Forensic Analysis on False Data Injection Attack on IoT Environment

Saiful Amin Sharul Nizam[1], Zul-Azri Ibrahim[2], Fiza Abdul Rahim[3]
Hafizuddin Shahril Fadzil[4], Haris Iskandar Mohd Abdullah[5], Muhammad Zulhusni Mustaffa[6]
UNITEN R&D Sdn. Bhd., Selangor, Malaysia[1, 4, 5, 6]
College of Computing and Informatics, Universiti Tenaga Nasional, Malaysia[2]
Razak Faculty of Technology and Informatics, Universiti Teknologi Malaysia, Malaysia[3]
Institute of Informatics and Computing Energy, Universiti Tenaga Nasional, Malaysia[2, 3]

Abstract—**False Data Injection Attack (FDIA) is an attack that could compromise Advanced Metering Infrastructure (AMI) devices where an attacker may mislead real power consumption by falsifying meter usage from end-users smart meters. Due to the rapid development of the Internet, cyber attackers are keen on exploiting domains such as finance, metering system, defense, healthcare, governance, etc. Securing IoT networks such as the electric power grid or water supply systems has emerged as a national and global priority because of many vulnerabilities found in this area and the impact of the attack through the internet of things (IoT) components. In this modern era, it is a compulsion for better awareness and improved methods to counter such attacks in these domains. This paper aims to study the impact of FDIA in AMI by performing data analysis from network traffic logs to identify digital forensic traces. An AMI testbed was designed and developed to produce the FDIA logs. Experimental results show that forensic traces can be found from the evidence logs collected through forensic analysis are sufficient to confirm the attack. Moreover, this study has produced a table of attributes for evidence collection when performing forensic investigation on FDIA in the AMI environment.**

Keywords—Advanced Metering Infrastructure (AMI); False Data Injection Attack (FDIA); man in the middle (MITM); internet of things (IoT); forensic analysis

I. Introduction

Internet of Things (IoT) offers many benefits and advantages to people in the current modern era [1]. Besides, even in our daily life, IoT has proven to be beneficial. IoT is a system of interrelated intelligent devices that are provided with unique identifiers and given the ability to connect with other devices by exchanging information over a communication network. The IoT is seen as one of the foremost important zones in future development and is expanding tremendous consideration from a wide scope of businesses [2]. IoT will play a major role in improving many sectors such as manufacturing, public security, health care, accommodation, entertainment, environment protection, agriculture, industrial monitoring, intelligent transportation, and traditional metering system.

However, little consideration has been paid to IoT adoption that may affect the IoT device's security measure, such as lack of authentication and insecure communication are among the main problems in most IoT devices [3]. These vulnerabilities will lead to many forms of attacks taking place, such as malware injection, SQL injection [4], false data injection (FDI), man-in-the-middle (MITM) [5], zero-day exploit, distributed denial-of-service (DDoS), DNS tunnelling [6], and many more cyber-attacks. Since the case of the Mirai botnet in 2016, over 600,000 IoT devices were targeted to launch cyberattacks that reached 620 Gigabits at the peak. The number of malware in the cyber world has been growing, giving threats to cybersecurity to face other types of aggravated attacks.

There is also concern about one of the IoT environments, the Advanced Metering Infrastructure (AMI). AMI is a system consisting of modern electronic-digital hardware and software, which enables data measurement intermittently and remote communication continuously. The system gives a few important capacities that were not already possible or had to be performed manually. For instance, the ability to remotely and automatically measure power usage, connect and disconnect service, and voltage monitoring. FDIA is one of the popular attacks that can impact AMI as countries around the globe are implementing an AMI in their infrastructure. Like the MITM attack, FDIA is more toward creating falsified data, which the attacker injected from compromised smart meters to change the actual value sent by another smart meter in AMI. This threat can negatively affect both utilities and customers as it is difficult to investigate from the available log in the AMI [7]. This paper aims to simulate the impact of FDIA on the IoT environment and perform forensic analysis on digital traces from data obtained.

In the next section of this paper, related literature on cyber attacks in the smart grid was reviewed. Subsequently, Section III presents the development of a testbed that is used to simulate the cyber attack in components of the smart grid. Section IV presents the result from the simulation and how forensic investigations are done to investigate FDI attacks in the smart grid environment. Section V provides a conclusion to the paper.

II. Literature Review

A. False Data Injection Attack (FDIA)

The operation of the smart grid faces extreme consequences when the smart meters have been compromised and reporting false power consumption. Most current cases include the crime of electricity stealing. However, a few other sorts of data falsification attacks are conceivable such as FDIA. AMI would be affected by this kind of attack badly as data falsification is difficult to detect.

Based on [8], they work on detecting falsification of data injection attacks focusing on smart grid systems. They made a successful and real-time scheme to distinguish FDIA in smart grids where they evaluate the reliabilities of state estimations by misusing spatial-temporal correlations and trust-based voting. The study's objective is to minimize the harm from the threat of FDIA in smart grids by using these solutions to conduct detection of an attack. This case study was done by simulation of the smart grid and the proposed solutions to detect malicious FDIA. It is suggested that powerful countermeasures are necessary as these kinds of attacks can become highly potential threats as those FDIA are evolving by implementing anti-forensic techniques to prevent detection of the attack.

In [9], the study proposed a system to detect cyber-attacks that aim to sabotage the Instrumentation and Control (I&C) environment. The study intends to provide a last line of defense to sabotage attacks. A system called Goosewolf was produced which has the capability to detect when an adversary has manipulated the process control of the Programmable Logic Controller (PLC). The result obtained in that study shows that the proposed system is effective in checking the capabilities of the PLC and the ability to detect FDIA.

Another study by [10] has focused their work on statistical anomaly detection techniques to solve the difficulties in detecting data falsification in AMI. To identify compromised smart meters for deductive and additive attacks, they have proposed a trust model based on Kullback-Leibler divergence. Moreover, techniques such as the generalized linear and Weibull function-based kernels were proposed for camouflage and conflict attacks. After investigation on comparison under various attacks, which is additive, deductive, camouflage, and conflict, they found out that their models have good high true positive detection and the average false positive is just 8 per cent for most attacks conducted.

B. IoT Testbeds

For the purpose of better understanding on vulnerabilities of IoT devices, researchers utilized a security testbed designed to simulate the attack in a particular environment. The author in [11] illustrated a testbed for securing IoT devices by producing a testbed that can be used as a penetration testing platform to evaluate risks and vulnerabilities of IoT devices. The penetration testing included were port scanning, vulnerability scanning, downgrading attack, search exploits, brute force directories, passwords, port services, and SSL configuration. The software used to perform the testing were Snitch, ZAP, Wascan, Skipfish, Nmap, TLS proper, SSLScan, Nikto, Wireshark, Ettercap, Dirb, SQLmap, WAFWooF, Metasploit, Dex2jar, Binwalk, and UART. The network protocols used was WIFI and BLE. Penetration testing for this analysis was conducted on a smart bulb and IP camera. Vulnerabilities found were very common problems in IoT-based products such as no firewall, authentication in plain text, open ports, lack of certificate, etc.

Moreover, paper [12] displayed a testbed designed to analyze security issues in IoT devices. This testbed indicated design and architecture prerequisites to support the development of penetration testing for the purpose of cybersecurity forensic investigation. They conducted the tests based on the security vulnerabilities in the IoT products such as Amazon Echo, Nest Cam, Phillips hue, SENSE Mother, Samsung SmartThings, Witching HOME, WeMo Smart Crock-Pot, and Netatmo Security Camera. The study was conducted using WIFI and Bluetooth. For control and administration, they handle the process and events using NI TestStand software. A closed source software runs only on Windows OS, which is intensely prohibitive and proprietary. Following a huge downside from limitation in network penetration testing capabilities, the software used to avoid testing from handling passive capture of packets, wireless cards, and other network or low-level functionalities.

In [13], researchers used SecuWear to recognize the weaknesses of commercial hardware. The testbed collects the data needed for distinguishing different attacks, thereby assessing the security of wearables devices. Besides, it gives a method for mitigating information and performing attacks in a network that used WIFI and BLE. The software used to perform the vulnerability assessments, and penetration testing was Wireshark. In that study, the eavesdropping and the Denial of Service (DOS) attack were executed. The results of the study found that SecuWear vulnerabilities may be similar to certain open sources such as false positives when recognizing security issues.

III. TESTBED DEVELOPMENT

Fig. 1 shows the topology of our testbed that is used to perform FDIA. The testbed consists of 4 main hardware components, two units Raspberry Pi 4 Model B, a computer and a switch. The smart meter (192.168.1.13) generates random data to mimic a real smart meter then sends the generated data to the data collector containing one virtual machine running Ubuntu Version 21.04 operating system to act as a data collector using the MYSQL version 10.14.9-MariaDB database which receives incoming data from the smart meter. Attacker smart meter (192.168.1.11) will act as an attacker to perform FDIA that will attempt to tamper the smart meter data to the data collector.

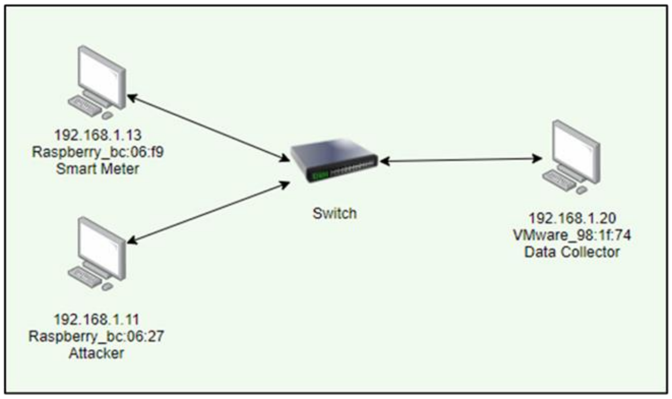


Fig. 1. Testbed Topology.

A comparative analysis between normal traffic logs and logs during the attack was made to verify the FDIA investigations in the IoT environment. Forensic evidence was analyzed based on the packet captured using Wireshark in the form of PCAP files.

IV. Discussion of Findings

The experiments conducted on the testbed were carried out in two phases. The first phase of the experiment was the ‘Normal operation’, and the second phase was the ‘Under Attack’. The details of the experiments will be explained later on in this chapter. Fig. 2 shows the flow of the experiment during normal traffic and under attack.

A. Normal Operations

During the normal traffic phase, the smart meter as shown in Fig. 1 with an IP address of 192.168.1.13 with MAC address of Raspberry bc : 06: f9 will send data to the data collector where the IP address of the database is 192.168.1.20 with MAC address of VMware_98:1f:74. The smart meter will send power consumption data with an interval of 10 seconds between each data to imitate data for 1 week with an interval of 30 minutes between each data interval. In this experiment, 137 data will be collected using the Wireshark version 3.4.5 packet capturing tool. The consumption transmission script will run in 25 minutes to collect data in a total range of 135 to 140 data. Fig. 3 shows the data sent by the smart meter to the data collector, the value of power consumption with the timestamp.

Fig. 4 shows the sample data sent by the smart meter to the data collector in the MySQL database. The first column shows the numbers of data in the database. The second column shows the ID of the smart meter, the third and fourth rows display the timestamp of when the data was accepted, and the last row shows the data value of the power consumption.

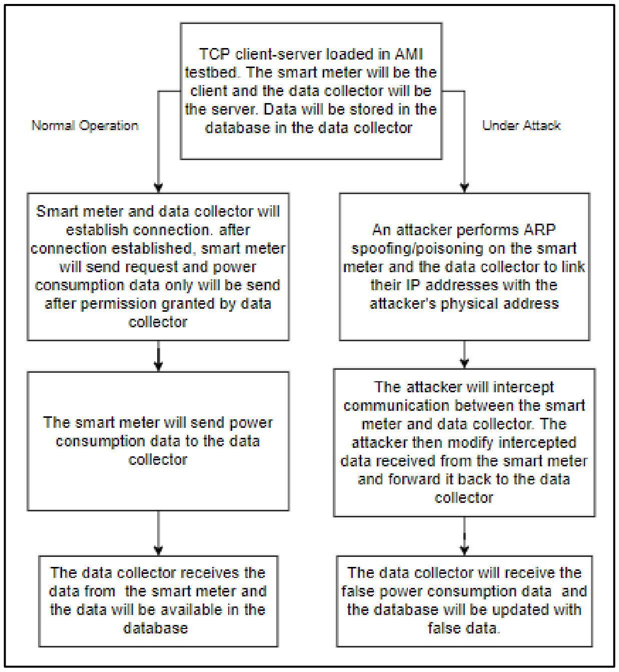
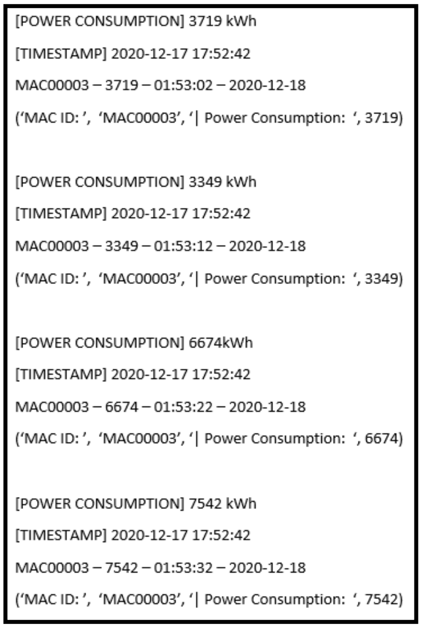
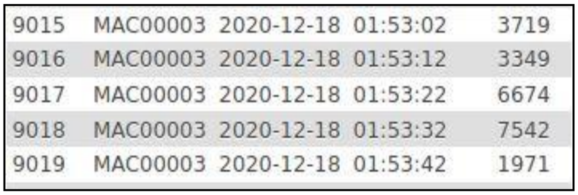

TCP client-server loaded in AMI testbed. The smart meter will be the client and the data collector will be the server. Data will be stored in the database in the data collector
Normal Operation
Under Attack
Smart meter and data collector will establish connection. after connection established, smart meter will send request and power consumption data only will be send after permission granted by data collector
An attacker performs ARP spoofing/poisoning on the smart meter and the data collector to link their IP addresses with the attacker's physical address
The smart meter will send power consumption data to the data collector
The attacker will intercept communication between the smart meter and data collector. The attacker then modify intercepted data received from the smart meter and forward it back to the data collector
The data collector receives the data from the smart meter and the data will be available in the database
The data collector will receive the false power consumption data and the database will be updated with false data.

Fig. 2. Flow of Experiments.



```
[POWER CONSUMPTION] 3719 kWh
[TIMESTAMP] 2020-12-17 17:52:42
MAC00003 – 3719 – 01:53:02 – 2020-12-18
(‘MAC ID: ’, ‘MAC00003’, ‘| Power Consumption: ‘, 3719)

[POWER CONSUMPTION] 3349 kWh
[TIMESTAMP] 2020-12-17 17:52:42
MAC00003 – 3349 – 01:53:12 – 2020-12-18
(‘MAC ID: ’, ‘MAC00003’, ‘| Power Consumption: ‘, 3349)

[POWER CONSUMPTION] 6674kWh
[TIMESTAMP] 2020-12-17 17:52:42
MAC00003 – 6674 – 01:53:22 – 2020-12-18
(‘MAC ID: ’, ‘MAC00003’, ‘| Power Consumption: ‘, 6674)

[POWER CONSUMPTION] 7542 kWh
[TIMESTAMP] 2020-12-17 17:52:42
MAC00003 – 7542 – 01:53:32 – 2020-12-18
(‘MAC ID: ’, ‘MAC00003’, ‘| Power Consumption: ‘, 7542)
```

Fig. 3. Smart Meter Sending Data in Normal Traffic.

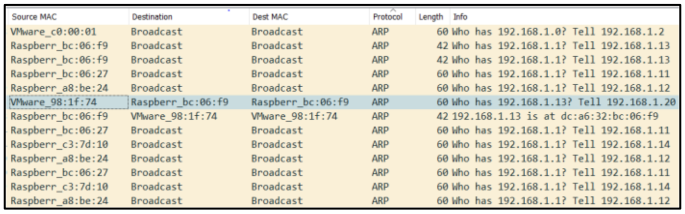


9015	MAC00003	2020-12-18	01:53:02	3719
9016	MAC00003	2020-12-18	01:53:12	3349
9017	MAC00003	2020-12-18	01:53:22	6674
9018	MAC00003	2020-12-18	01:53:32	7542
9019	MAC00003	2020-12-18	01:53:42	1971

Fig. 4. Accepted Data in a Database.

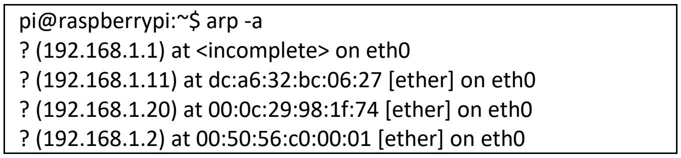
Fig. 5 shows only the ARP packet for this communication. The results show that the smart meter with MAC address Raspberry_bc:06:f9 with IP address 192.168.1.13 made a broadcast asking for the MAC address of the default gateway with the IP address 192.168.1.1. Moreover, it shows that the data collector is asking for the MAC address of the destination with the IP address of 192.168.1.13. As highlighted in Fig. 5, it is shown that the destination or the data smart meter answer the ARP request of the smart meter by giving its MAC address Raspberry_bc:06:f9. The smart meter also give ARP, a reply to its MAC address as shown in Fig. 5. Sample of ARP tables for smart meter and data collector are shown in Fig. 6 and Fig. 7.

Fig. 6 shows the ARP cache of the smart meter during normal traffic, while Fig. 7 also shows the ARP cache of the data collector when there is no attack on the network.



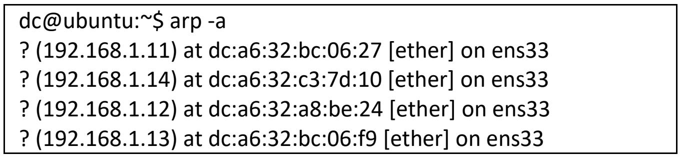
Source MAC	Destination	Dest MAC	Protocol	Length	Info
VMware_c0:00:01	Broadcast	Broadcast	ARP	60	Who has 192.168.1.0? Tell 192.168.1.2
Raspberr_bc:06:f9	Broadcast	Broadcast	ARP	42	Who has 192.168.1.1? Tell 192.168.1.13
Raspberr_bc:06:f9	Broadcast	Broadcast	ARP	42	Who has 192.168.1.1? Tell 192.168.1.13
Raspberr_bc:06:27	Broadcast	Broadcast	ARP	60	Who has 192.168.1.1? Tell 192.168.1.11
Raspberr_a8:be:24	Broadcast	Broadcast	ARP	60	Who has 192.168.1.1? Tell 192.168.1.12
VMware_98:1f:74	Raspberr_bc:06:f9	Raspberr_bc:06:f9	ARP	60	Who has 192.168.1.13? Tell 192.168.1.20
Raspberr_bc:06:f9	VMware_98:1f:74	VMware_98:1f:74	ARP	42	192.168.1.13 is at dc:a6:32:bc:06:f9
Raspberr_bc:06:27	Broadcast	Broadcast	ARP	60	Who has 192.168.1.1? Tell 192.168.1.11
Raspberr_c3:7d:10	Broadcast	Broadcast	ARP	60	Who has 192.168.1.1? Tell 192.168.1.14
Raspberr_a8:be:24	Broadcast	Broadcast	ARP	60	Who has 192.168.1.1? Tell 192.168.1.12
Raspberr_bc:06:27	Broadcast	Broadcast	ARP	60	Who has 192.168.1.1? Tell 192.168.1.11
Raspberr_c3:7d:10	Broadcast	Broadcast	ARP	60	Who has 192.168.1.1? Tell 192.168.1.14
Raspberr_a8:be:24	Broadcast	Broadcast	ARP	60	Who has 192.168.1.1? Tell 192.168.1.12

Fig. 5. ARP Reply Captured by Wireshark during Normal Traffic.



```
pi@raspberrypi:~$ arp -a
? (192.168.1.1) at <incomplete> on eth0
? (192.168.1.11) at dc:a6:32:bc:06:27 [ether] on eth0
? (192.168.1.20) at 00:0c:29:98:1f:74 [ether] on eth0
? (192.168.1.2) at 00:50:56:c0:00:01 [ether] on eth0
```

Fig. 6. ARP Cache on the Smart Meter.



```
dc@ubuntu:~$ arp -a
? (192.168.1.11) at dc:a6:32:bc:06:27 [ether] on ens33
? (192.168.1.14) at dc:a6:32:c3:7d:10 [ether] on ens33
? (192.168.1.12) at dc:a6:32:a8:be:24 [ether] on ens33
? (192.168.1.13) at dc:a6:32:bc:06:f9 [ether] on ens33
```

Fig. 7. ARP Cache on the Data Collector.

Based on the data gathered during the normal operations experiment, no anomalies were detected in Wireshark, ARP cache on the smart meter and ARP cache on the data collector. The data sent from the smart meter has the same value as the data stored in the database.

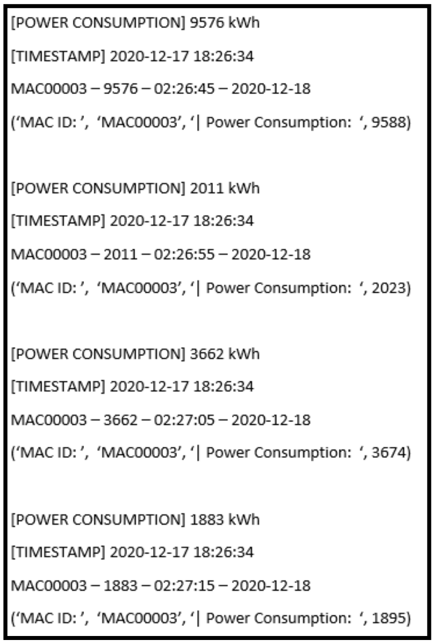
B. *Under Attack*

The smart meter with an IP address of 192.168.1.13 will send data as usual for the attack simulation. However, another Raspberry Pi will be included that will imitate the attacker for this phase. The attacker with IP address 192.168.1.11 and the corresponding MAC address of Raspberry_bc:06:27 will perform ARP spoofing on the respective smart meter and data collector in the topology. Once the attacker managed to intercept and change the power consumption value, the tampered packet will be forward back to the data collector using IPV4.

By performing ARP spoofing on a legitimate smart meter and data collector, the attacker machine will be the gateway for both of these devices. The attacker can now sniff and perform further attacks as the attacker already has access to data transferred. All communication between the smart meter and data collector now needs to go through the attacker's machine first before reaching the destination.

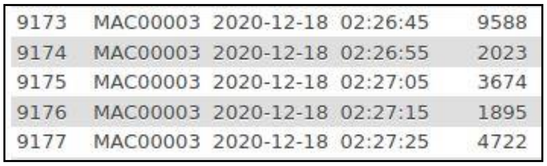
Packet manipulation script is used to change the value of power consumption. In this experiment, the power consumption is increased by 12 on every reading. The difference is shown in Fig. 8, where the data generated and sent to the data collector is not tally with Fig. 9 which displays that the data accepted by the data collector was not the legitimate data sent by the smart meter. The data in the database has been modified because the data has been intercepted and sent to the data collector by the attacker.

Fig. 10 shows the view of the attacker machine. The data from the smart meter will be intercepted, modified, and then forwarded to the destination. Fig. 10 shows that every data intercepted will be applied increment by 12. For the MITM part, this study successfully perform packet manipulation by using pattern searching tools and some modifications on the iptables to filter only the packet that needs to be modified to come through.



```
[POWER CONSUMPTION] 9576 kWh
[TIMESTAMP] 2020-12-17 18:26:34
MAC00003 – 9576 – 02:26:45 – 2020-12-18
(‘MAC ID: ’, ‘MAC00003’, ‘| Power Consumption: ‘, 9588)

[POWER CONSUMPTION] 2011 kWh
[TIMESTAMP] 2020-12-17 18:26:34
MAC00003 – 2011 – 02:26:55 – 2020-12-18
(‘MAC ID: ’, ‘MAC00003’, ‘| Power Consumption: ‘, 2023)

[POWER CONSUMPTION] 3662 kWh
[TIMESTAMP] 2020-12-17 18:26:34
MAC00003 – 3662 – 02:27:05 – 2020-12-18
(‘MAC ID: ’, ‘MAC00003’, ‘| Power Consumption: ‘, 3674)

[POWER CONSUMPTION] 1883 kWh
[TIMESTAMP] 2020-12-17 18:26:34
MAC00003 – 1883 – 02:27:15 – 2020-12-18
(‘MAC ID: ’, ‘MAC00003’, ‘| Power Consumption: ‘, 1895)
```

Fig. 8. Smart Meter Sending Data during FDIA.



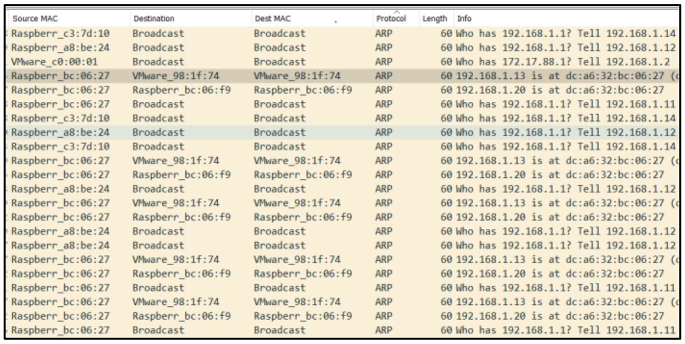
9173	MAC00003	2020-12-18	02:26:45	9588
9174	MAC00003	2020-12-18	02:26:55	2023
9175	MAC00003	2020-12-18	02:27:05	3674
9176	MAC00003	2020-12-18	02:27:15	1895
9177	MAC00003	2020-12-18	02:27:25	4722

Fig. 9. Database Accepted Falsified Data.

```
root@raspberrypi:/home/pi/Desktip# python3 testingfinal.py
[*] waiting for data

Original Data is: 9576
New Data: 9588
Payload sent!

Original Data is: 2011
New Data: 2023
Payload sent!

Original Data is: 3662
New Data: 3674
Payload sent!

Original Data is: 1883
New Data: 1895
Payload sent!

Original Data is: 4710
New Data: 4722
Payload sent!
```

Fig. 10. View on Attacker's Machine during FDIA.

The evidence captured using Wireshark is explained based on Fig. 11. Note on the highlighted line, the attacker sending a broadcast reply telling the data collector that the smart meter's MAC address is now at his MAC address which is Raspberry_bc:06:27. Also, there are presents of duplicate use in the collected evidence.



```
Source MAC          Destination         Dest MAC            Protocol  Length Info
Raspberr_c3:7d:10   Broadcast           Broadcast           ARP           60 Who has 192.168.1.1? Tell 192.168.1.14
Raspberr_a8:be:24   Broadcast           Broadcast           ARP           60 Who has 192.168.1.1? Tell 192.168.1.12
VMware_c0:00:01     Broadcast           Broadcast           ARP           60 Who has 172.17.88.1? Tell 192.168.1.2
Raspberr_bc:06:27   VMware_98:1f:74     VMware_98:1f:74     ARP           60 192.168.1.13 is at dc:a6:32:bc:06:27 (d
Raspberr_bc:06:27   Raspberr_bc:06:f9   Raspberr_bc:06:f9   ARP           60 192.168.1.20 is at dc:a6:32:bc:06:27
Raspberr_bc:06:27   Broadcast           Broadcast           ARP           60 Who has 192.168.1.1? Tell 192.168.1.11
Raspberr_c3:7d:10   Broadcast           Broadcast           ARP           60 Who has 192.168.1.1? Tell 192.168.1.14
Raspberr_a8:be:24   Broadcast           Broadcast           ARP           60 Who has 192.168.1.1? Tell 192.168.1.12
Raspberr_c3:7d:10   Broadcast           Broadcast           ARP           60 Who has 192.168.1.1? Tell 192.168.1.14
Raspberr_bc:06:27   VMware_98:1f:74     VMware_98:1f:74     ARP           60 192.168.1.13 is at dc:a6:32:bc:06:27 (d
Raspberr_bc:06:27   Raspberr_bc:06:f9   Raspberr_bc:06:f9   ARP           60 192.168.1.20 is at dc:a6:32:bc:06:27
Raspberr_a8:be:24   Broadcast           Broadcast           ARP           60 Who has 192.168.1.1? Tell 192.168.1.12
Raspberr_bc:06:27   VMware_98:1f:74     VMware_98:1f:74     ARP           60 192.168.1.13 is at dc:a6:32:bc:06:27 (d
Raspberr_bc:06:27   Raspberr_bc:06:f9   Raspberr_bc:06:f9   ARP           60 192.168.1.20 is at dc:a6:32:bc:06:27
Raspberr_a8:be:24   Broadcast           Broadcast           ARP           60 Who has 192.168.1.1? Tell 192.168.1.12
Raspberr_a8:be:24   Broadcast           Broadcast           ARP           60 Who has 192.168.1.1? Tell 192.168.1.12
Raspberr_bc:06:27   VMware_98:1f:74     VMware_98:1f:74     ARP           60 192.168.1.13 is at dc:a6:32:bc:06:27 (d
Raspberr_bc:06:27   Raspberr_bc:06:f9   Raspberr_bc:06:f9   ARP           60 192.168.1.20 is at dc:a6:32:bc:06:27
Raspberr_bc:06:27   Broadcast           Broadcast           ARP           60 Who has 192.168.1.1? Tell 192.168.1.11
Raspberr_bc:06:27   VMware_98:1f:74     VMware_98:1f:74     ARP           60 192.168.1.13 is at dc:a6:32:bc:06:27 (d
Raspberr_bc:06:27   Raspberr_bc:06:f9   Raspberr_bc:06:f9   ARP           60 192.168.1.20 is at dc:a6:32:bc:06:27
Raspberr_bc:06:27   Broadcast           Broadcast           ARP           60 Who has 192.168.1.1? Tell 192.168.1.11
```

Fig. 11. ARP Traffic Captured by Wireshark during under Attack.

Fig. 12 and Fig. 13 show the difference in ARP cache when there is no attack and under attack. During the attack, it is shown that there are two IP addresses with the MAC addresses. Supposedly, the MAC address for 192.168.1.13 was Raspberry_bc:06:f9 but after ARP spoofing, the attacker managed to link the victim's IP address to his MAC address.

```
pi@raspberrypi:~$ arp -a
? (192.168.1.1) at <incomplete> on eth0
? (192.168.1.11) at dc:a6:32:bc:06:27 [ether] on eth0
? (192.168.1.20) at 00:0c:29:98:1f:74 [ether] on eth0
? (192.168.1.2) at 00:50:56:c0:00:01 [ether] on eth0
```

Fig. 12. ARP Cache on the Smart Meter.

```
dc@ubuntu:~$ arp -a
? (192.168.1.11) at dc:a6:32:bc:06:27 [ether] on ens33
? (192.168.1.14) at dc:a6:32:c3:7d:10 [ether] on ens33
? (192.168.1.12) at dc:a6:32:a8:be:24 [ether] on ens33
? (192.168.1.13) at dc:a6:32:bc:06:27 [ether] on ens33
```

Fig. 13. ARP Cache on the Data Collector.

Based on the data gathered during the under attack experiment, anomalies were detected in Wireshark, ARP cache on the smart meter, and ARP cache on the data collector. The data sent from the smart meter has a different value from the data stored in the database as it was changed by the attacker.

C. Forensic Analysis

In this section, the PCAP file that stored all the digital evidence was extracted and analyzed. The analysis and comparison of the collected evidence in this study are used for in-depth analysis. Fig. 14 shows the steps taken during the forensic analysis.

The analysis process begins by collecting packets captured using Wireshark from the client during normal traffic and during under attack. The packets are also collected from the data collector during normal traffic and during the network under attack. The records from the normal traffic phase will be used as a benchmark for comparative analysis to investigate the FDIA in AMI.

Fig. 15 shows that Wireshark captured another MAC address (bc:06:27). In addition, Fig. 16 shows the use of duplicate IP addresses was reported. This strengthens the evidence collected, as shown in Fig. 12 and Fig. 13. It could be observed that the IP address 192.168.137.13, which was earlier known to be the IP address of the smart meter, now has two MAC bindings: Raspberry_bc:06:f9 (initial MAC address), and Raspberry_bc:06:27 (owned by the attacker machine in the network), which was the outcome of ARP poisoning/spoofing.

Fig. 16 shows that the time to live (TTL) of the packet from the client was 64 (left), and it was still 64 when it reached the data collector (right). This is normal as there is no router involved in this topology. However, Fig. 17 shows that TTL is different when the data was sent from the smart meter (left) and when it was accepted at the data collector (right) during the network was under attack.

As shown in Fig. 17, when the data was sent out from the smart meter (left), the TTL was 64 but when it reached the database, the TTL of the packet was 63, indicate that the packet had travel somewhere else before reached the data collector. The normal topology is assumed that the smart meter should directly deliver data to the data collector with a switch and not include a router, so it should not modify the TTL of the packet. This happens because the attacker intercepted the packet and modified the packet's data before forwarding it to the real destination.

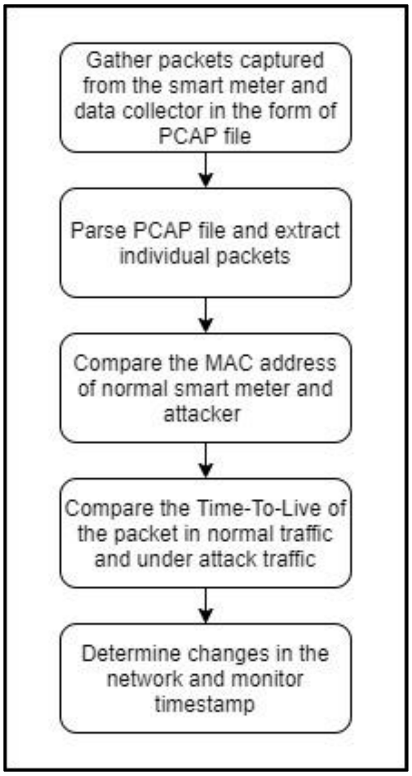
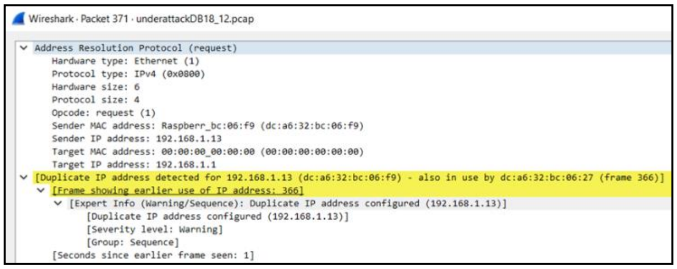

Gather packets captured from the smart meter and data collector in the form of PCAP file
Parse PCAP file and extract individual packets
Compare the MAC address of normal smart meter and attacker
Compare the Time-To-Live of the packet in normal traffic and under attack traffic
Determine changes in the network and monitor timestamp

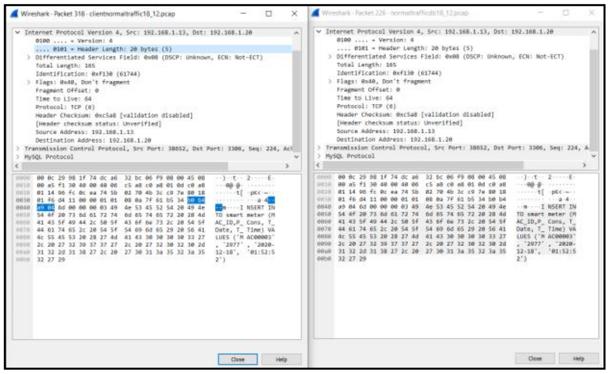
Fig. 14. Forensic Analysis Flow Chart



Wireshark · Packet 371 · underattackDB18_12.pcap

```
v Address Resolution Protocol (request)
      Hardware type: Ethernet (1)
      Protocol type: IPv4 (0x0800)
      Hardware size: 6
      Protocol size: 4
      Opcode: request (1)
      Sender MAC address: Raspberr_bc:06:f9 (dc:a6:32:bc:06:f9)
      Sender IP address: 192.168.1.13
      Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
      Target IP address: 192.168.1.1
v [Duplicate IP address detected for 192.168.1.13 (dc:a6:32:bc:06:f9) - also in use by dc:a6:32:bc:06:27 (frame 366)]
  v [Frame showing earlier use of IP address: 366]
    v [Expert Info (Warning/Sequence): Duplicate IP address configured (192.168.1.13)]
          [Duplicate IP address configured (192.168.1.13)]
          [Severity level: Warning]
          [Group: Sequence]
      [Seconds since earlier frame seen: 1]
```

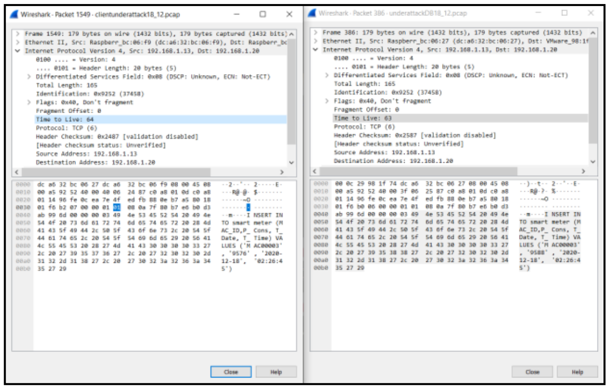
Fig. 15. Use of Duplicate IP Address in MAC-IP Address Binding.



```
Wireshark · Packet 318 · clientnormaltraffic18_12.pcap
v Internet Protocol Version 4, Src: 192.168.1.13, Dst: 192.168.1.20
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: Unknown, ECN: Not-ECT)
    Total Length: 165
    Identification: 0xf130 (61744)
  > Flags: 0x40, Don't fragment
    Fragment Offset: 0
    Time to Live: 64
    Protocol: TCP (6)
    Header Checksum: 0xc5a8 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.1.13
    Destination Address: 192.168.1.20
> Transmission Control Protocol, Src Port: 38652, Dst Port: 3306, Seq: 224, Ack
> MySQL Protocol

0000  00 0c 29 98 1f 74 dc a6  32 bc 06 f9 08 00 45 08   ··)··t·· 2·····E·
0010  00 a5 f1 30 40 00 40 06  c5 a8 c0 a8 01 0d c0 a8   ···0@·@· ········
0020  01 14 96 fc 0c ea 74 5b  02 70 4b 3c c9 7e 80 18   ······t[ ·pK<·~··
0030  01 f6 d4 11 00 00 01 01  08 0a 7f 61 b5 34 b0 b4   ········ ···a·4··
0040  a9 04 6d 00 00 00 03 49  4e 53 45 52 54 20 49 4e   ··m····I NSERT IN
0050  54 4f 20 73 6d 61 72 74  6d 65 74 65 72 20 28 4d   TO smart meter (M
0060  41 43 5f 49 44 2c 50 5f  43 6f 6e 73 2c 20 54 5f   AC_ID,P_ Cons, T_
0070  44 61 74 65 2c 20 54 5f  54 69 6d 65 29 20 56 41   Date, T_ Time) VA
0080  4c 55 45 53 20 28 27 4d  41 43 30 30 30 30 33 27   LUES ('M AC00003'
0090  2c 20 27 32 39 37 37 27  2c 20 27 32 30 32 30 2d   , '2977' , '2020-
00a0  31 32 2d 31 38 27 2c 20  27 30 31 3a 35 32 3a 35   12-18',  '01:52:5
00b0  32 27 29                                           2')
```

```
Wireshark · Packet 226 · normaltrafficdb18_12.pcap
v Internet Protocol Version 4, Src: 192.168.1.13, Dst: 192.168.1.20
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x08 (DSCP: Unknown, ECN: Not-ECT)
    Total Length: 165
    Identification: 0xf130 (61744)
  > Flags: 0x40, Don't fragment
    Fragment Offset: 0
    Time to Live: 64
    Protocol: TCP (6)
    Header Checksum: 0xc5a8 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.1.13
    Destination Address: 192.168.1.20
> Transmission Control Protocol, Src Port: 38652, Dst Port: 3306, Seq: 224, A
> MySQL Protocol

0000  00 0c 29 98 1f 74 dc a6  32 bc 06 f9 08 00 45 08   ··)··t·· 2·····E·
0010  00 a5 f1 30 40 00 40 06  c5 a8 c0 a8 01 0d c0 a8   ···0@·@· ········
0020  01 14 96 fc 0c ea 74 5b  02 70 4b 3c c9 7e 80 18   ······t[ ·pK<·~··
0030  01 f6 d4 11 00 00 01 01  08 0a 7f 61 b5 34 b0 b4   ········ ···a·4··
0040  a9 04 6d 00 00 00 03 49  4e 53 45 52 54 20 49 4e   ··m····I NSERT IN
0050  54 4f 20 73 6d 61 72 74  6d 65 74 65 72 20 28 4d   TO smart meter (M
0060  41 43 5f 49 44 2c 50 5f  43 6f 6e 73 2c 20 54 5f   AC_ID,P_ Cons, T_
0070  44 61 74 65 2c 20 54 5f  54 69 6d 65 29 20 56 41   Date, T_ Time) VA
0080  4c 55 45 53 20 28 27 4d  41 43 30 30 30 30 33 27   LUES ('M AC00003'
0090  2c 20 27 32 39 37 37 27  2c 20 27 32 30 32 30 2d   , '2977' , '2020-
00a0  31 32 2d 31 38 27 2c 20  27 30 31 3a 35 32 3a 35   12-18',  '01:52:5
00b0  32 27 29                                           2')
```

Fig. 16. TTL of the Packet during Normal Traffic from the Smart Meter (Left) and the Data Collector (Right).



```
Wireshark · Packet 1549 · clientunderattack18_12.pcap
> Frame 1549: 179 bytes on wire (1432 bits), 179 bytes captured (1432 bits)
> Ethernet II, Src: Raspberr_bc:06:f9 (dc:a6:32:bc:06:f9), Dst: Raspberr_bc
v Internet Protocol Version 4, Src: 192.168.1.13, Dst: 192.168.1.20
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x08 (DSCP: Unknown, ECN: Not-ECT)
    Total Length: 165
    Identification: 0x9252 (37458)
  > Flags: 0x40, Don't fragment
    Fragment Offset: 0
    Time to Live: 64
    Protocol: TCP (6)
    Header Checksum: 0x2487 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.1.13
    Destination Address: 192.168.1.20

0000  dc a6 32 bc 06 27 dc a6  32 bc 06 f9 08 00 45 08   ··2··'·· 2·····E·
0010  00 a5 92 52 40 00 40 06  24 87 c0 a8 01 0d c0 a8   ···R@·@· $·······
0020  01 14 96 fe 0c ea 7e 4f  ed fb 88 0e b7 a5 80 18   ······~O ········
0030  01 f6 b2 07 00 00 01 01  08 0a 7f 80 b7 e6 b0 d3   ········ ········
0040  ab 99 6d 00 00 00 03 49  4e 53 45 52 54 20 49 4e   ··m····I NSERT IN
0050  54 4f 20 73 6d 61 72 74  6d 65 74 65 72 20 28 4d   TO smart meter (M
0060  41 43 5f 49 44 2c 50 5f  43 6f 6e 73 2c 20 54 5f   AC_ID,P_ Cons, T_
0070  44 61 74 65 2c 20 54 5f  54 69 6d 65 29 20 56 41   Date, T_ Time) VA
0080  4c 55 45 53 20 28 27 4d  41 43 30 30 30 30 33 27   LUES ('M AC00003'
0090  2c 20 27 39 35 37 36 27  2c 20 27 32 30 32 30 2d   , '9576' , '2020-
00a0  31 32 2d 31 38 27 2c 20  27 30 32 3a 32 36 3a 34   12-18',  '02:26:4
00b0  35 27 29                                           5')
```

```
Wireshark · Packet 386 · underattackDB18_12.pcap
> Frame 386: 179 bytes on wire (1432 bits), 179 bytes captured (1432 bits)
> Ethernet II, Src: Raspberr_bc:06:27 (dc:a6:32:bc:06:27), Dst: VMware_98:1f
v Internet Protocol Version 4, Src: 192.168.1.13, Dst: 192.168.1.20
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x08 (DSCP: Unknown, ECN: Not-ECT)
    Total Length: 165
    Identification: 0x9252 (37458)
  > Flags: 0x40, Don't fragment
    Fragment Offset: 0
    Time to Live: 63
    Protocol: TCP (6)
    Header Checksum: 0x2587 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.1.13
    Destination Address: 192.168.1.20

0000  00 0c 29 98 1f 74 dc a6  32 bc 06 27 08 00 45 08   ··)··t·· 2··'··E·
0010  00 a5 92 52 40 00 3f 06  25 87 c0 a8 01 0d c0 a8   ···R@·?· %·······
0020  01 14 96 fe 0c ea 7e 4f  ed fb 88 0e b7 a5 80 18   ······~O ········
0030  01 f6 b0 06 00 00 01 01  08 0a 7f 80 b7 e6 b0 d3   ········ ········
0040  ab 99 6d 00 00 00 03 49  4e 53 45 52 54 20 49 4e   ··m····I NSERT IN
0050  54 4f 20 73 6d 61 72 74  6d 65 74 65 72 20 28 4d   TO smart meter (M
0060  41 43 5f 49 44 2c 50 5f  43 6f 6e 73 2c 20 54 5f   AC_ID,P_ Cons, T_
0070  44 61 74 65 2c 20 54 5f  54 69 6d 65 29 20 56 41   Date, T_ Time) VA
0080  4c 55 45 53 20 28 27 4d  41 43 30 30 30 30 33 27   LUES ('M AC00003'
0090  2c 20 27 39 35 38 38 27  2c 20 27 32 30 32 30 2d   , '9588' , '2020-
00a0  31 32 2d 31 38 27 2c 20  27 30 32 3a 32 36 3a 34   12-18',  '02:26:4
00b0  35 27 29                                           5')
```

Fig. 17. TTL of the Packet during under Attack from the Smart Meter (Left) and the Data Collector (Right).

As displayed in Fig. 18, the time taken for the data collector to respond to the smart meter when the smart meter wants to send data is much lower than when under attack. Fig. 19 shows the average time taken by the data collector to respond when the smart meter wants to send data. This shows a huge gap of time taken for the data collector to reply during normal traffic, and the network was under attack. It can be concluded that the delay that occurred during the network was under attack is caused by the path and process that happened on the data to reach the destination. The data went through a longer path and was processed by the attacker first before the data was forwarded back to the destination.

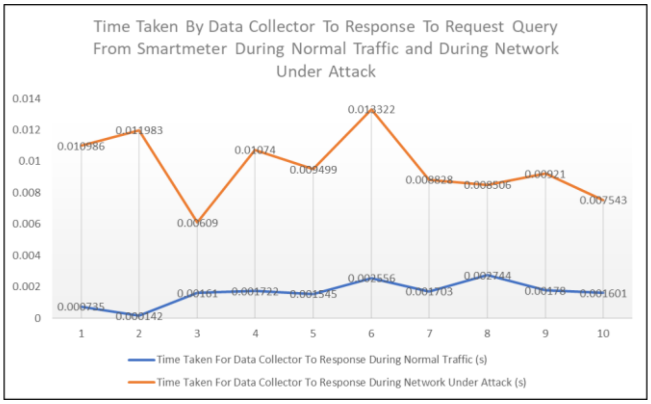


Fig. 18. Comparison of the Time Taken for the Data Collector to Respond to the Request Query from the Smart Meter during Normal Traffic and under Attack.

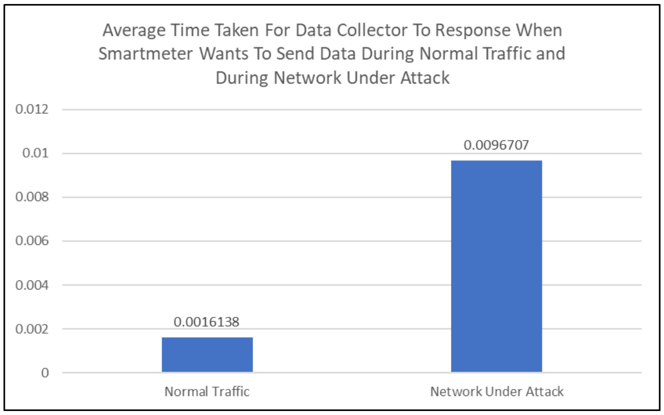


Fig. 19. The Average Data Collector Response Time.

TABLE Ishows a list of attributes that forensic investigators can use as references on what attributes of data to be collected to perform forensic analysis in tracing FDIA. The list can be used as a reference for forensic investigators to perform evidence collection during FDIA investigations.

TABLE I. TABLE OF ATTRIBUTES

Attributes	Description
SrcIp	Source IP address
SrcPort	Source port address
DstIp	Destination IP address
DstPort	Destination port address
SrcMac	Source MAC address
DstMac	Destination MAC address
TTL	Time to live of the packets
ARPReq	ARP request traffic
ARPRep	ARP reply traffic
TimeDelay	Time delay for the client to receive a reply from the server

Based on the forensic analysis conducted, changes were detected in Wireshark such as a single IP address having two different MAC addresses, one MAC address belongs to the normal smart meter and the other MAC address belonging to the attacker. Other changes that were detected are in the TTL of the packet and the time taken for the data collector to respond to the request query.

V. CONCLUSION

This study's primary motivation was to study FDIA impact in the IoT environment and perform forensics analysis on digital traces from data obtained. Based on the data obtained from the experiments, the proposed list of attributes for forensic analysis could be useful to trace FDIA. In future works, there is a need to explore different types of attacks, such as buffer overflow payloads that may results in a system crash, creating a path for the hackers to initiate their malicious actions. Future studies may also focus on the integration of forensic-by-design principles in the design of any critical system because it will be quite difficult to know what has happened if there is no log or no proof. If the system is able to produce a series of events, it would be very helpful for the forensic investigator to reconstruct the events in order to identify available sources and different types of potential evidence in such cases. Therefore, another potential study could explore how to integrate forensic-by-design principles in the design of such systems.

ACKNOWLEDGMENT

This study was funded by Tenaga Nasional Berhad Seed Fund (U-TD-RD-19-24) in collaboration with TNB Asset Management Department. This research team would like to thank UNITEN R&D Sdn. Bhd. for fund management.

REFERENCES

[1] P. Brous, M. Janssen, and P. Herder, “The dual effects of the Internet of Things (IoT): A systematic review of the benefits and risks of IoT adoption by organizations,” Int. J. Inf. Manage., vol. 51, no. May 2019, p. 101952, 2020, doi: 10.1016/j.ijinfomgt.2019.05.008.

[2] I. Lee and K. Lee, “The Internet of Things (IoT): Applications, investments, and challenges for enterprises,” Bus. Horiz., vol. 58, no. 4, pp. 431–440, 2015, doi: 10.1016/j.bushor.2015.03.008.

[3] P. Alto, “Impacts of Cyberattacks on IoT Devices,” Impacts Cyberattacks IoT Devices, 2019.

[4] S. Sharma, M. Manuja, and K. Kishore, “Vulnerabilities, attacks and their mitigation: An implementation on internet of things (IoT),” Int. J. Innov. Technol. Explor. Eng., vol. 8, no. 10, pp. 146–150, 2019, doi: 10.35940/ijitee.F3761.0881019.

[5] M. Kuzlu, C. Fair, and O. Guler, “Role of Artificial Intelligence in the Internet of Things (IoT) cybersecurity,” Discov. Internet Things, vol. 1, no. 1, 2021, doi: 10.1007/s43926-020-00001-4.

[6] W. A. Dimitrov and G. S. Panayotova, “The impacts of dns protocol security weaknesses,” J. Commun., vol. 15, no. 10, pp. 722–728, 2020, doi: 10.12720/jcm.15.10.722-728.

[7] M. Ahmed and A. S. K. Pathan, “False data injection attack (FDIA): an overview and new metrics for fair evaluation of its countermeasure,” Complex Adapt. Syst. Model., vol. 8, no. 1, 2020, doi: 10.1186/s40294-020-00070-w.

[8] D. B. Rawat and C. Bajracharya, “Detection of false data injection attacks in smart grid communication systems,” IEEE Signal Process. Lett., vol. 22, no. 10, pp. 1652–1656, 2015, doi: 10.1109/LSP.2015.2421935.

[9] D. Allison, P. Smith, K. Mclaughlin, F. Zhang, J. Coble, and R. Busquim, “PLC-Based Cyber-Attack Detection: A Last Line of Defence,” Int. Conf. Nucl. Secur. Sustain. Strength. Efforts, pp. 1–10, 2020.

[10] S. Bhattacharjee, A. Thakur, S. Silvestri, and S. K. Das, “Statistical security incident forensics against data falsification in smart grid advanced metering infrastructure,” CODASPY 2017 - Proc. 7th ACM Conf. Data Appl. Secur. Priv., pp. 35–45, 2017, doi: 10.1145/3029806.3029833.

[11] O. Abu Waraga, “Design and Implementation of an Automated IoT security testbed.” 2019.

[12] V. Sachidananda, J. Toh, S. Siboni, S. Bhairav, A. Shabtai, and Y. Elovici, “Let the cat out of the bag: A holistic approach towards security analysis of the internet of things,” IoTPTS 2017 - Proc. 3rd ACM Int. Work. IoT Privacy, Trust. Secur. co-located with ASIA CCS 2017, pp. 3–10, 2017, doi: 10.1145/3055245.3055251.

[13] M. L. Hale, K. Lotfy, R. F. Gamble, C. Walter, and J. Lin, “Developing a platform to evaluate and assess the security of wearable devices,” Digit. Commun. Networks, vol. 5, no. 3, pp. 147–159, 2019, doi: 10.1016/j.dcan.2018.10.009.