7-1-2023

DoS/DDoS-MQTT-IoT: A dataset for evaluating intrusions in IoT networks using the MQTT protocol

Alaa Alatram
Edith Cowan University

Leslie F. Sikos
Edith Cowan University

Mike Johnstone
Edith Cowan University

Patryk Szewczyk
Edith Cowan University

James Jin Kang
Edith Cowan University

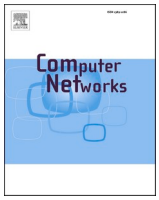
10.1016/j.comnet.2023.109809
Alatram, A., Sikos, L. F., Johnstone, M., Szewczyk, P., & Kang, J. J. (2023). DoS/DDoS-MQTT-IoT: A dataset for evaluating intrusions in IoT networks using the MQTT protocol. Computer Networks, 231, article 109809. https://doi.org/10.1016/j.comnet.2023.109809

Computer Networks 231 (2023) 109809

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet



Data Article

DoS/DDoS-MQTT-IoT: A dataset for evaluating intrusions in IoT networks using the MQTT protocol

Alaa Alatram, Leslie F. Sikos *, Mike Johnstone, Patryk Szewczyk, James Jin Kang

Edith Cowan University, WA, Australia

ARTICLE INFO

Keywords:
IoT
MQTT
DoS
DDoS
Cybersecurity Dataset
Machine learning

ABSTRACT

Adversaries may exploit a range of vulnerabilities in Internet of Things (IoT) environments. These vulnerabilities are typically exploited to carry out attacks, such as denial-of-service (DoS) attacks, either against the IoT devices themselves, or using the devices to perform the attacks. These attacks are often successful due to the nature of the protocols used in the IoT. One popular protocol used for machine-to-machine IoT communications is the Message Queueing Telemetry Protocol (MQTT). Countermeasures for attacks against MQTT include testing defenses with existing datasets. However, there is a lack of real-world test datasets in this area. For this reason, this paper introduces a DoS/DDoS-MQTT-IoT dataset—that contains various DoS/DDoS attack scenarios using MQTT traffic—to help develop and test countermeasures against such attacks. To this end, a physical IoT testbed was constructed and a large volume of IoT data was generated that included standard MQTT traffic as well as 10 DoS scenarios. The usability of the dataset has been evaluated via machine learning.

1. Introduction

The Internet of Things (IoT) constitutes a range of devices, including, but not limited to, IP cameras, smart vehicles, surveillance technologies, and wearable devices. All these devices are designed to be convenient and improve the users' quality of life. IoT devices should be constantly available, make use of cloud services, and operate using wireless protocols. The plethora of IoT standards, protocols, and devices has contributed to the evolution of smart cities, in which multiple devices communicate across networks. This diversity introduced numerous security issues [1]. The emergence of heterogeneous technologies necessitated the introduction of new communication protocols for transferring data; one example of which is the *Message Queuing Telemetry Transport (MQTT)* protocol [12].

MQTT was designed to be lightweight and efficient in high-latency, unreliable networks, making it ideal for IoT devices. However, the MQTT protocol is vulnerable to a number of cyber threats, one of which is denial of service (DoS) [10]. DoS attacks have evolved and become more sophisticated due to an increase in the number of comprised IoT devices [3]. Vulnerabilities that can be exploited to execute a DoS attack on the MQTT protocol include limited payload size and QoS levels. DoS attacks may also exploit an operating system's Transmission Control Protocol (TCP) implementation, and in turn, MQTT (which is based on TCP/IP) [4].

This paper presents the outcomes of our physical IoT testbed, the MQTT network configuration, IoT data generation, and the evaluation of the dataset using a selection of conventional machine learning techniques often used within intrusion detection systems (IDS). Section II presents related works. Section III describes our physical IoT testbed. Section IV presents the outcome of using the testbed (i.e., the dataset). Section V discusses the experiments and results of the classifier algorithms. The conclusion and future work are presented in Section VI.

2. Related works

Datasets for testing intrusion detection algorithms have been published for over two decades (see Table 1). Detection algorithms have progressively evolved; new protocols have been developed and topologies have advanced. There are numerous imperative features of an effective IDS. The first one is to have real-world test data, because algorithms tested with synthetic data tend to perform poorly when faced with the volume and variety of actual network traffic [6]. During collection, raw data needs to be collected for training and testing, through a single or multiple algorithms. The quality of the data used to train a model(s) determines the effectiveness of the developed model(s). If the training datasets include quality samples of, benign and malicious

* Corresponding author.
E-mail address: l.sikos@ecu.edu.au (L.F. Sikos).

https://doi.org/10.1016/j.comnet.2023.109809
Received 20 March 2023; Received in revised form 2 May 2023; Accepted 3 May 2023
Available online 18 May 2023

traffic (e.g., samples that demonstrate DoS attacks), then the machine learning algorithms will be effective in detecting attacks. This is why capturing real-world network traffic labelled for supervised learning techniques and containing both benign and malicious scenarios is imperative. Note that full packet capture is preferred so that new features can be generated [7]. As outlined by Praseed and Thilagam [11], researchers have difficulty obtaining datasets with application layer DoS traffic. Table 1 compares the commonly used online datasets for testing intrusion detection algorithms with the proposed dataset.

The DARPA9, KDD99 and CAIDA datasets are legacy datasets that may not reflect the complexity and heterogeneity typical to modern-day network traffic. The DEFCON, LBNL, UNSW-NB15, ISCX, and CICIDS 2017 datasets do not encompass contemporary IoT data. In contrast, Bot-IoT [7] captures IoT data in a realistic network environment encompassing both normal and botnet traffic with more than 72 million records. The Bot-IoT dataset includes DoS and DDoS attacks with protocols including TCP, UDP, and HTTP. However, the Bot-IoT dataset does not contain any occurrences of attacks on the MQTT protocol. The MQTTset dataset covers a broad range of attacks, namely, DoS, MQTT Publish flood, SlowITe, malformed data, and brute force attacks [15]. The MQTT-IoT-IDS2020 dataset contains; aggressive scanning, UDP scanning, Sparta SSH brute-force, and MQTT brute-force attacks [5].

The contribution of this paper is centered around variations of DoS attacks against the MQTT protocol. The abnormal data comprising five scenarios for each DoS and DDoS attack are the following:

- CONNECT Flooding Attack (BF_DoS),[1]
- CONNECT Flooding Attack (BF_DDoS),
- Delayed CONNECT Flooding Attack (Delay_DoS),[2]
- Delayed CONNECT Flooding Attack (Delay_DDoS),
- Invalid Subscription Flooding Attack (Sub_DoS),[3]
- Invalid Subscription Flooding Attack (Sub_DDoS),
- CONNECT Flooding with WILL Payload Attack (WILL_DoS),[4]
- CONNECT Flooding with WILL Payload Attack (WILL_DDoS),
- TCP SYN Flooding Attack (SYN_DoS),[5]
- TCP SYN Flooding Attack (SYN_DDoS).

3. The IoT testbed

Fig. 1 depicts the IoT testbed purposefully developed to generate the dataset. The topology for this testbed was constructed to facilitate the use of the MQTT protocol and generate a realistic IoT dataset focusing on DoS/DDoS attacks. First, a variety of physical IoT sensors collected actual sensor data, such as temperature and pressure, and communicated this data using Raspberry Pi's (publishers) through the MQTT protocol, sending the collected data to the MQTT Broker, and subsequently to Raspberry Pi's (subscribers). Second, attacker machines were used to create ten DoS/DDoS scenarios against the MQTT protocol. This process enabled both legitimate and attack traffic data to be generated and thus collected.

3.1. Instrumentation

This section describes the resources used to generate the DoS/DDoS-MQTT-IoT dataset. The physical sensors used for the physical IoT testbed are presented in Table 2. These sensors were used to build a realistic IoT network.

The sensors were attached to sixteen Raspberry Pi 3 Model *B*+ SBCs running Lubuntu (henceforth referred to as "R-Pi"). Both digital and analog sensors were used, measuring various physical properties, including temperature, humidity, voltage, water level, carbon monoxide level, vibration, smoke, flame, motion, touch, light, buzzer, sound, and barometric pressure. All sensors were digital, except for the water and the voltage sensors, which used the analog MCP3008 ADC converter.

Table 1
Comparison of online datasets.

Dataset	Configuration of a realistic testbed	Realistic traffic data	Dataset with labels	IoT data	MQTT attack data	MQTT DoS/DDoS attack data
DARPA99 [8]	+	–	+	–	–	–
KDD99 [14]	+	–	+	–	–	–
CAIDA [2]	+	+	–	–	–	–
DEFCON [13]	–	–	–	–	–	–
LBNL [13]	–	+	–	–	–	–
UNSW-NB15 [9]	+	+	+	–	–	–
ISCX [13]	+	+	+	–	–	–
CICIDS 2017 [13]	+	+	+	–	–	–
Bot-IoT [7]	+	+	+	+	–	–
MQTTset [15]	+	+	+	+	+	–
MQTT-IoT-IDS2020 Dataset [5]	+	+	+	+	+	–
Proposed Dataset (DoS/DDoS-MQTT-IoT)[11]	+	+	+	+	+	+

[11] https://www.kaggle.com/datasets/alaaalatram/dosddos-mqtt-iot?datasetId=3157,581.

The sixteen R-Pies were configured as publishers, with two groups, of 8 publishers connected to a wireless access point in a separate network, as illustrated in Fig. 1. Two subscribers were assigned to each of the 8 publishers.

The hardware for implementing the physical IoT testbed is shown in Table 3. Each Raspberry Pi was powered by a 5 V/2.5A Micro USB power

[1] A large volume of requests of CONNECT packets are sent to the target machine to overwhelm it during the authentication requests.

[2] This forces a certain period of delay between the TCP three-way handshake and the first packet regarding the MQTT that is a CONNECT packet. Subsequently, this opens many TCP sessions and necessitates waiting for the CONNECT packet. Because the processor will match the credentials sent by the CONNECT packet, the CPU usage is increased.

[3] With valid credentials, an attacker can either overwhelm the server (broker) using invalid subscriptions or publish requests to the subscriber, resulting in the consumption of the broker's CPU resources.

[4] By sending a CONNECT packet for authentication requests, an attacker can consume the broker's CPU resources via increasing the CONNECT packet size through piggy-backing a WILL payload.

[5] Attacks exploit the TCP protocol's state retention mechanism. This attack exploits the three-way handshake technique of the TCP protocol, leading to open the maximum number of simultaneous TCP half-connections.

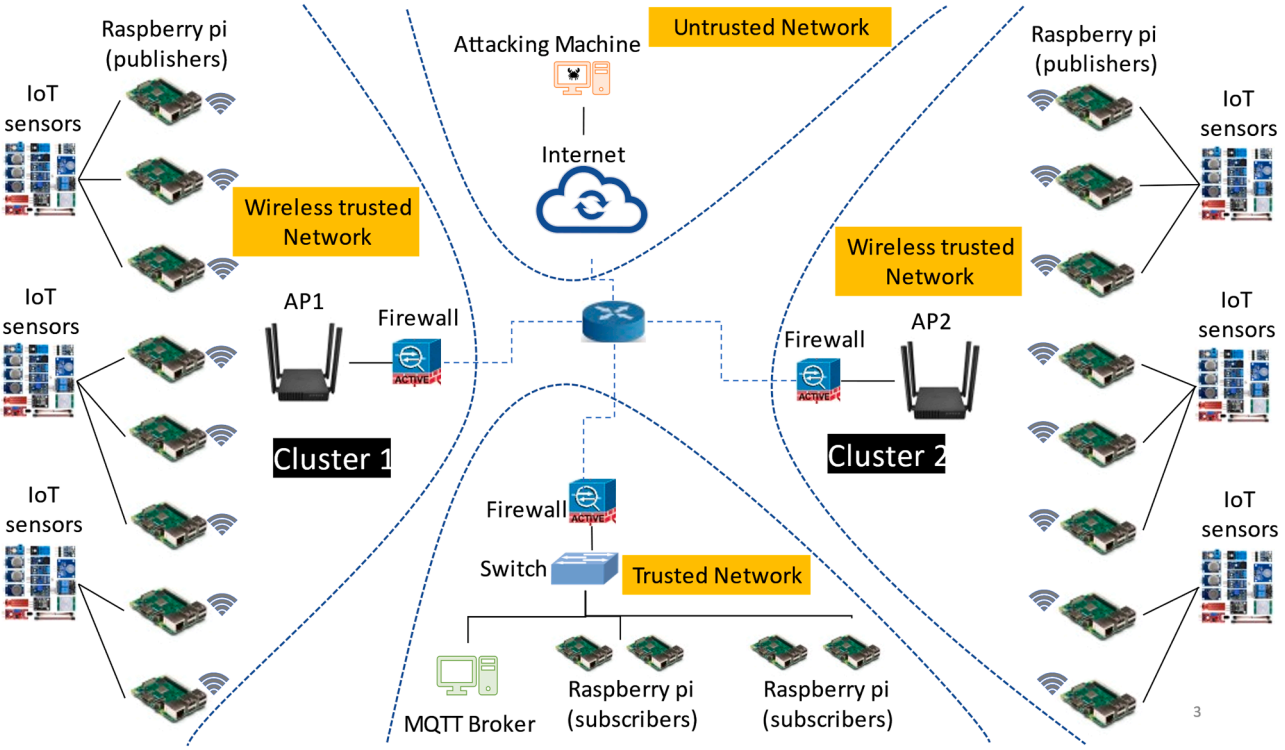


Fig. 1. The Network Topology of the Physical IoT Testbed Used to Generate the Proposed Dataset.

Table 2
Sensors Used in the Physical IoT Testbed.

Sensors	Model	Quantity
Voltage Detection Sensor	PHI1071219	4
Gas Smoke Sensor	MQ-2	4
CO Carbon Monoxide Sensor	MQ-7	4
Flame Detection Sensor	HCARDU0024	4
Temperature & Humidity Sensor	DHT11	4
Digital Barometric Pressure Sensor	BMP180	4
Digital Touch Sensor	TTP223B	4
Photosensitive Light Sensor	LM393	4
Vibration Sensor	SW420	4
Sound Detection Sensor	KY-037	4
Buzzer Alarm Sensor	AA117	4
Infrared RIP Motion Sensor	HC-SR501	4
Water Level Sensor	TA0165	4

supply, and its operating system was stored on a SanDisk Ultra 16GB Ultra Micro SDHC UHS-I/Class 10 card.

Table 4 describes the software tools used. *Lubuntu*[6] was installed on all R-Pis and an algorithm has been implemented in Python to read data from the sensors and send data to the MQTT broker, as well as to receive data on the MQTT subscribers. The MQTT broker, a *VMware Workstation*,[7] and a scientific Python distribution (*Anaconda*)[8] have been installed on Windows. VMware Workstation was used to set up the attack machines to generate DoS and DDoS attacks. *Google Colaboratory*[9] facilitated the execution of Python code through a browser, allowing five code instances to be executed simultaneously.

[6] https://lubuntu.me
[7] https://www.vmware.com/content/vmware/vmware-published-sites/us/products/workstation-pro.html
[8] https://www.anaconda.com
[9] https://colab.research.google.com

3.2. Network design

Once the sensors have been connected to the R-Pies (i.e., publishers), the other equipment was set up, which included the other 4 R-Pies (i.e., subscribers), the MQTT broker, two access points, the Cisco switch, the Cisco router, and the attack machine. The network addresses associated with each network in the physical testbed are shown in Table 5.

3.3. Cisco router and switch configurations

The Cisco router connected all networks in the physical topology. There was a total of six networks, as shown in Table 5. A Cisco switch was used to configure three networks: the MQTT broker, two R-Pies that were set up as subscribers for the 8 publishers. The remaining two R-Pies were set up as subscribers for the 8 publishers, as shown in Fig. 3. Virtual local area networks (VLAN) were used to configure three separate networks.

3.4. MQTT broker configuration

The Eclipse Mosquitto broker was selected for its popularity as a server used for the MQTT protocol. It was installed on a separate laptop. To create a realistic testbed, 1000 credentials have been created and configured in the Mosquitto broker, and a wide range of access control list commands were written.

3.5. Programming the publishers

To control the sensors and publishers, algorithms have been implemented in Python, reading data from sensors and sending messages to the MQTT broker via the R-Pies. In brief, the sensors were connected as a cluster to an R-Pi, and each sensor read and sent data it to its associated R-Pi. Consequently, the R-Pi was configured as a publisher whose primary responsibility was to send messages to the MQTT broker during a specific period of time. 11 algorithms had been developed for the various sensors used in the clusters, to create realistic scenarios for

Table 3
Hardware Equipment Used for the Physical IoT Testbed.

Equipment	Specification	Role	Quantity
Raspberry Pi3 Model B+	Processor: 1.4 GHz 64-bit quad-core processor. RAM: 1GB LPDDR2 SDRAM. USB: 4 USB 2.0 ports. HDMI: Full-size HDMI. Ethernet: Gigabit Ethernet over USB 2.0. Wireless: 2.4 GHz and 5 GHz IEEE 802.11.b/g/n/ac wireless LAN. Bluetooth: 4.2, BLE. Power supply: 5 V/2.5A DC via micro-USB connector.	16 R-Pi's were connected with the sensors, and they were used as publishers to publish messages from the surroundings to the MQTT broker. 4 R-Pi's were used as subscribers to get the published messages during the MQTT broker.	20
Breadboard	10 pcs	Breadboards are used to connect sensors to the raspberries.	10
Archer c54 Wi-Fi router	Dual Band Gigabit & Access point	Each access point was connected with its cluster that includes 8 publishers.	2
Cisco Router	Cisco 2600	This router was connected to all the different networks.	1
Cisco Switch	Catalyst 2960	This switch has been connected to the subscribers and the MQTT broker.	1
MQTT Broker (Laptop)	Processor: Intel® Core™ 2 Extreme CPU Q9300, 2.53 GHz (4 CPUs) RAM: 4GB Operating System: Windows 10 Pro 64-bit	This device was set up as an MQTT broker (i.e., server) to receive messages from publishers and transmit them to subscribers.	1
Attack machine (Laptop)	Processor: Intel Core i7–4500 U CPU, 180 GHz (4CPUs) RAM: 8GB Operating System: Windows 10 Pro 64-bit	This device was set up as an attack machine to generate 10 DoS/DDoS attack scenarios.	1

Table 4
Software Resources.

Name	Version	Role
Linux (Lubuntu)	20.04	Operating system used for the R-Pis
Windows	10 Pro	Operating system used for the broker
VMware Workstation	15.5.6	Virtualisation software used to set up the attack machine
Kali Linux	2020.3	A Linux distribution designed used to launch DoS and DDoS attacks
Hping3	3.0.0	A tool used to launch the SYN flooding attack
Putty	0.74	A telnet/SSH client used for Cisco router and switch configuration
Eclipse Mosquitto[11]	1.6.2	A message broker that implements between publishers and subscribers using the MQTT protocol
Wireshark	3.2.7	A packet analyser used for raw data processing
Tshark[12]	3.2.7	A packet analyser used for raw data processing
Python 3	3.6.9	A high-level programming language used for; DoS/DDoS attacks, data analysis, IoT sensors, creating usernames/passwords, and for the optimiser
Anaconda (spider)	5.0.5	Python package distribution installer
Colaboratory (Google)	N/A	A Google data analysis utility that allows writing and executing of Python code within a browser
Excel	16	A spread sheet for processing and cleaning the dataset
Raspberry Pi Imager	1.6.1	A tool used to install Linux onto MicroSD

[11] https://mosquitto.org.
[12] https://tshark.dev.

gathering data from the surrounding environments. For example, Algorithm 1 is a water level sensor that measures the level of water in a tank and gives periodic readings, as shown in Fig. 2.

The 11 algorithms that were implemented for the sensors were the following:

- Water level detection for tank,
- Reading voltage for solar power system,
- Controlling air conditioner based on reading the temperature,
- Carbon monoxide (CO) gas detector,
- Flame detection system,
- Smoke gas detection,
- Vibration detection system,
- Motion detection system,
- Touch detection sensor,
- Sound detection system, and
- Barometric pressure measurement system.

The publishers have been programmed with algorithms similar to the one in Fig. 2. Two steps were required to complete the physical testbed: setting up the subscribers and the attack machine. Two subscribers have been set up to receive messages from their 8 publishers, and the other two subscribers were ready to receive messages from the remaining 8 publishers. The attack machine was additionally set up, and this machine conducted various DoS and DDoS attacks over a variety of scenarios.

4. Data collection

Normal MQTT traffic was created using the normal states of the protocol, as demonstrated in Fig. 3.

A TCP packet was sent to establish a three-way handshake, followed by a CONNECT packet containing authentication credentials for either the publisher or the subscriber. This way, publishers could transmit data for certain topics providing their authentication credentials were valid. Similarly, subscribers could subscribe to certain topics. Invalid authentication credentials resulted in a closed connection.

Using the Tshark tool, benign data collection began with selected data sizes of 50 MB and 200 MB. The same sizes have been used for the malicious data. In addition to collecting the benign data, the attack machine was run to launch 10 (D)DoS attack scenarios, which collected both the benign and malicious data. The DoS and DDoS attack scenarios described in Syed et al. (2020) were implemented in Python. These were basic CONNECT flooding (BF DoS and DDoS), delayed CONNECT flooding (delay DoS and DDoS), invalid subscription flooding (sub-DoS and -DDoS) and WILL payload CONNECT flooding (WILL DoS and DDoS) attacks. In contrast, SYN DoS and DDoS attacks were both launched with the hping3 tool.

The BF_DoS attacks have been launched using a loop that attempted to connect to the broker using random usernames and passwords, as shown in Fig. 4. The attack machine was connected to the broker on port 1883 and the keepalive value was 3600. For the BF_DoS attacks, there was no delay between the DoS attempts: the packets have been sent as fast as the attack machine could send them. This connection was closed by the broker because the credentials were incorrect. Nevertheless, the TCP three-way handshake occurred before sending the CONNECT packet from the attack machine. The attack machine also successfully attempted to connect to the broker by sending a second packet. It was noticed that the MQTT broker was accepting all attempts from the attack machine instead of blocking the IP address or taking further actions. The same setting was used to launch BF DDoS attacks, but using three machines at the same time.

The Delay_DoS attacks have been launched using a loop that attempted to send TCP three-way handshakes. However, before sending the CONNECT packet, the attack machine remained in sleep mode for a period of time. The Sleep-Time value was set to 0.1 s. This resulted in

Table 5
Network Design and Plan for the Physical IoT Testbed.

Networks	Network ID	Subnet Mask	CIDR	Broadcast	Interfaces
Cluster 1	192.168.70.0	255.255.255.0	/24	192.168.70.255	FastEthernet0/0
Cluster 2	192.168.60.0	255.255.255.0	/24	192.168.60.255	FastEthernet0/1
Subscribers Network 1	192.168.80.32	255.255.255.248	/29	192.168.80.39	Ethernet1/0.10
Subscribers Network 2	192.168.80.40	255.255.255.248	/29	192.168.80.47	Ethernet1/0.20
MQTT Broker Network	192.168.80.48	255.255.255.248	/29	192.168.80.55	Ethernet1/0.30
Attack Machine Network	192.168.90.0	255.255.255.0	/24	192.168.90.255	Ethernet1/3

```
Inputs:
Water-Channel, Broker-IP, Port, Sleep-Time, SPICLK, SPIMISO, SPIMOSI,
SPICS, Username, Password
Outputs:
Water level sensor sent in MQTT messages
Initialisations:
Water-Channel ← GPIO-Pins
Broker-IP ← B
Port ← P
Username ← U
Password ← P
SPICLK = Initial value
SPIMISO = Initial value
SPIMOSI = Initial value
SPICS = Initial value
Username-password-set (Username, Password)
CONNECT (Broker-IP, Port)
While True:
    Init ()
    ADC-Value = Read-ADC (Water-Channel, SPICLK, SPIMOSI, SPIMISO,
SPICS)
    If ADC-Value >0 and ADC-Value < 60:
        PUBLISH (Topic, “No water”)
    Else if ADC-Value >=60 and ADC-Value <1000
        PUBLISH (Topic, “The need to fill the tank”)
    Else if ADC-Value >=1000 and ADC-Value <1050:
        PUBLISH (Topic, "There is water in the tank")
Time-sleep (Sleep-Time)
        end
end
```

Fig. 2. Algorithm 1 Water Level Detection for Tank.

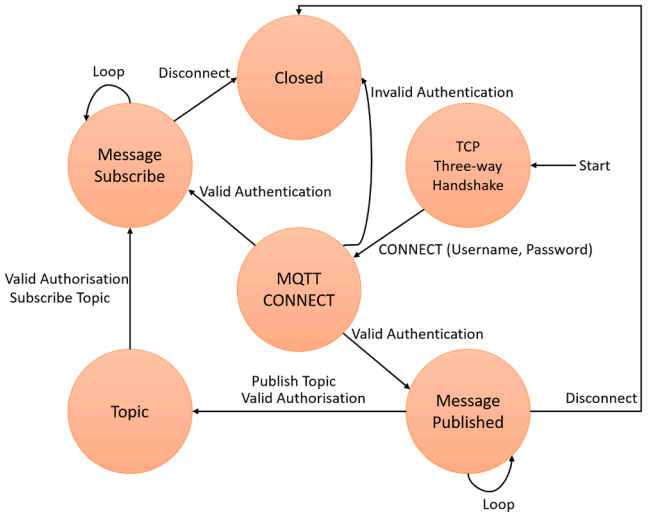


Fig. 3. State Model of Normal MQTT Traffic.

many TCP connections waiting for CONNECT packets. The keepalive value was the same used for the BF_DoS attacks, and the Delay_DoS attacks sent random usernames and passwords. In addition to Delay_DoS attacks, Delay_DDoS attacks have been generated using three attack machines. Due to the set Sleep-Time value for the Delay_DoS and Delay_DDoS attacks, fewer packets have been sent than in the case of the BF_DoS and BF_DDoS attacks.

The Sub_DoS attacks have been launched using a loop that attempted to subscribe to multiple topics. After connecting to the broker, attacks have been launched. In every packet sent from the attack machine, the Sub_DoS attack subscribed to large topics. In addition, there was no delay between the two packets. The WILL DoS attack was launched by sending large topics to the broker. This attack could be launched during and after the connection to the broker. In other words, there is no need to be successfully connected to launch this type of attack.

During the first stage of dataset generation, 38.8 GB of data have been captured, including 364 PCAP files, comprising 262,786,226 records in total. Features were extracted from PCAP files using Tshark and saved in CSV format. Table 6 presents a description of the selected features.

```
Inputs:
Broker-IP, Port, Sleep-Time, Username, Password
Outputs:
Initialisations:
Broker-IP ← B
Port ← P
While True:
  TCP ← Established
  Username ← Random U
  Password ← Random P
     Username-password-set (Username, Password)
     CONNECT (Broker-IP, Port, Keepalive)
     Time-sleep (Sleep-Time)
     end
end
```

Fig. 4. Pseudocode for BF DoS Attack.

Table 6
Description of Each Feature and its Associated Data Type (S = string, N = numeric).

No	Feature	Description	Data Type
1	frame.time_epoch	Epoch Time	N
2	frame.len	Frame Length	N
3	frame. time_delta_displayed	Time delta from previous displayed frame	N
4	frame.time_relative	Time since reference or first frame	N
5	ip.src	Source IP Address	S
6	ip.proto	Protocol	S
7	tcp.stream	Stream index	N
8	tcp.analysis.initial.rtt	iRTT	N
9	tcp.time_relative	Time since first frame in this TCP stream	N
10	tcp.len	TCP Segment Len	N
11	tcp.window_size	Calculated window size	N
12	tcp.flags.syn	Syn	S
13	tcp.flags.reset	Reset	S
14	tcp.flags.ack	Acknowledgment	S
15	mqtt.msgtype	Message Type	S
16	mqtt.qos	QoS Level	S
17	mqtt.conflag.qos	QoS Level Flag	S
18	mqtt.sub.qos	MQTT Subscriber QoS	N
19	mqtt.conflag.cleansess	Clean Session Flag	S
20	mqtt.kalive	Keep Alive	N
21	mqtt.username_len	User Name Length	N
22	mqtt.passwd_len	Password Length	N
23	mqtt.retain	Retain	N
24	mqtt.conflag.retain	Will Retain	S
25	mqtt.conflag.willflag	Will Flag	S
26	mqtt.willmsg_len	Will Message Length	N
27	mqtt.willtopic_len	Will Topic Length	N
28	mqtt.topic_len	Topic Length	N
29	mqtt.len	Msg Len	N
30	mqtt.conack.val	Return Code	N

30 features have been extracted from the data link layer, network layer, transport layer, and the application layer. Some of these were specific to the MQTT protocol, while others were related to TCP. IP addresses have been extracted from the network layer, and the time and length values have been extracted from the data link layer.

4.1. Sanitizing and preprocessing the data

The captured data has been sanitized and preprocessed. 3654560 records have been used for evaluating the ML algorithms, which corresponds to 2.3% of all the records. The following actions have been taken in the CSV files:

- Duplicate data have been removed,
- The sum of numbers in the cells has been calculated,
- Some string values have been transformed into numbers,
- Empty rows have been zero-filled, and
- The data have been labelled.

String values have been transformed as shown in Table 7. To delete duplicate data, a Microsoft Visual Basic for Applications (VBA) program was used. For example, in the case of “Publish Message, Publish Message, Publish Message”, the VB function kept only the first Publish Message, and removed the others. To add the numbers within a single cell, an integrated tool called *Kutools*[10] was used in Excel. The next step was to replace strings with numbers, as detailed in Table 7. 8 columns have string values: Protocol, SYN Flag, Rest Flag, Acknowledgment Flag, Clean Session Flag, Retain, Will Retain, and the Will Flag.

A VBA program was used to fill the empty rows with zeros. Finally, data were labelled; attacked data was labelled "1" and normal data was labelled "0". The labelled data was based on the IP address, so for the DoS attack scenarios, any rows that were assigned the IP address "192.168.90.100", "192.168.90.101", or "192.168.90.102" were labelled as "1". Otherwise, they were labelled with zeros. Similarly, all of the IP addresses listed in the DDoS attack scenarios contained 1, whereas the rest were zeros.

The MQTT-IoT datasets have been split and classified as per Table 8. The MQTT-IoT dataset was divided into normal MQTT data, and normal MQTT data mixed with abnormally attacked data. The abnormal data were divided into 10 sets of DoS and DDoS attacks. Three columns in Table 8 are file size, quantity, and record. Quantity refers to the number of files of either 50 MB files or 200 MB files. Lastly, record refers to the

Table 7
How String Values Replacement Was Accomplished by Replacing Strings with Numbers.

Class	String	Replacement
Protocol	TCP	0
	MQTT	1
SYN Flag	Not set	0
	Set	1
Rest Flag	Not set	0
	Set	1
Acknowledgment Flag	Not set	0
	Set	1
Clean Session Flag	Not set	0
	Set	1
Retain	Not set	1
	Set	2
Will Retain	Not set	1
	Set	2
Will Flag	Not set	1
	Set	2

[10] https://www.extendoffice.com/product/kutools-for-excel.html

Table 8
MQTT-IoT datasets (benign and malicious data).

Data Name	File size	Quantity of files	#Records per file
Normal MQTT	50 MB	20	≈ 490000
	200 MB	30	≈ 1900000
BF_DoS	50 MB	20	≈ 510000
	200 MB	10	≈ 2000000
BF_DDoS	50 MB	20	≈ 510000
	200 MB	10	≈ 2000000
Delay_DoS	50 MB	20	≈ 500000
	200 MB	10	≈ 660000
Delay_DDoS	50 MB	20	≈ 510000
	200 MB	10	≈ 2000000
Sub_DoS	50 MB	20	≈ 130000
	200 MB	10	≈ 800000
Sub_DDoS	50 MB	20	≈ 200000
	200 MB	10	≈ 750000
WILL_DoS	50 MB	20	≈ 190000
	200 MB	10	≈ 650000
WILL_DDoS	50 MB	20	≈ 250000
	200 MB	10	≈ 1000000
SYN_DoS	50 MB	33	≈ 500000
	200 MB	10	≈ 1500000
SYN_DDoS	50 MB	20	≈ 500000
	200 MB	10	≈ 1500000

number of records in each file.

5. Experiments

To develop effective countermeasures, a realistic testbed has been set up with authentic network traffic to generate datasets. Hence, the primary research question was: How can IoT-MQTT DoS and DDoS datasets for countermeasures be generated?

Hypothesis H_1stated that a BF_DoS CONNECT flooding attack will be successful. Accordingly, the BF_DoS attack was designed in accordance with Fig. 4, and the Tshark tool was used to capture the data successfully.Nine further hypotheses were created to align with the nomenclature used in Table 9. Similarly, the rest of the hypotheses for the attacks such as a Delay_DoS delayed CONNECT flooding attack will be successful, a Sub_DoS invalid subscription flooding attack will be successful, and a WILL_DoS CONNECT flooding with WILL payload attack will be successful.

First, the sample dataset used to evaluate performance was considered. Hence, the evaluation results are presented for imbalanced datasets and balanced data. ML algorithms are applied to the imbalanced data before considering balanced data. Table 9 displays the distribution of the data within each of the classes of the dataset.

The algorithms used to test the datasets were decision trees (DT), k-nearest neighbors (K-NN), kernel-support vector machines (k-SVM), logistic regression (LR), naïve Bayes (NB), random forest (RF), extreme gradient boosting (XGBoost), and artificial neural network (ANN). The ML/DL algorithms were employed to demonstrate the effectiveness of the DoS/DDoS-MQTT-IoT dataset via the following metrics: accuracy, error rate (ER), true positive rate (TPR), and false positive rate (FPR). The demonstration of the classifiers for the DoS/DDoS-MQTT-IoT dataset were based on the imbalanced and balanced data and expressed in the form of accuracy, error rate, TPR, and FPR.

Table 10 depicts the comparison of the accuracy, error, true positive, and false positive rates for the various machine learning algorithms used to evaluate the dataset.

In terms of accuracy, the XG Boost and random forest algorithms performed best.

During the experiments, hypotheses H_1 to H_{10} were tested, and all of them were accepted in that the realistic datasets for the 10 scenarios of DoS and DDoS attacks had been captured. These results confirm the hypotheses and thus answer the research question.

Table 9
Data Distribution Used in the Confusion Matrix in Various Classes in the Dataset.

Name	Imbalanced Data	Balanced Data
BF_DoS	76,119	38,603
BF_DDoS	78,745	18,862
Delay_DoS	72,997	62,049
Delay_DDoS	77,389	39,652
Sub_DoS	27,288	9,403
Sub_DDoS	45,298	41,949
WILL_DoS	28,619	21,189
WILL_DDoS	39,014	12,915
SYN_DoS	25,944	11,567
SYN_DDoS	124,395	48,330

Table 10
Machine and Deep learning algorithms for evaluating Sub_DDoS imbalanced dataset.

ML/DL Algorithm	ACC	Error	TPR	FPR
Decision Tree	91.33	8.66	0.91	0.08
K-Nearest Neighbors	90.23	9.76	0.88	0.08
Kernel-SVM	87.09	12.90	0.81	0.06
Logistic Regression	86.32	13.67	0.81	0.08
Naïve Bayes	69.80	30.20	1.00	0.36
Random Forest	92.69	7.30	0.91	0.06
XG Boost	92.72	7.27	0.90	0.05
Artificial Neural Networks	89.90	10.09	0.85	0.04
Decision Tree	95.94	4.05	0.95	0.03
K-Nearest Neighbors	95.21	4.78	0.94	0.04
Kernel-SVM	82.80	17.19	0.70	0.04
Logistic Regression	82.66	17.33	0.70	0.06
Naïve Bayes	75.33	24.66	1.00	0.28
Random Forest	96.65	3.34	0.95	0.02
XG Boost	96.31	3.68	0.96	0.03
Artificial Neural Networks	94.58	5.41	0.96	0.06
Decision Tree	95.75	4.24	0.87	0.03
K-Nearest Neighbors	94.32	5.67	0.87	0.04
Kernel-SVM	89.84	10.15	0.94	0.10
Logistic Regression	89.52	10.47	0.96	0.11
Naïve Bayes	89.08	10.91	0.99	0.12
Random Forest	96.70	3.29	0.91	0.02
XG Boost	96.74	3.25	0.93	0.03
Artificial Neural Networks	94.34	5.65	0.90	0.05
Decision Tree	99.01	0.98	0.98	0.01
K-Nearest Neighbors	98.73	1.26	0.98	0.01
Kernel-SVM	97.16	2.83	0.99	0.03
Logistic Regression	97.20	2.79	1.00	0.03
Naïve Bayes	96.76	3.23	1.00	0.04
Random Forest	99.14	0.85	0.99	0.01
XG Boost	99.24	0.75	0.99	0.01
Artificial Neural Networks	98.62	1.37	0.98	0.01
Decision Tree	100.00	0.00	1.00	0.00
K-Nearest Neighbors	99.99	0.00	1.00	0.00
Kernel-SVM	99.98	0.01	1.00	0.00
Logistic Regression	99.95	0.04	1.00	0.00
Naïve Bayes	100.00	0.00	1.00	0.00
Random Forest	100.00	0.00	1.00	0.00
XG Boost	100.00	0.00	1.00	0.00
Artificial Neural Networks	80.43	19.56	0.80	N/A

6. Conclusion and future work

In this paper, a dataset called DoS/DDoS-MQTT-IoT was proposed for evaluating intrusion detections in IoT networks that use MQTT, filling a gap in a domain for which no similar datasets are available. This new dataset can be used by security analysts to devise countermeasures to attacks in MQTT networks, and by researchers to investigate existing and develop new algorithms for the efficient processing of attack data in such networks.

DoS/DDoS-MQTT-IoT was generated using a physical IoT testbed on which actual DoS attacks have been performed. The dataset contains both normal traffic data and data accompanied by attack data (10 types of DoS attacks). We detailed the algorithms used for programming the

sensors, as well as the machine learning classifiers used (DT, k-NN, K-SVM, LR, NB, RF, XGBoost, and ANN). An initial analysis was conducted using confusion matrices, followed by a detailed analysis using conventional metrics (accuracy, error, TPR, and FPR). A comparison of classifiers has been conducted, based on which two algorithms can be recommended: RF and XGBoost. The applicability of the dataset was verified via experiments based on real-world attack scenarios.

The dataset contains thirty different features, several protocols and a mixture of benign and malicious data. Other uses of the dataset include:

- Testing other attack scenarios, as it (the dataset) contains benign as well as malicious traffic. The benign traffic could be incorporated into other attack data that is not DoS/DDoS-based; and
- Testing with other protocols. The feature set also includes protocol data that is not MQTT. The data could be used to test other protocols.

Declaration of Competing Interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:Dr Leslie F Sikos reports was provided by Edith Cowan University

Data availability

Data will be made available on request.

References

[1] Z. Baig, P. Szewczyk, C. Valli, P. Rabadia, P. Hannay, M. Chernyshev, M. Johnstone, P. Kerai, A. Ibrahim, K. Sansurooah, N. Syed, M. Peacock, Future challenges for smart cities: cyber-security and digital forensics, Digital Invest. 22 (2017) 3–13, https://doi.org/10.1016/j.diin.2017.06.015.

[2] CAIDA, The CAIDA "DDoS Attack 2007" Dataset, 2007. https://www.caida.org/catalog/datasets/ddos-20070804_dataset/.

[3] S.N. Firdous, IoT-MQTT Based Denial of Service Attack Modelling and Detection, 2020. https://ro.ecu.edu.au/theses/2303/.

[4] S.N. Firdous, Z. Baig, C. Valli, A. Ibrahim, Modelling and evaluation of malicious attacks against the iot mqtt protocol, in: 2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData). Exeter, UK, 2017.

[5] H. Hindy, E. Bayne, M. Bures, R. Atkinson, C. Tachtatzis, X. Bellekens, Machine Learning Based IoT Intrusion Detection System: An MQTT Case Study (MQTT-IoT-IDS2020 Dataset), International Networking Conference, 2020.

[6] M. Johnstone, M. Peacock, Seven Pitfalls of Using Data Science in Cybersecurity. Data Science in Cybersecurity and Cyberthreat Intelligence (115-129), Springer, 2020, https://doi.org/10.1007/978-3-030-38788-4_6.

[7] N. Koroniotis, N. Moustafa, E. Sitnikova, B. Turnbull, Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: bot-iot dataset, Future Generat. Comput. Syst. 100 (2019) 779–796, https://doi.org/10.1016/j.future.2019.05.041.

[8] Massachusetts Institute of Technology., 1999 DARPA Intrusion Detection Evaluation Dataset, 1999. https://www.ll.mit.edu/r-d/datasets/1999-darpa-intrusion-detection-evaluation-dataset.

[9] N. Moustafa, J. Slay, UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set), in: 2015 Military Communications and Information Systems Conference (MilCIS), 2015.

[10] OASIS. (2019). *MQTT Version 5.0*. https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html.

[11] A. Praseed, P.S. Thilagam, DDoS attacks at the application layer: challenges and research perspectives for safeguarding web applications, IEEE Commun. Surv. Tutor. 21 (1) (2018) 661–685.

[12] P. Sethi, S.R. Sarangi, Internet of things: architectures, protocols, and applications, J. Electric. Comput. Eng. 2017 (2017), https://doi.org/10.1155/2017/9324035.

[13] I. Sharafaldin, A.H. Lashkari, A.A. Ghorbani, Toward generating a new intrusion detection dataset and intrusion traffic characterization, ICISSP 1 (2018) 108–116.

[14] University of California, KDD Cup 1999 Data, 1999. https://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[15] I. Vaccari, G. Chiola, M. Aiello, M. Mongelli, E. Cambiaso, MQTTset, a new dataset for machine learning techniques on MQTT, Sensors 20 (22) (2020) 6578.

Alaa Alatram received a B.Sc. degree in computer engineering from the University of Jordan in 2018, and an M.Sc. degree in computing and security from Edith Cowan University, Australia in 2021. His current research interests include Internet of Things, the MQTT protocol, DoS/DDoS attacks, and network forensics.

Leslie F. Sikos is a computer scientist specializing in cybersecurity applications powered by AI and data science. He holds two PhD degrees and 20+ industry certificates. He has industry experience in data center and cloud infrastructures, cyberthreat prevention and mitigation, and firewall management. Dr. Sikos is a certified professional of the Australian Computer Society and a senior member of the IEEE. He is a member of the IEEE Computer Society's Technical Committee on Security and Privacy and Special Technical Community on Cybersecurity, and a founding member of the IEEE Special Interest Group on Big Data for Cybersecurity and Privacy. He has worked on cybersecurity research projects with the DST Group of the Australian Government's Department of Defence, CSIRO's Data61, and the Cyber Security CRC. His community engagement includes public talks, media appearances on ABC News and 7NEWS, and professional articles in the Cyber Risk Leaders Magazine and the CISO Magazine. He is a reviewer at flagship journals in cybersecurity, such as Computers & Security and the IEEE Transactions on Dependable and Secure Computing. Dr. Sikos published 70+ publications, including more than 20 books, his most influential works being AI in Cybersecurity and Data Science in Cybersecurity and Cyberthreat Intelligence.

Mike is an Associate Professor at the School of Science at Edith Cowan University where he teaches network security and mobile app development. As a member of the Security Research Institute at ECU, his work on resilient systems covers secure development methodologies, wireless sensor networks and the security of IoT devices with a focus on critical infrastructure. With over 30 years of experience in ICT, he provides consultancy services in cyber security for private industry, government and research organisations and has held various IT roles including programmer, systems analyst, project manager and network manager before moving to academia.

Dr Patryk Szewczyk is a senior cyber security lecturer at Edith Cowan University, Australia and a senior member of the Australian Computer Society. Patryk's research specialisations include cyber security, digital forensics and digital privacy. He has served as a reviewer for numerous international journals and conferences. Patryk has attained national awards for his research and community service achievements towards addressing end-user cyber security challenges.

James has worked in various areas including Health Informatics, IoT, Computing and Cybersecurity. He has worked in the telecommunications industry for over 25 years with projects in Telecom NZ (Spark NZ), Nokia, NBN Co, Telstra, Siemens and Vodafone Australia. He has specialised in Network Intelligence for wired and mobile networks during the earlier stages of his career and is teaching smart medical health informatics at National Taiwan University. He has also worked in Ethiopia as a volunteer.