [9], and Broucek and Turner [10] undertook similar investigations in preparation for an offensive cyberwar.

Currently, three methods are commonly used to elicit passwords: *brute force*, *dictionary*, and *Hybrid* attacks. The present work examines BFAs that are used to find the combination of password to access FTP services. BFA purposes are to break/decrypt secret codes by trying all possible keystrokes for which the probability of success is highly dependent on the level of difficulty for the password combination.

Venter's benchmark works [11, 12] presented possibilities for breaking password codes both offline and online and have been referred to by many researchers. The researches present about some possibilities which can be done to break the password in both ways: offline and online. Moreover, Helkala et al. [13] reinforced Venter's research by using small instruments that yielded high impacts. In addition, Pilli et al. [14] and Vykopal [15] described other aspects of BFAs regarding taxonomy, multiple approaches, and distributions.

In essence, BFAs force the inclusion of characters that hazard guesses password and can be done remotely by an attacker machine. In brief, BFA is a password experiment that uses a mix of possible ASCII characters in isolation or in combinations.

Generally, BFAs are divided into two attack classes, insider or outsider, as reported by [3, 16, 17]. Both of these attacks are illustrated in Figure 1. Meanwhile, online password hacking has been described in [18], and offline hacking research has demonstrated that a number of characters and password combinations greatly influence the length of time required for hacking [13, 15]. Overall, all cited investigators stressed that BFAs have real-time capability to actually deduce valid passwords on FTP servers.

According to Jang-Jaccard and Nepal [3] and Nithiyanandam et al. [19], several types of potential internal/insider attacks are possible. These include the man in the middle attack, bring your own device (BYOD) attack, malware, device/physical data theft, and sabotage. The observations allowed to characterize and conclude the following:

(a) The insider attack is usually perceived as a valid user of the institution/company

(b) The insider attack has limited access to some services without additional coatings on different service packages and also differs from inbound packages from outside the network that are tightly scrutinized by filters with multiple DMZ services

(c) An insider attack on IoT is a multiform that poses various problems related to malicious and accidental security incidents stemming from employees and outsources

(d) Since the attacker is inside, they have detailed knowledge of technical matters such as the network's backbone, IP address allocations, the virtual local area network (VLAN), the service clustering application, and IT staff members who monitor the network

Figure 2 demonstrates brute force attack approaches and methods and visualize patterns that describe brute force attacks. Some attack patterns were produced by using graphinfo's time-sensitive approach to statistical relationships, as discussed by Saoddodin and Ghorbani [20]. Other patterns were generated and simulated with the MIT

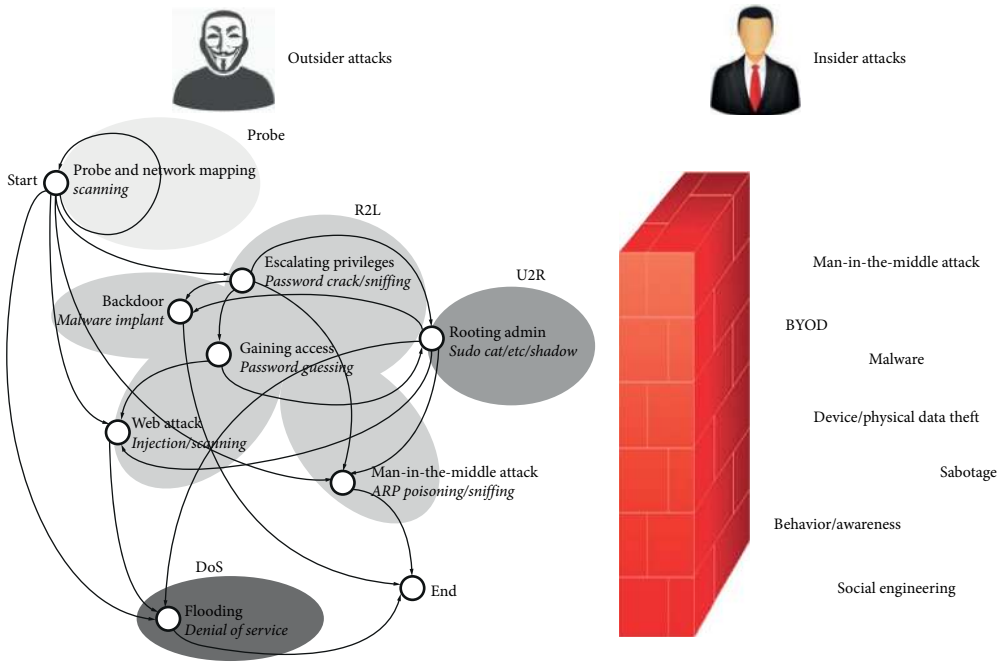


Figure 1: BFA illustration.

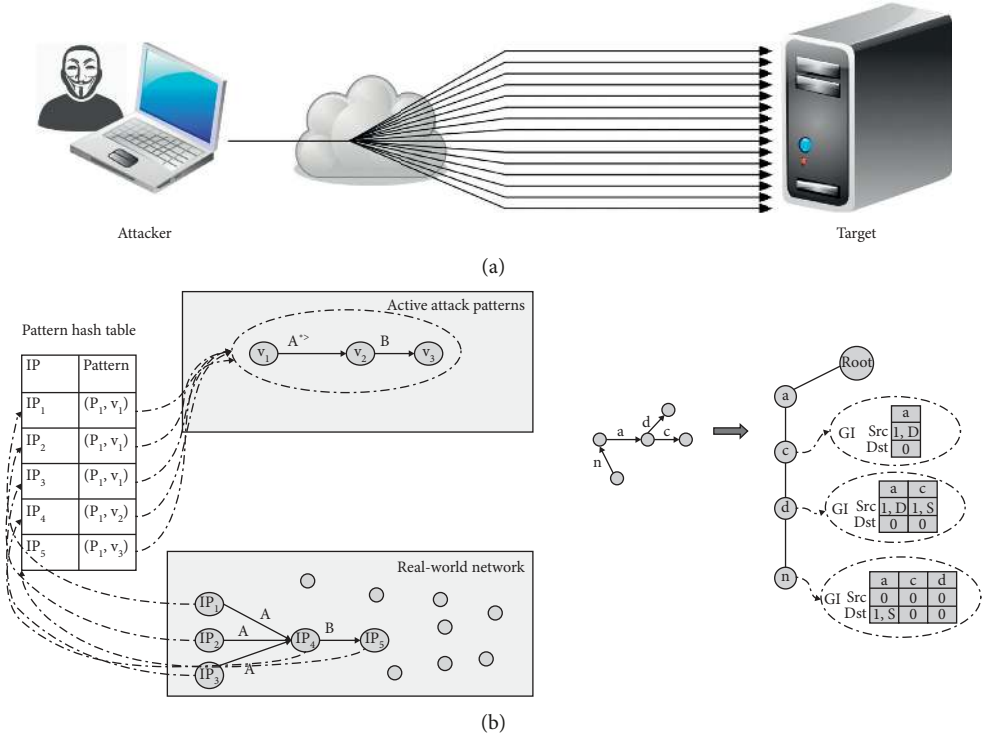


Figure 2: (a) Brute force attacks. (b) Attack pattern based on the correlation approach.

DARPA dataset. Distribution values for pattern outcomes obtained during simulated attacks matched results from extracted package data.

Attack detection tools are alternately used by many researchers including Snort detection software, which has a detection engine that produces alerts [21–24]. Its ability expressly relies on available rules (in/etc/snort/rules/) that effectively recognize attacks. Snort also compiles a pcap file of raw data derived from its sniffing process. Both abilities have made Snort a major tool and referenced instrument in the field of systems security researches.

The Snort engine is also used to report "front-end" attacks. Its engine identifies malicious attempts during real-time traffic based on well-known attack algorithms. When

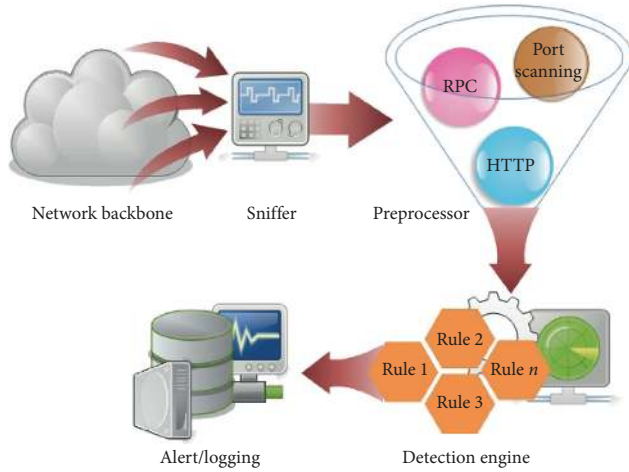


Figure 3: General architecture of Snort.

malicious activity occurs, Snort generates hundreds of events to warn that an activity has been identified as a potential threat. Snort also uses a variety of methods that categorize and log intrusions. Best of all, Snort alerts contain copious data such as IP addresses that identify source-destination, port addresses (source and destination), attack names, alert priority, TTL, and packet length. Snort Version 2.8.5 (Build 121) uses 65 rules that identify and detect threats from pcap files and then used to produce numerous alerts in the log directory ("/var/log/snort"). The number of rows generated during reiterative runs of the same data is simplified by initialization based on signature-id and priority. Each alert consists of a signature-id, priority, src_ip, src_port, dst_ip, dst_port, timestamp, TTL, ToS, IP_Len, and Dgm_Len. Total proceeds of the acquired alert information are then compared to verify all packets as "successfully identified" and "responded-to" for each penetration scenario.

Figure 3 shows the general architecture of Snort's three main modules: (i) preprocessor; (ii) detection engine; (iii) alert. A package that is successfully captured by the sniffer module is converted to the pcap library. The preprocessor sorts the content that is then classified into several categories for compilation in the Snort engine using available rules. These rules critically affect the attack's suggested outcome. On the contrary, researchers have proposed a modification mechanism to update and thus optimize rule capabilities [22, 25].

3. Investigation Method

The investigation uses a small-scale IoT network testbed consisting of multiple hardware including the DHT22 sensor, MQ2 sensor, soil moisture sensor, water level sensor, two Zigbee type sensors, and WeMos D1 microcontroller equipped with the ESP8266 Wi-Fi module. Two middleware modules using Raspberry Pi microcontroller are used for communicating the Zigbee and Wi-Fi types of equipment. In addition, the testbed utilizes supporting software such as MySQL database, DoS tools Hping3, Apache Web Server, and Snort as IDS. Figure 4 illustrates the topology of the testbed. Table 1 shows short descriptions of the equipment.

Figure 5 depicts the overall process flow of the packet capturing and decoding in the experiments. Figure 5(a) depicts the capturing work flow. Figure 5(b) shows the process of raw data extraction to obtain unique features after data processing and training. This process is necessary to extract parameters required to search for and identify common ground patterns of a BFA. When running a BFA scenario, the pcap file activates the sniffing process and produces raw data that is not humanly readable due to the unique structure of the IPv4 header, which has hidden layers that depend on protocols and sundry encapsulation processes. To facilitate the process of training, file types that can be processed and generally accepted are required. Here, a "csv" file type is used as the result of raw data processing. A search process for the same pattern comprises raw data derived from a harsher algorithm that classifies certain values in a field of interest. Figure 5(b) presents the algorithm's flowchart. Having finished the features extraction process, solid parameters are generated to benchmark the attack archetype.

3.1. Phases. For the purpose of simulating the insider attack by two attacker hosts that targeted a main services server, six arrangements and assumptions to the attack stages are set up:

(1) To distinguish normal traffic from attack traffic, the attacks are divided into several time-based stages, and both targeted machines and attack method are identified.

(2) Host machine bearing IP address 192.168.10.2 is the network-monitoring server as well as FTP server and able to capture traffic and has the visualization module.

(3) TCPdump directly records all network traffic using the libpcap library to capture packets. It also serves as a packet sniffer. TCPdump produces raw data (pcap

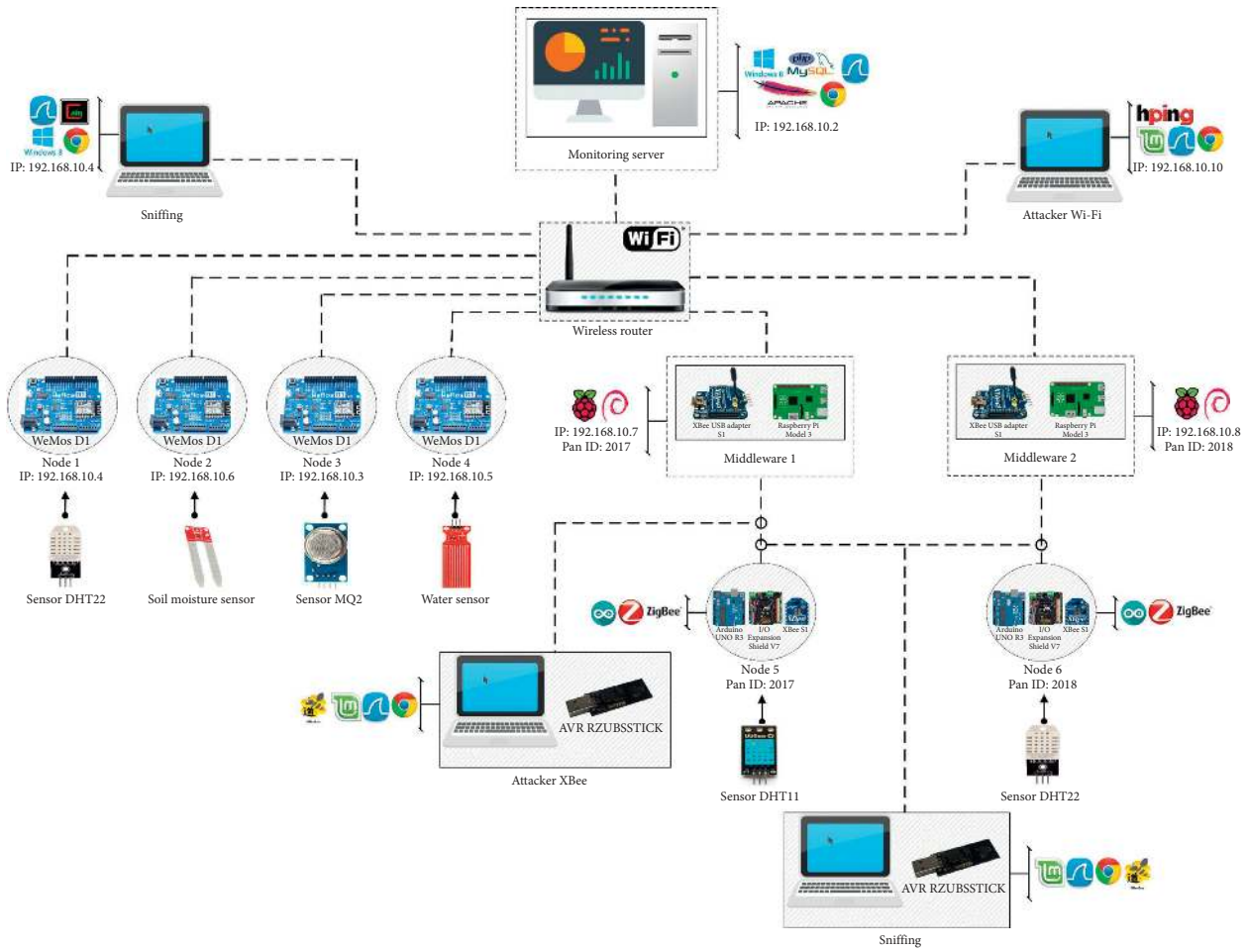


Figure 4: Testbed topology.

Table 1: List of equipment for the testbed.

Name	IP address	Description
Node 1	192.178.10.3	DHT22 sensor
Node 2	192.178.10.4	Soil moisture sensor
Node 3	192.178.10.5	MQ2 sensor
Node 4	192.178.10.6	Water level sensor
Node 5	Pan-ID: 2017	Zigbee sensor
Node 6	Pan-ID: 2018	Zigbee sensor
Middleware 1	192.178.10.7	Raspberry Pi
Middleware 1	192.178.10.8	Raspberry Pi
User	192.168.10.10	Laptop as sniffer
Attacker	192.168.10.11	Laptop as sniffer (using Wi-Fi)
Attacker	192.168.10.30	Laptop as sniffer (using Wi-Fi)
Server	192.168.10.2	Monitoring and FTP server, as well as sniffer
XBee attacker	—	Laptop (using XBee)
Router	—	Wireless router

file) during experiments through the all sniffer nodes.

(4) IDS Snort 2.8.5.2 (Build 121) PCRE ver. 8.12 is run on user machine 192.168.10.10. Snort identifies threat patterns and provides real-time attack alerts. The Snort captures all traffic for comparison with signatures database.

(5) Attacks are launched by machines 192.168.10.11, 192.168.10.30 (Wi-Fi Hackers) and are running Windows 8, and machine connects through the XBee protocol (XBee Hacker).

(6) Targeted host is a server running Windows Server 2003 (IP address 192.168.10.2) and multiple applications including Web, FTP, and MS SQL.

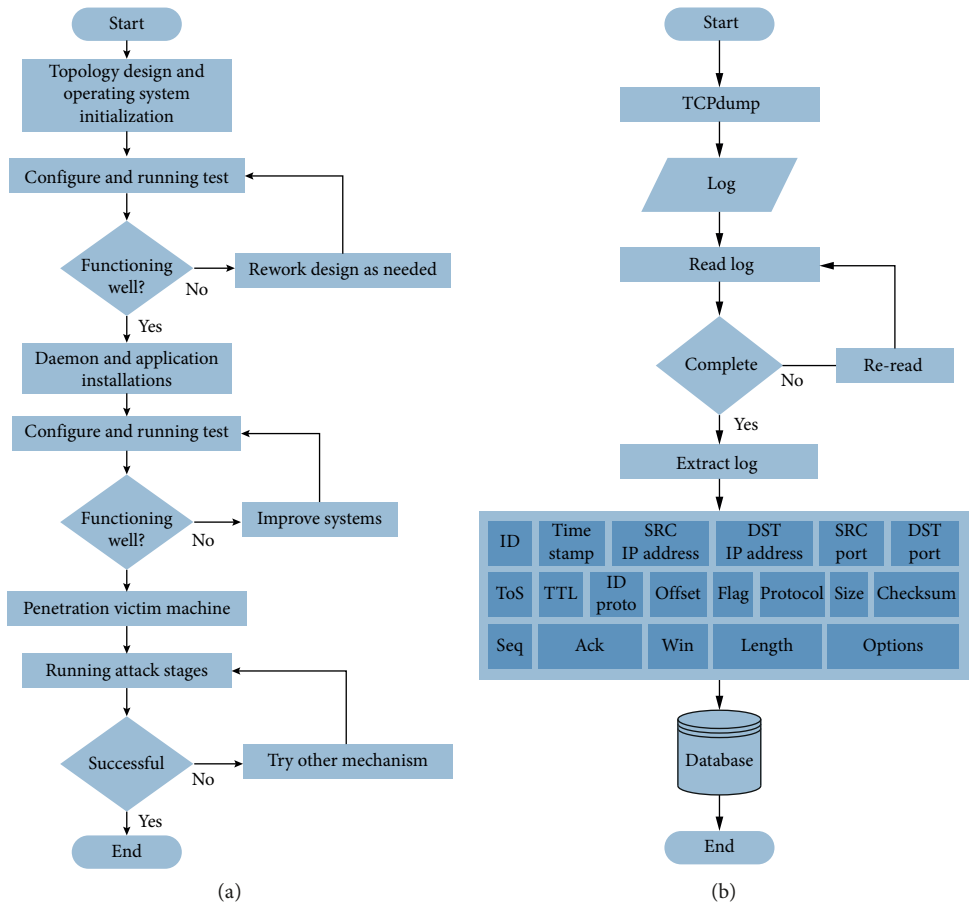


Figure 5: (a) Flowchart of packet capturing preparation; (b) flowchart of packet decoding.

3.2. Internal Attack Activity. The following are the detailed activities for the attack investigation:

(1) The attack machine 192.168.10.11 scans the local network 192.168.10.0 using Nmap

(2) The attack machine 192.168.10.11 scans the local network 192.168.10.0 using Nikto

(3) The attack machine 192.168.10.11 scans "FIN" to the local network 192.168.10.0 using Zenmap

(4) The attack machine 192.168.10.11 scans the local network 192.168.10.0 using HTTPrint

(5) Attacker probes the network 192.168.10.0 via attack machine 192.168.10.11 using Nessus

(6) Attacker probes the network 192.168.10.0 via attack machine 192.168.10.30 using GFI LanGuard

(7) Attacker probes the network 192.168.10.0 via attack machine 192.168.10.30 using Nsteatlth HTTP

(8) Attacker probes open port of the server machine 192.168.10.2 via attack machine 192.168.10.30 using Netools

(9) Attacker finds an open port for potential penetration: 21 (FTP), 22 (SSH), 80 (HTTP), 111 (RPCbin), and 3306 (MySQL)

(10) Attackers attempt XSS via HTTP port 80

(11) Attackers guess the password by the FTP brute force attempt

(12) Attackers try to login the host by WinSCP Software

(13) Attackers upload Trojan, copies the file to the host

(14) Attackers create/mkdir "tools" in the host by WinSCP

(15) Attackers login to the server machine 192.168.10.2 via Putty software

(16) Attackers login to the server via user: ant

(17) Attackers try to activate the backdoor through executing Trojan malware

(18) Attackers run command "#sudo cat/etc/shadow"

(19) Attackers browse directory "administrator" and "ant"

(20) Attackers two times login using "su" to try escalating privilege

(21) Attackers try to find the directory password via "#sudo/etc/passwd"

(22) Attackers attempt to ARP Poison via Cain&Abel

(23) Attackers launch ICMP flooding

(24) Attackers attempt to send TCP SYN via Trinoo

(25) Attackers flooding traffic to the server machine 192.168.10.2 via UDP.

The success of the experiment relies on proving the existence of the unique attack pattern scenarios by (i) comparing attack timelines, (ii) capturing packages by the sniffer, and (iii) retrieving data log results from the targeted machine. This completed set of data is combined to prove whether the observed pattern matched the BFA profile or any other attack types. The authors combined all three datasets with raw data compiled by the Snort engine to certify the correctly identified attack, as recognized by the Snort signature database. Results were reiteratively validated by the time line as well as by reviewing sundry information derived by the Snort alert. Hence, correct attack scenario results on the targeted machine were robustly demonstrated.

The observations will reveal a looping pattern in a single line of data packets at a certain point in time with the same value in the same field. Figure 6 shows a simplified process for sorting the data as a single plot that visualized captured alert results that were identified as an attack. Figure 7 demonstrates performance stages in attack identification as an online display that is spelled out by the pseudocode.

4. Experimental Results

This section presents investigation results in stages. Having completed the process of traffic sniffing on the network to produce raw data, the captured data are then extracted and read to determine the pattern for each attack model. In this case, both BFA pattern and normal FTP pattern were focal observations. Figure 8 depicts an example of the extracted data indicating that raw data hold unique reiterated fields that revealed an ongoing process. Fields of the extracted payload are timestamp, packet size, total packet length, protocol flags, windowing, protocol length, content, and signature. Figure 8 shows the traffic data with time stamp values: 12475, 12476, and 12477 are repeating. The alerts are displayed on an integrated dashboard, as depicted in Figure 9. The detection system displays any attack reports that originated from Snort alerts and then visualizes them in an online manner. This application produces values by sorting and filtering fields of traffic packets that are previously analyzed by the sniffing process.

At a separate process, Snort concurrently generates groups of alerts recognized as patterns that matched the database signature. Figure 10 shows some of these alerts, which were vigorously determined by employing "rules engines" of Snort. Each rule has a unique pattern that is recognized as an attack; however, due to a major problem in the Snort detection system—high false alarms that affect matrices values for false positives (FP), false negatives (FN), true positives (TP), and true negatives (TN)—Snort cannot serve as a primary reference. Nevertheless, Snort is the standard established by prior studies as the engine for comparison.

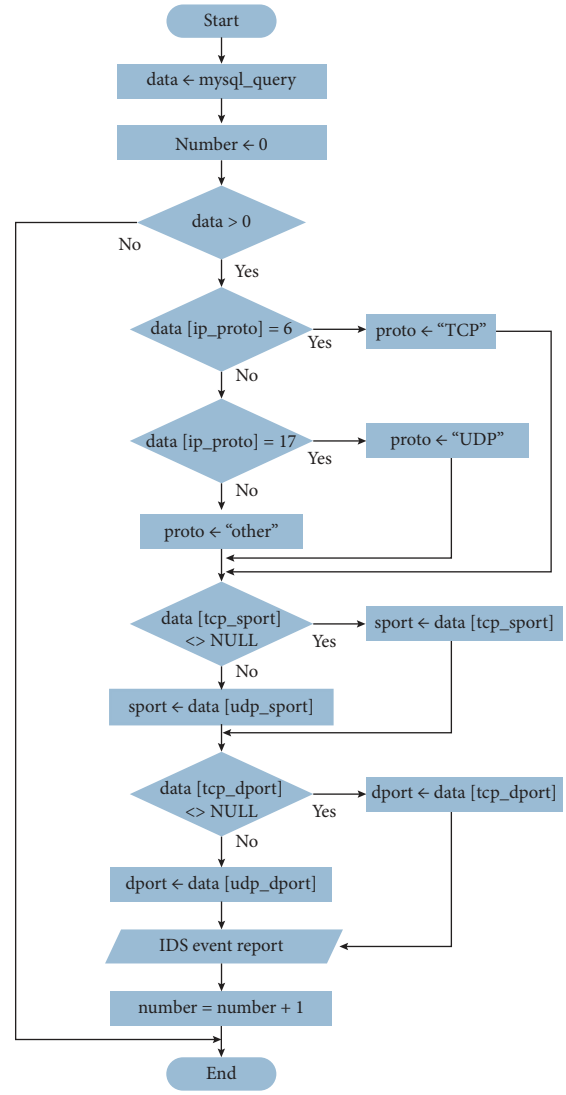


Figure 6: Flowchart of the sorting process.

```
Signature = getDataFromTable(signature)
event     = getDataFromTable(event)
iphdr     = getDataFromTable(iphdr)
tcphdr    = getDataFromTable(tcphdr)
udphdr    = getDataFromTable(udphdr)

IF match(signature,event) and match(event,iphdr) and
match(iphdr,tcphdr) and match(tcphdr,udphdr) then
  result <- signature
  result <- inner_join(result, event)
  result <- inner_join(result, iphdr)
  result <- left_join (result, tcphdr)
  result <- left_join (result, udphdr)
ELSE
  result <- NULL
ENDIF

IF result <> NULL then
  result[count]  = count_rows(result)
  result[ip_src] = convert_long2ip(result[ip_src])
  result[ip_dst] = convert_long2ip(result[ip_dst])

  sortByEvent(result)

  write (result)
ENDIF
```

Figure 7: Pseudocode for performing stages in identifying the attacks procedure.

12454	81	67	FTP Ctrl	AP	65535	27	S: 220-Microsoft FTP Service	
12455	62	48	FTP Ctrl	S	65535	0		
12456	73	59	FTP Ctrl	AP	65508	19	C: CSID Name=NESSUS;	Invalid FTP Command
12457	62	48	FTP Ctrl	A.S	16384	0		
12458	54	40	FTP Ctrl	A	65535	0		
12459	70	56	FTP Ctrl	AP	65516	16	S: 220 FTP SERVER	
12460	81	67	FTP Ctrl	AP	65535	27	S: 220-Microsoft FTP Service	
12461	54	40	FTP Ctrl	A.R	0	0		
12462	62	48	FTP Ctrl	S	65535	0		
12463	62	48	FTP Ctrl	A.S	16384	0		
12464	54	40	FTP Ctrl	A	65535	0		
12465	81	67	FTP Ctrl	AP	65535	27		
12466	62	48	FTP Ctrl	S	65535	0		
12467	62	48	FTP Ctrl	A.S	16384	0		
12468	54	40	FTP Ctrl	A	65535	0		
12469	81	67	FTP Ctrl	AP	65535	27		
12472	54	40	FTP Ctrl	A	65508	0	Seq=1851506777,Ack=3622385529,F=.A ....,Len= 0,Win=65508	Invalid FTP Command
12473	54	40	FTP Ctrl	A	65508	0		
12474	54	40	FTP Ctrl	A	65508	0		
12475	70	56	FTP Ctrl	AP	65535	16	S: 220 FTP SERVER	
12476	70	56	FTP Ctrl	AP	65535	16	S: 220 FTP SERVER	
12477	70	56	FTP Ctrl	AP	65535	16	S: 220 FTP SERVER	
12478	85	71	FTP Ctrl	AP	65492	31	C: USER nessus856270379752370680	
12479	107	93	FTP Ctrl	AP	65504	53		
12480	62	48	FTP Ctrl	S	65535	0		
12481	62	48	FTP Ctrl	A.S	16384	0		
12482	54	40	FTP Ctrl	A	65535	0		

Figure 8: Features extraction.

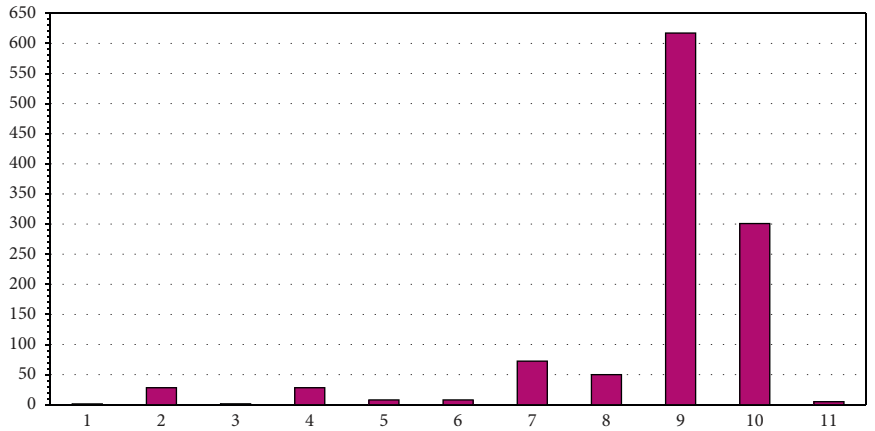
Figures 11 and 12 show attack patterns after insider brute force attacks happened on the FTP server. Figure 11(a) displays the frequency of attack when it happened (from 21.05 to 21.10 hours). The attacker uses two techniques: brute force and dictionary. Figures 11(b) and 11(c) show the number of attacks per-second for brute force attack and dictionary attack, respectively. Both types show similar pattern that explain the attacks have same main characteristics. During the "FTP SITE EXEC attempt" attack—included in the remote to local (R2L)—the attacker can perform the command "SITE EXEC" on the targeted machine by providing the path name using certain characteristics. In other words, a remote attacker can execute commands on the FTP server, including the creation of certain directories. Consequently, this attack allows the attacker to gain root-level access to the system.

Figure 12 shows the traffic from one node to another during the BFA happens. Both FTP SITE EXEC command and FTP parameters were malformed and hence identified by the "pattern of attack" rules identification procedure. The red line indicates a successful attack. Thus, the IoT system administrator visually is able to spot something wrong is happening on the IoT network.

Figure 13 shows a characteristic/pattern of a change working directory (CWD) attack where the attack is included in R2L. R2L focuses on successful anonymous logins that access the right to write in the system and plant backdoor or other malware. Here, the attacker repeatedly assaults the system with a pattern that differs from previous attacks. The pattern reflects several CWD stages. Having successfully entered anonymously, the attacker attempts to change the directory, which is preceded by a passive mode command (PASV) that enables responsive communication. The attacker then follows with a Network Services Lead Team (NLST) to restore files to a specified directory. Figure 13 also displays traffic during BFA scenario testing, scanning Denial-of-Service (DOS) flooding of the target. The traffic information in Figure 13 clearly illustrates offensive package flow from the attacker to the targeted machine.

Information inside the box A of Figure 13 indicates alert from Snort that is displayed in the form of information on suspected attacks. This information is to be compared with proof that attacks occurred from the detection procedure. Information inside the box B of Figure 13 shows extraction results from the raw data using the identification procedure in Figure 7. Both information shows similar attack characteristic/pattern that generated alerts along with unique field values that repeated in a single attack scenario. Hence, it is clear that the simulated BFA scenario has generated a unique characteristic/pattern. This conclusion was confirmed by Snort alert results. Thus, the pattern identification procedure shown in Figure 7 works well in detecting the BFA.

No	Event ID	Event Name	Time	Count	Priority	IP source	IP Destination	Port Source	Port Destination	Protocol (TCP/UDP)	TTL	IPlen	Dgm Len	TCPlen
1	50224	SNMP request udp	2013-03-26 20:49:16	2	2	202.9.69.15	202.9.69.77	34531	161	UDP	63	5	75	N/A
2	1843	(snort_decoder): Experimental Tcp Options found	2013-07-18 15:52:21	4	3	115.114.5.66	202.9.69.78	33195	22	TCP	114	5	64	N/A
3	1840	COMMUNITY SIP TCP/IP message flooding directed to SIP proxy	2013-07-15 15:54:43	2	2	222.36.0.46	202.9.69.78	33195	22	TCP	48	5	52	N/A
4	1839	ICMP Destination Unreachable Communication with Destination Host is Administratively Prohibited	2013-07-15 06:12:54	1	3	192.96.206.37	202.9.69.78	59749	22	other (1)	42	5	68	N/A
5	1837	COMMUNITY MISC BAD-SSL tcp detect	2013-07-15 04:02:00	2	3	202.9.69.78	222.73.244.22	32993	22	TCP	64	5	92	N/A
6	1836	ICMP Destination Unreachable Communication with Destination Host is Administratively Prohibited	2013-07-14 12:36:32	1	3	218.63.65.189	202.9.69.78	32993	22	other (1)	47	5	76	N/A
7	1835	COMMUNITY MISC BAD-SSL tcp detect	2013-07-14 12:01:22	2	3	202.9.69.78	58.32.217.105	32993	22	TCP	64	5	104	N/A
8	1826	COMMUNITY SIP TCP/IP message flooding directed to SIP proxy	2013-07-12 05:25:13	2	2	202.9.69.78	121.14.104.232	59887	22	TCP	64	5	92	N/A



1. SNMP request udp
2. (spp_ssh) Protocol mismatch
3. (portscan) TCP Distributed Portscan
4. (snort_decoder): Experimental Tcp Options found
5. ICMP Destination Unreachable Communication Administratively Prohibited
6. (portscan) Open Port
7. COMMUNITY MISC BAD-SSL tcp detect
8. (portscan) TCP Portscan
9. COMMUNITY SIP TCP/IP message flooding directed to SIP proxy
10. ICMP Destination Unreachable Communication with Destination Host is Administratively Prohibited
11. (snortdecoder) WARNING: ICMP

Figure 9: Snort alert snapshot.

5. Conclusion and Future Works

This paper investigated brute force attack that attempts to gain escalating privileges on an FTP server of the IoT network. The attack likely occurs due to weaknesses in the FTP's service that lacks encryption at a moment when running the process of a three-way handshake. Moreover, attacks can originate within the network and potentially occur because an extensive upholstery system was improperly set up to limit local user access, which, in turn, affects the entire security of the system.

Experimental observations recognized BFA patterns on an FTP service that matched the Snort analysis of captured data. Snort provides information to the system administrator in the form of a warning alert to report network occurrences. Findings from the experiments provide some visual protection assistance for researchers and practitioners. The authors intend to investigate IoT attack patterns with more complicated network topologies and scenarios, specifically, those launched by botnets on the IoT network. Finally, Table 2 summarizes the findings from the experiments.

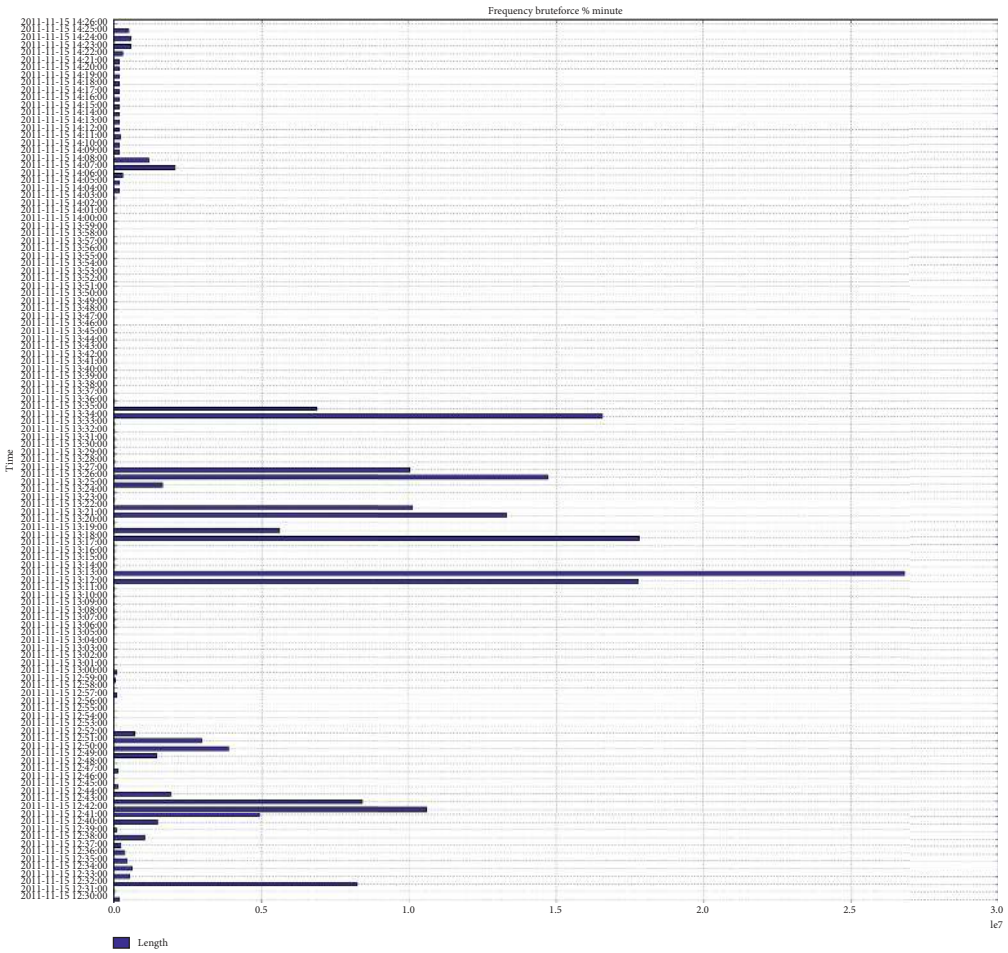
Table 2 shows the patterns (features) used as signature in the identification procedure of Figure 7 accurately characterizing the attacks, and thus, the attack is successfully detected. This result was confirmed by Snort that also produces an alert on the detection. Therefore, this result indirectly confirms that the statistical relationship is

```
[**] [1:361:15] FTP SITE EXEC attempt [**]
[Classification: Potentially Bad Traffic] [Priority: 2]
11/15-13:42:10.870697 10.10.10.15:5718 -> 10.10.10.25:21
TCP TTL:128 TOS:0x0 ID:38290 IpLen:20 DgmLen:73 DF
***AP*** Seq: 0x8FE3BDED  Ack: 0x6924059D  Win: 0x43BE  TcpLen: 20

[**] [125:4:1]FTP command parameters were malformed [**][Priority: 3]
11/15-13:42:10.871329 10.10.10.15:5718 -> 10.10.10.25:21
TCP TTL:240 TOS:0x10 ID:0 IpLen:20 DgmLen:69
***AP*** Seq: 0x8FE3BE0E  Ack: 0x692405DD  Win: 0xFF99  TcpLen: 20

[**] [1:336:10] FTP CWD ~root attempt [**]
[Classification: Potentially Bad Traffic] [Priority: 2]
11/15-13:42:08.901613 10.10.10.15:5683 -> 10.10.10.25:21
TCP TTL:128 TOS:0x0 ID:38101 IpLen:20 DgmLen:51 DF
***AP*** Seq: 0xFEB2D317  Ack: 0x6DF284FD  Win: 0x43BE  TcpLen: 20
```

Figure 10: Snort alert snapshot.



(a)

Figure 11: Continued.

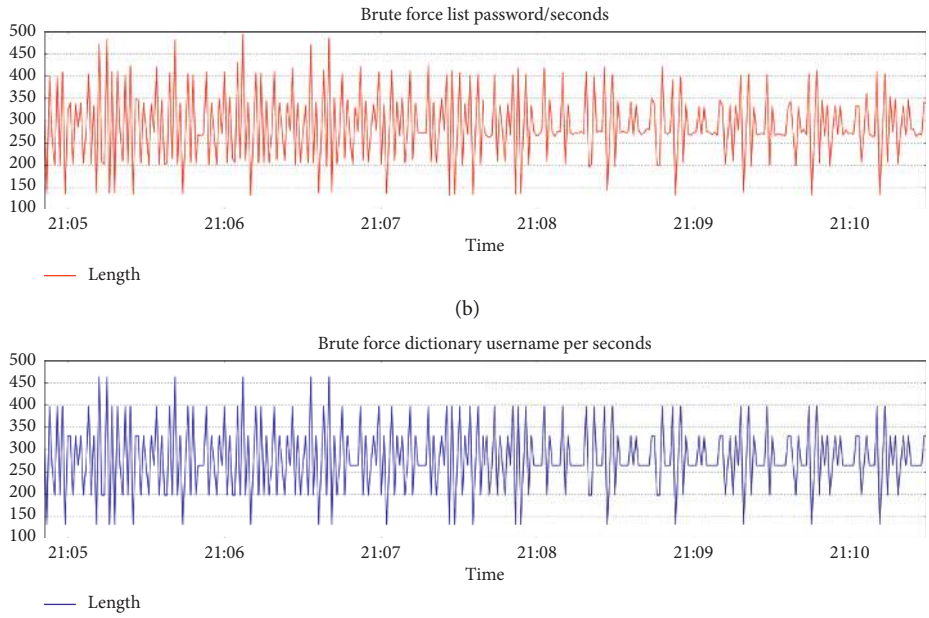


Figure 11: Snapshot of brute force attempts.

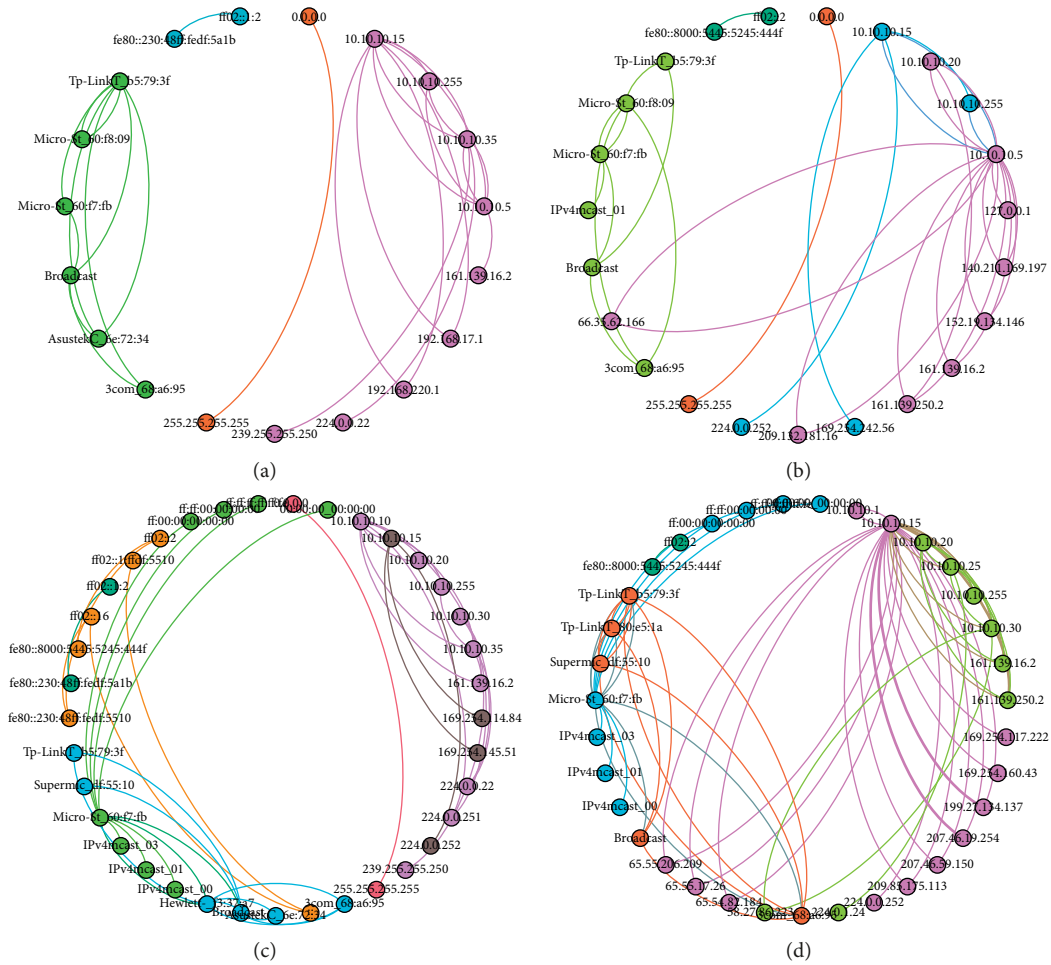


Figure 12: Continued.

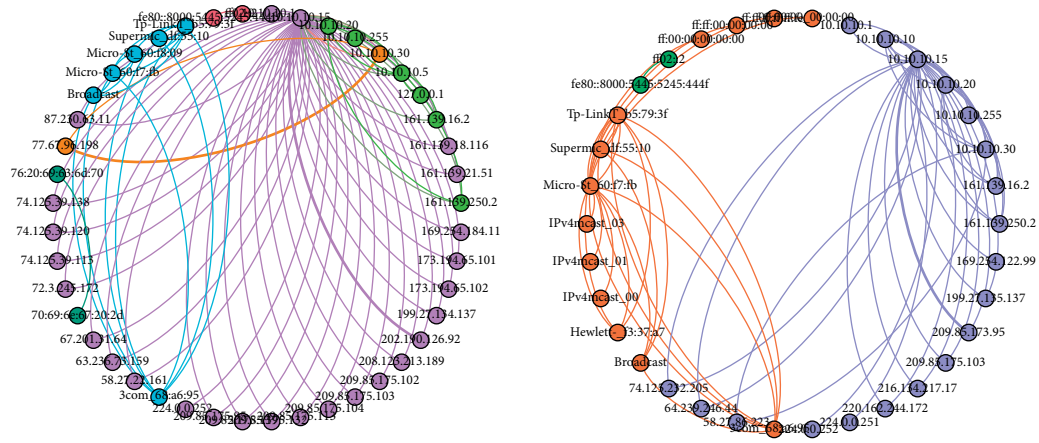


Figure 12: Snapshot of the node-to-node traffics during the attack.

[**] [1:2002383:11] ET SCAN Potential FTP Brute-Force attempt [**]
[Classification: Unsuccessful User Privilege Gain] [Priority: 1]
11/04-22:04:54.225947 10.10.10.5:21 -> 10.10.10.15:32228
TCP TTL:64 TOS:0x0 ID:58716 IpLen:20 DgmLen:62 DF
AP Seq: 0x343934E4 Ack: 0x41BD3B4F Win: 0x16D0 TcpLen: 20
[Xref => http://www.emergingthreats.net/cgi-bin/cvsweb.cgi/sigs/SCAN/SCAN_FTP_Brute_Force]

No	Timestamp	SRC_IP	DST_IP	SRC_port	DST_port	Protocol	Size	Flag	Total_Len	Content
173	04:54.0	10.10.10.15	10.10.10.5	32227	21	FTP Ctrl	54	A.R	40	
174	04:54.0	10.10.10.15	10.10.10.5	32238	21	FTP Ctrl	62	S	48	
175	04:54.0	10.10.10.5	10.10.10.15	21	32238	FTP Ctrl	62	A.S	48	
176	04:54.0	10.10.10.15	10.10.10.5	32238	21	FTP Ctrl	54	A	40	
177	04:54.1	10.10.10.5	10.10.10.15	21	32238	FTP Ctrl	74	AP	60	S: 220 (vsFTPd 1.1.3)
178	04:54.1	10.10.10.15	10.10.10.5	32238	21	FTP Ctrl	66	AP	52	C: USER guest
179	04:54.1	10.10.10.5	10.10.10.15	21	32238	FTP Ctrl	60	A	40	
180	04:54.1	10.10.10.5	10.10.10.15	21	32238	FTP Ctrl	88	AP	74	S: 331 Please specify the password.
181	04:54.1	10.10.10.15	10.10.10.5	32238	21	FTP Ctrl	66	AP	52	C: PASS runny
182	04:54.1	10.10.10.5	10.10.10.15	21	32238	FTP Ctrl	60	A	40	
183	04:54.2	10.10.10.5	10.10.10.15	21	32228	FTP Ctrl	76	AP	62	S: 530 Login incorrect.
184	04:54.2	10.10.10.15	10.10.10.5	32228	21	FTP Ctrl	54	A.F	40	

[**] [1:1229:7] FTP CWD ... [**]
[Classification: Potentially Bad Traffic] [Priority: 2]
08/07-02:33:30.183629 10.10.10.15:5241 -> 10.10.10.25:21
TCP TTL:128 TOS:0x0 ID:60987 IpLen:20 DgmLen:94 DF
AP Seq: 0xA0670E4B Ack: 0x38AD9A35 Win: 0xFF4E TcpLen: 20
[Xref => http://www.securityfocus.com/bid/9237]

A

No	Timestamp	SRC_IP	DST_IP	SRC_port	DST_port	Protocol	Size	Flag	Total_Len	Content
24186	33:30.0	10.10.10.25	10.10.10.15	21	5241	FTP Ctrl	70	AP	56	220 FTP SERVER
24187	33:30.0	10.10.10.15	10.10.10.25	5241	21	FTP Ctrl	70	AP	56	USER anonymous
24188	33:30.0	10.10.10.25	10.10.10.15	21	5241	FTP Ctrl	126	AP	112	331 Anonymous access allowed, send identity (e-mail name) as password.
24189	33:30.0	10.10.10.15	10.10.10.25	5241	21	FTP Ctrl	78	AP	64	PASS nessus @nessus.org
24190	33:30.0	10.10.10.25	10.10.10.15	21	5241	FTP Ctrl	85	AP	71	230-WELCOM TO FTP SERVER 2003
24191	33:30.2	10.10.10.15	10.10.10.25	5241	21	FTP Ctrl	54	A	40	
24192	33:30.2	10.10.10.25	10.10.10.15	21	5241	FTP Ctrl	85	AP	71	230 Anonymous user logged in.
24193	33:30.2	10.10.10.15	10.10.10.25	5241	21	FTP Ctrl	108	AP	94	CWD ../../../../../../../../../../../../
24194	33:30.2	10.10.10.25	10.10.10.15	21	5241	FTP Ctrl	83	AP	69	250 CWD command successful.
24195	33:30.2	10.10.10.15	10.10.10.25	5241	21	FTP Ctrl	60	AP	46	PASV
24196	33:30.2	10.10.10.25	10.10.10.15	21	5241	FTP Ctrl	101	AP	87	227 Entering Passive Mode (10,10,10,25,18,4).
24197	33:30.2	10.10.10.15	10.10.10.25	5241	21	FTP Ctrl	62	S	48	
24198	33:30.2	10.10.10.25	10.10.10.15	21	5241	FTP Ctrl	62	A.S	48	
24199	33:30.2	10.10.10.15	10.10.10.25	5241	21	FTP Ctrl	54	A	40	
24200	33:30.2	10.10.10.25	10.10.10.15	21	5241	FTP Ctrl	69	AP	55	RETR boot.ini
24201	33:30.2	10.10.10.15	10.10.10.25	5241	21	FTP Ctrl	113	AP	99	550 boot.ini: The system cannot find the file specified.
24202	33:30.2	10.10.10.25	10.10.10.15	21	5241	FTP Ctrl	54	A.R	40	
24203	33:30.2	10.10.10.15	10.10.10.25	5241	21	FTP Ctrl	60	AP	46	QUIT

B

Figure 13: Confirmation of alert on attack by the detection algorithm (A) and by Snort (B).

Table 2: Summary of the findings.

Method	Findings
Snort	Successfully detected.
Attack identification procedure (Figure 7)	Successfully detected. Consistent to Snort results
Statistical relationship approach	Successfully provides attack traffic patterns
Visualization	Provides clear traffic pattern of the brute force attacks

used for analyzing the attack works well. Visualization assists the network administrator to identify any anomalies/attacks easily.

Data Availability

The dataset is available online at http://dataset.ilkom.unsri.ac.id.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The research leading to these results has received funding from International Collaboration Grant between Universitas Sriwijaya and Universiti Teknologi Malaysia.

References

[1] A. R. Sfar, E. Natalizio, Y. Challal, and Z. Chtourou, "A roadmap for security challenges in the Internet of Things," *Journal of Digital Communications and Networks*, vol. 4, no. 2, pp. 118–137, 2018.

[2] H. Zhang, D. Yao, N. Ramakrishnan, and Z. Zhang, "Causality reasoning about network events for detecting stealthy malware activities," *Computers & Security*, vol. 58, pp. 180–198, 2016.

[3] J. Jang-Jaccard and S. Nepal, "A survey of emerging threats in cybersecurity," *Journal of Computer and System Sciences*, vol. 80, no. 5, pp. 973–993, 2014.

[4] P. Steiner, "Why FTP is no longer up to the job," *Network Security*, vol. 2010, no. 5, pp. 7–9, 2010.

[5] B. Sieklik, R. Macfarlane, and W. J. Buchanan, "Evaluation of TFTP DDoS amplification attack," *Computers & Security*, vol. 57, pp. 67–92, 2016.

[6] A. Joshi, M. Wazid, and R. H. Goudar, "An efficient cryptographic scheme for text message protection against brute force and cryptanalytic attacks," *Procedia Computer Science*, vol. 48, pp. 360–366, 2015.

[7] D. Stiawan, M. Y. Idris, A. H. Abdullah, M. AlQurashi, and R. Budiarto, "Penetration testing and mitigation of vulnerabilities windows server," *International Journal of Network Security*, vol. 18, pp. 501–513, 2016.

[8] D. Stiawan, M. Y. Idris, and A. H. Abdullah, "Penetration testing and network auditing: Linux," *Journal of Information Processing Systems*, vol. 11, pp. 104–115, 2015.

[9] A. Austin, C. Holmgreen, and L. Williams, "A comparison of the efficiency and effectiveness of vulnerability discovery techniques," *Information and Software Technology*, vol. 55, no. 7, pp. 1279–1288, 2013.

[10] V. Broucek and P. Turner, "Technical, legal and ethical dilemmas: distinguishing risks arising from malware and cyber-attack tools in the 'cloud'-a forensic computing perspective," *Journal of Computer Virology and Hacking Techniques*, vol. 9, no. 1, pp. 27–33, 2013.

[11] H. S. Venter, J. H. P. Eloff, and Y. L. Li, "Standardising vulnerability categories," *Computers & Security*, vol. 27, no. 3-4, pp. 71–83, 2008.

[12] H. S. Venter and J. H. P. Eloff, "A taxonomy for information security technologies," *Computers & Security*, vol. 22, no. 4, pp. 299–307, 2003.

[13] K. Helkala, N. Svendsen, P. Thorsheim, and A. Wiehe, "Cracking associative passwords," in *Secure IT Systems*, A. Jøsang and B. Carlsson, Eds., vol. 7617, pp. 153–168, Springer Berlin Heidelberg, Berlin, Germany, 2012.

[14] E. S. Pilli, R. C. Joshi, and R. Niyogi, "Network forensic frameworks: survey and research challenges," *Digital Investigation*, vol. 7, no. 1-2, pp. 14–27, 2010.

[15] J. Vykopal, "A flow-level taxonomy and prevalence of brute force attacks," in *Advances in Computing and Communications*, A. Abraham, J. Lloret Mauri, J. F. Buford, J. Suzuki, and S. M. Thampi, Eds., vol. 191, pp. 666–675, Springer Berlin Heidelberg, Berlin, Germany, 2011.

[16] M. GhasemiGol, A. Ghaemi-Bafghi, and H. Takabi, "A comprehensive approach for network attack forecasting," *Computers & Security*, vol. 58, pp. 83–105, 2016.

[17] S. Zeadally and A. Flowers, "Cyberwar: the what, when, why, and how [commentary]," *IEEE Technology and Society Magazine*, vol. 33, no. 3, pp. 14–21, 2014.

[18] Y. Joshi, D. Das, and S. Subir, "Mitigating man in the middle attack over secure sockets layer," in *Proceedings of the 2009 IEEE International Conference on Internet Multimedia Services Architecture and Applications (IMSAA)*, Bangalore, India, December 2009.

[19] C. Nithiyanandam, D. Tamilselvan, S. Balaji, and V. Sivaguru, "Advanced framework of defense system for prevention of insider's malicious behaviors," in *Proceedings of the International Conference on Recent Trends In Information Technology (ICRTIT)*, pp. 434–438, Chennai, India, April 2012.

[20] R. Sadoddin and A. A. Ghorbani, "An incremental frequent structure mining framework for real-time alert correlation," *Computers & Security*, vol. 28, no. 3-4, pp. 153–173, 2009.

[21] J. Shangjie, L. Meijian, and W. Zhentao, "Research and Design of Preprocessor plugin based on PCRE under snort platform," in *Proceedings of the International Conference on Control, Automation and Systems Engineering (CASE)*, pp. 1–4, Singapore, July 2011.

[22] K. Thongkanchorn, S. Ngamsuriyaroj, and V. Visoottiviseth, "Evaluation studies of three intrusion detection systems under various attacks and rule sets," in *Proceedings of the IEEE International Conference of IEEE Region 10 (TENCON 2013)*, pp. 1–4, Xi'an, China, October 2013.

[23] V. Visoottiviseth and N. Bureenok, "Performance comparison of ISATAP implementations on FreeBSD, RedHat, and Windows," in *Proceedings of the 22nd International Conference on Advanced Information Networking and Applications—Workshops (Aina Workshops 2008)*, pp. 547–552, Ginowan, Japan, March 2008.

[24] L. Yang and D. Weng, *Snort-based Campus Network Security Intrusion Detection System Information Engineering and Applications*, R. Zhu and Y. Ma, Eds., Vol. 154, Springer, London, UK, 2012.

[25] G. Gowrison, K. Ramar, K. Muneeswaran, and T. Revathi, "Minimal complexity attack classification intrusion detection system," *Applied Soft Computing*, vol. 13, no. 2, pp. 921–927, 2013.