A Distributed Mitigation Strategy against DoS attacks in Edge Computing

Giuseppe Potrino
DIMES Department
University of Calabria
P. Bucci 39/c, 87036 Rende (CS), Italy
giuseppe.potrino@unical.it

Floriano De Rango
DIMES Department
University of Calabria
P. Bucci 39/c, 87036 Rende (CS), Italy
derango@dimes.unical.it

Peppino Fazio
DIMES Department
University of Calabria
P. Bucci 39/c, 87036 Rende (CS), Italy
pfazio@dimes.unical.it

***Abstract*— Internet of Things (IoT) is a platform where every day devices become smarter, every day processing becomes intelligent, and every day communication becomes informative. Numerous challenges prevent to secure IoT devices and their end-to-end communication in an IoT environment. In fact, the IoT security is still an open challenge. The purpose of this work is to examine a distributed strategy for mitigating Denial of Service (DoS) attacks against the fog node in an edge computing context in which the nodes exchange messages through Message Queue Telemetry Transport (MQTT) protocol. The proposed strategy is based on a dynamic message sending frequency of the lightweight nodes. It is also mitigated data tampering and eavesdropping by using Elliptic Curve Cryptography (ECC).**

Keywords—IoT, DoS, MQTT, ECC, Edge computing

I. Introduction

According to Gartner, IoT is one of the top ten strategic technology trends [1]. The "Internet of Things" (sometimes also referred to as the "Internet of Everything," or IoE) generally refers to the multiple networks of devices or technology platforms ("things") that communicate with each other via wireless protocols and without direct human interaction [2]. When IoT is augmented with sensors and actuators, the technology becomes an instance of the more general class of cyber-physical systems, which also encompasses technologies such as smart grids, smart homes, intelligent transportation and smart cities [3]. The number of connected devices on the Internet will exceed 50 billion by 2020, this according to Cisco. By 2022, 1 trillion networked sensors will be embedded in the world around us, with up to 45 trillion in 20 years [4]. The global internet of things (IoT) market reached USD 598.2 Billion in 2015 and the market is expected to reach USD 724.2 Billion by 2023 [5]. Then, nowadays there are ever more IoT challenges that need to be faced for example scalability, data volumes (Big data), self-organizing, data interpretation, interoperability, automatic discovery, software complexity, security and privacy, wireless communication [6]. Since IoT devices typically have limited resources, the generated data are typically forwarded to a cloud computing platform for data processing and analysis. A cloud computing platform is a collection of centralized networking, computing, and storage resources that are accessible through the Internet. This means that network latency and jitter can become significant. To address the above issue, the edge computing paradigm has emerged in recent years. The edge computing is an intermediate computing layer between the cloud layer and the smart devices [7]. The edge computing paradigm offers many benefits, but it needs to face many challenges, for example scalability, complexity, dynamicity, heterogeneity, latency and security [8]. IoT is available for various platforms and this makes hard to find complete solutions for the actual challenges by security researcher. The purpose of this work is to increase the security of edge computing layer. In the considered IoT context there are several lightweight nodes sending messages to a fog node (a node with more computational and power resources situated at edge computing layer) by using the MQTT protocol. The fog node tries to respect some messages priorities. In case of DoS attack, each lightweight node tries to help the fog node decongestion by adapting its message sending frequency. The MQTT payloads are encrypted by using ECC for mitigating data tampering and eavesdropping.

The paper is organized as follows: related work is presented in section II; section III introduces the basic technologies involved in the proposal such as MQTT and ECC; the dynamic security system to mitigate DoS attack is discussed in section IV; in section V is presented the performance evaluation; finally, conclusions are summarized in section VI.

II. Related works

A. IoT security

For detecting silent attacks, precise and swift safety monitoring and intrusion detection are of utmost importance in IoT-based systems. An Intrusion Detection System (IDS) will prevent failures caused by adversaries and decide proper alert to prevent intrusion or to mitigate the impact of an intrusion [15]. The control of malicious activities can be done in two modalities [16]: Host based IDS (HIDS) which collects information about activities on a single host, and Network based IDS (NIDS) which acts on the whole network. An IDS cannot protect from cloning of things,

978-1-5386-8380-4/19/S31.00 ©2019 IEEE

malicious substitution of things, firmware replacement and extraction of security parameters, but it can offer protection from Eavesdropping, Man-in-the-middle attacks, Routing attacks, DoS attacks [17]. DoS attacks increase mostly the packet loss and the delay variation, which are critical factors, e.g. in real-time or streaming communication, as reported in [28, 29].

To face the IoT challenges, in [18] it was proposed a new approach to communication and manage the security key which is based on the MQTT protocol and ECC. Even, in [19] it was proposed a novel lightweight security solution for publish-subscribe based protocol in an IoT Fog networks using ECC.

In [20] it was proposed the Secure-MQTT (SMQTT) protocol with the purpose of increasing the security of the MQTT protocol. In particular, the Publish messages are replaced by a new message type called SPublish which is identified with the reserved MQTT header code "0000" and it is used to specify that the payload is ciphered with ECC.

In [21] it was proposed a security architecture for IoT publish/subscribe networks, in which MQTT is divided in two separated communication channels: a data channel and a channel for security control (CoP). CoP is a communication channel by which a device can authenticate itself to the broker and through which message reports can delay or enforce the current security policy for reaching the requested performance.

B. *Work objectives*

To increase the IoT security in [11] the use of SSL/TLS was proposed, but this can consume a lot of energy in an IoT device. Then, in [18], [19] and [20] the use of ECC was proposed. Like the works in [18], [19] and [20], the proposal wants to apply ECC to an IoT edge computing network. This work applies ECC on a MQTT based communication for mitigating data tampering and eavesdropping. Ciphered packets are recognized by using a mechanism like the proposal of [20]. DoS attacks in an MQTT context are already considered in [21] where it is proposed an architecture based on TCP transport layer by using TLS enabled transport channel for ensuring privacy in the communication. Differently, our work is based on UDP protocol (lighter than TCP), and reliability is managed by using the MQTT ACK messages. The use of MQTT payloads encryption can increase the packets processing delay. This can cause a DoS attack if the fog node receives many messages. The proposed mitigation for DoS attacks is made in a distributed manner. In particular, each lightweight node adapts its sending messages frequency on the basis of the response delay from the fog node. This helps the fog node in the decongestion process. Moreover, on the fog node it is placed a simple IDS for trying to respect message type priorities.

III. TECHNOLOGIES INVOLVED IN THE PROPOSAL

A. *An IoT lightweight protocol: MQTT*

In recent years, in the IoT context various lightweight protocols are proposed like MQTT, MQTT-SN and CoAP. In [9] it is observed that when the packet loss rate is low, MQTT deliver messages with lower delay than CoAP. In this work, we consider the use of MQTT protocol. MQTT [10] is a lightweight application level protocol generally used in IoT contexts. MQTT IoT devices communicate through an MQTT broker. The protocol architecture is composed by: Topic (a queue of messages supporting publish/subscribe pattern), Client (publisher or subscriber on a specified topic), Broker (the server on which topics are maintained), Session (it identifies the connection between a client and the server), Subscription (it attaches a client to a topic), Messages (data units exchange). The main message types that are exchanged in this protocol are: CONNECT (it starts a Session), PUBLISH (it publishes data on a topic), SUBSCRIBE (it starts a Subscription), UNSUBSCRIBE (it cancels a Subscription), DISCONNECTED (it closes a Session), CONNACK, PUBACK, SUBACK, UNSUBACK. MQTT protocol doesn't use cryptography, then it can be susceptible to data tampering and eavesdropping. Moreover, the broker can be subjected to DoS attacks (which can exhaust many resources [23]). The security can become a critical issue in e-Health applications in which this protocol is used [24, 25]. Moreover, the security management can use a lot of energy and this can significantly reduce the nodes lifetime [26, 27].

B. *Elliptic curve cryptography*

An approach to face data tampering and eavesdropping in MQTT protocol is to use Secure Socket Layer/Transport Layer Security (SSL/TLS) [11] but it can consume a significant amount of energy in an IoT device. Elliptic curve cryptosystems over finite field have some benefits like the key size can be considerably smaller compared to additional cryptosystems like RSA, Diffie-Hellman since only exponential time attack is known so far if the curve is carefully chosen [12], [13] and Elliptic Curve Cryptography depend on the difficulty of explaining the Elliptic Curve Discrete Logarithm Problem (ECDLP). The elliptic curve cryptography (ECC) is a type of asymmetric cipher based on points arithmetic on an elliptic curve [14].

IV. DYNAMIC IOT SECURITY SYSTEM FOR FOG NETWORKS

A. *Context*

The proposed system is collocated in an IoT context. A generic IoT context is composed by a Smart Devices Layer in which we can find IoT devices, an Edge Layer in which there are fog nodes each of which gathers data from its fog network nodes (in the Smart Devices Layer) and a Cloud

Layer which can communicate with all fog nodes (in the Edge Layer) (Fig. 1). Generally, between two layers there are security mechanisms. The proposed system is collocated between the Smart Devices Layer and the Edge Layer and it is composed by several lightweight nodes and a fog node (star topology). Lightweight nodes sense data and send them to the fog node periodically by using the MQTT protocol. On the fog node, which has a power source and more processing and storing capacity, it is positioned the MQTT broker which processes these messages.

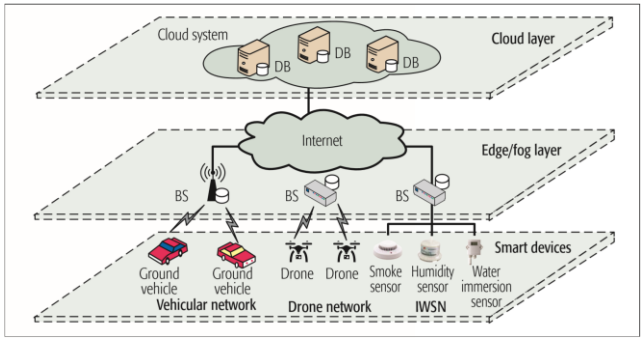

Fig. 1: Three Tier Architecture [7]

B. Proposed system

Generally, a ciphered payload needs of a longer elaboration time than a plain text one. If many lightweight nodes send messages at high frequency, the fog node can become congested. A possible attacker can compromise a lightweight node for making it sending messages at high frequency. This can make the system unusable. The proposal wants to adapt the lightweight nodes to these situations for helping the fog node to decongest itself. Then, this is made in a distributed manner. In particular, each of the lightweight nodes is able to detect possible congestion on the fog node and consequently to react. Each lightweight node measures the delay between the sent packet and the received acknowledgment. If the fog node is congested, its buffer becomes full and it starts to drop packets. As a consequence, some lightweight nodes will not receive an acknowledgement and someone will receive it in delay. These situations help lightweight nodes to detect a congestion on the fog node, and then, to decrease their sending messages frequency until the fog node become uncongested or until reaching the minimum frequency. At the end of the congestion, each lightweight node tries to increase its sending messages frequency until reaching the maximum frequency or until causing a new congestion on the fog node on the basis of the number of connected nodes. In fact, higher is the number of connected lightweight nodes and lower must be the sending messages frequency for avoiding congestion. This mechanism can be reassumed in the following pseudocode, that contains the lightweight node functioning.

Algorithm lightweight_node_behaviour:

```
int current_packet_type=KEY_EXCHANGE;
long current_interval=MINIMUM_INTERVAL;
loop(){
  Packet pack=null;
  if(current_packet_type==KEY_EXCHANGE){
    pack=forge_key_packet();
  }else if(current_packet_type==CONNECT){
    pack=forge_connect_packet();
  }else{
    pack=forge_publish_packet();
  }
  long start_time=System.current_time();
  Packet ack=send_packet_fognode(WAITING_TIMEOUT,
     pack);
  If(ack==null){
    current_interval=min(MAXIMUM_INTERVAL,
         current_interval*INCR_COEF);
  }else{
    long rtt=System.current_time()-start_time;
    If(rtt<WAITING_TIMEOUT*THRESHOLD){
      current_interval=max(MINIMUM_INTERVAL,
        current_interval*DECR_COEF);
    }else{
      current_interval=min(MAXIMUM_INTERVAL,
        current_interval*INCR_COEF);
    }
    if(current_packet_type==KEY_EXCHANGE){
      current_packet_type=CONNECT;
    }else if(current_packet_type==CONNECT){
      current_packet_type=PUBLISH;
    }
    elaborate_packet(ack);
  }
  sleep(current_interval);
}
```

Each lightweight node needs to exchange first the ECC keys, then it can send a CONNECT packet to initialize an MQTT connection and finally it can send PUBLISH messages. The "current_packet_type" variable represents the packet type that must be sent next. The "current_interval" variable represents the current interval that must elapse before sending the next message. At the beginning this interval is set to the minimum, then it is dynamically adapted to the current fog node congestion and can change between MINIMUM_INTERVAL and MAXIMUM_INTERVAL that are initially fixed. Even the INCR_COEF and DECR_COEF are initially fixed. Each lightweight node, after sending a packet, it waits for a WAITING_TIMEOUT to receive a response. If it does not happen, then it is supposed that the fog node is congested and then the "current_interval" can be increased. Otherwise, if it receives a response in time, if the Round-Trip-Time is

lower than WAITING_TIMEOUT*THRESHOLD the "current_interval" variable is decreased by supposing that the fog node is not congested else it is increased by supposing that the fog node is going to congest. The Fig. 2 summarizes the described mechanism between a generic lightweight node and the fog node.

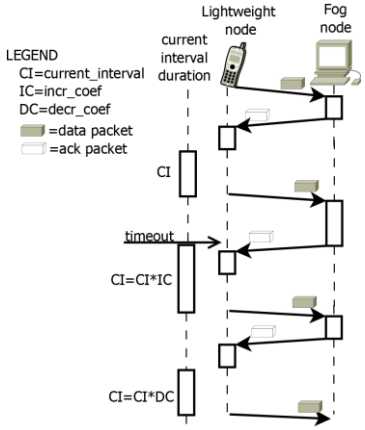

Fig. 2: Current_interval update

To avoid the reaching of the buffer fullness, the fog node starts to drop messages when the buffer fullness reaches a specified threshold. Respecting some priorities, the messages are dropped in the following order: key exchange messages, CONNECT messages and PUBLISH messages.

V. PERFORMANCE EVALUATION

A. Simulation environment

The proposed IoT system was evaluated by implementing an event driven simulator. It is implemented entirely in Java language and uses bcprov-jdk15on-160 [22] library for ECC. In Fig. 3, it is reported a diagram containing the main simulator modules.

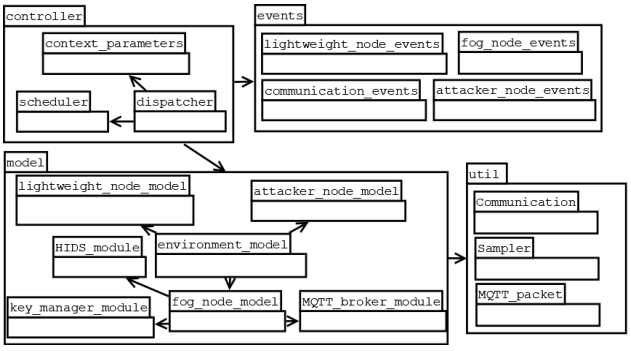

Fig. 3: UML of implemented simulator

The environment module creates, initializes and maintains all nodes. The fog node uses a MQTT broker for managing topics, a HIDS module for managing buffer level and a key manager for managing keys exchange and maintaining lightweight nodes public keys. The attacker node is a special lightweight node that sends packet at high frequency trying to create a congestion on the fog node. Lightweight nodes send and receive packets periodically. The simulator provides several context parameters included in a formatted CSV file. Each parameter can be modified by the user directly from file, where it is provided a detailed description. If the sampling mode is on, during the simulation some data regarding the dropped and timeout packets are collected and at the end of the simulation, these data are stored on a formatted CSV file from where it is possible to generate some graphs. The following paragraphs contain various graphs generated by the sampled data in several simulations. The table in Table I contains the used main default parameters

Context parameters		
Name	***Description***	***Value***
samplingInterval	It represents the sampling interval for data collection	5 s
lightweightNodeNumber	It represents the number of lightweight nodes	100
environmentSizes	It represents the length, the width and the height of the environment in meters	100, 100, 100
simulationEndTime	It represents the maximum simulation time	600 s
attackInterval	It represents the interval between two packet sent by attacker	10 ms
attackInitialTime	It represents the attack initial time	100 s
brokerBufferSize	It represents the size of the broker buffer for receiving messages	10240 bytes
brokerBufferMaxPerc	It represents the percentage of buffer that must be reached for starting dropping packets (fog node)	0.8
waiting_timeout	It represents the time for which each lightweight node waits for receiving a response after a message senting	40 ms
minimum_interval	It represents represents the minimum interval that must elapse between two message sending	50 ms
maximum_interval	It represents represents the maximum interval that must elapse between two message sending	8 s
threshold	It is applied to the waiting_timeout variable for compute an acceptable waiting period for supposing that the fog node is or isn't congested	0.5
incr_coef	It represents the coefficient for incresing the current_interval if a fog node congestion was supposed	1.5
decr_coef	It represents the coefficient for decreasing the current_interval if a fog node congestion wasn't supposed	0.75

Table I: Main default context parameters

The following simulations compare the static message sending (SMS) strategy (in which INCR_COEF and DECR_COEF are both set to 1.0) and the proposed adaptive message sending strategy (AMS). In particular, we use the values reported in Table II for the incr_coef and decr_coef variables.

Name	*Incr_coef value*	*Decr_coef value*
SMS strategy	1.0	1.0
AMS strategy (1.1, 0.9)	1.1	0.9
AMS strategy (1.5, 0.75)	1.5	0.75
AMS strategy (3.0, 0.33)	3.0	0.33

Table II: Strategies parameters

B. Dropped messages by the fog node

In this paragraph, we analyze the number of dropped messages by the fog node in relations to their type. In Fig. 4 it is shown the number of dropped key messages in time by the fog node for the various types of message sending strategy. We can see that the number of dropped key messages in the SMS strategy is much higher than the AMS ones. Moreover, by using a high dynamicity for "incr_coef" and "decr_coef" helps the system to rapidly adapt the message sending frequency and this causes a lower number of dropped key messages. With the start of the DoS attack and after that the fog node buffer has become full, the curves trends are different because in SMS strategy all nodes continue to send messages at the same frequency while in AMS strategies the nodes adapt their sending messages frequencies for helping the fog node. In Fig. 5 it is shown the number of dropped CONNECT messages in time by the fog node for various types of message sending strategy. Even in this case the number of dropped CONNECT messages in the SMS strategy is much higher than the AMS ones and a high dynamicity for "incr_coef" and "decr_coef" helps the system to rapidly adapt the message sending frequency and this causes a lower number of dropped CONNECT messages. This happens for the same previously described reason. In Fig. 6 it is shown the number of dropped PUBLISH messages in time by the fog node for various types of message sending strategy. Even in this case the number of dropped PUBLISH messages in the SMS strategy is higher than the AMS ones for the same reason. From this graph we can also see that by using a high dynamicity the number of dropped PUBLISH messages increases. This can be a consequence of the use of very dynamic coefficients that can cause several jumps from the optimal frequency. In fact if it is sensed no congestion on the fog node and we drastically increase the frequency, we can cause a congestion. This can happen repeatedly causing an increase of dropped messages. In Fig. 7 it is shown the number of dropped messages because of full buffer reached by the fog node. Unlike previous graphs, where the dropping process is controlled by the fog node at reaching of "brokerBufferMaxPerc" fullness, in this graph the dropping process is uncontrolled because the fog node buffer is completely full. However, also in this case, the number of dropped messages for the SMS strategy is higher than the AMS ones, but in a minor way. We also can see that by using high dynamic coefficient the number of dropped messages for full buffer reaching can become lower. From Fig. 6 and Fig. 7, we can see that AMS Strategy (3.0, 0.33) is able to cause a more controlled dropping process. In the controlled dropping process of the first three graphs we can see that this dropping respect the desired priorities. In fact, the number of dropped key messages is higher than those of CONNECT messages, that in turn, is higher than those of PUBLISH messages. In all previously described graphs it is possible to note the presence of some step in the trends of the AMS strategy. These steps represent, for example, the start of the DoS attack and the time when the fog node buffer becomes full. Each step is followed by a lowering of the growth trend coefficient because of frequency adaptation by lightweight nodes.

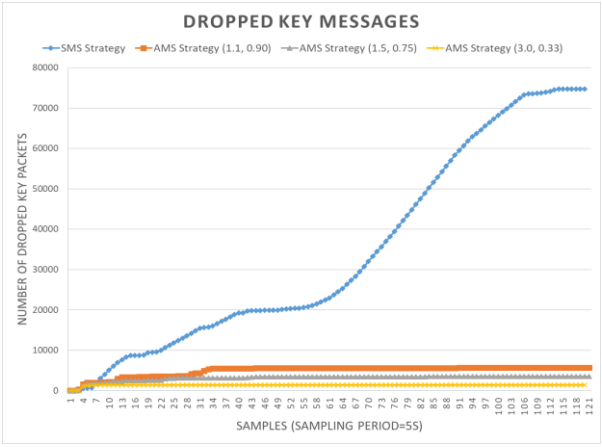

Fig. 4: Dropped key messages

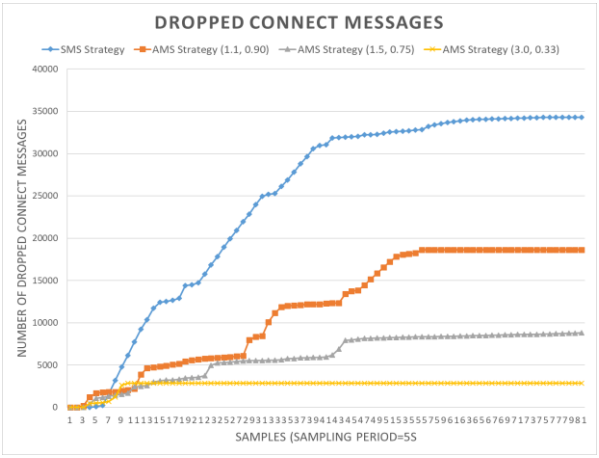

Fig. 5: Dropped CONNECT messages

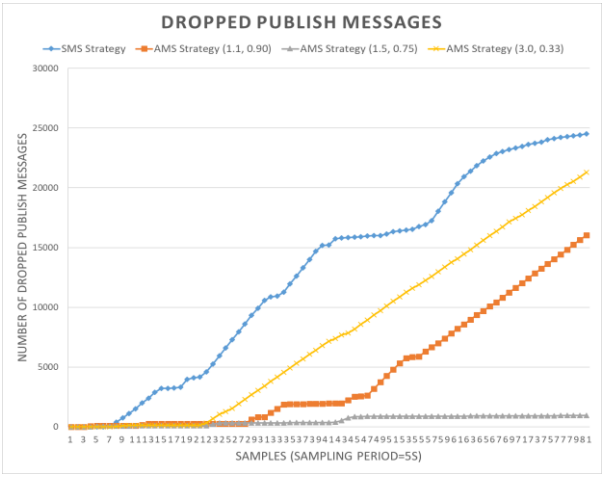

Fig. 6: Dropped PUBLISH messages

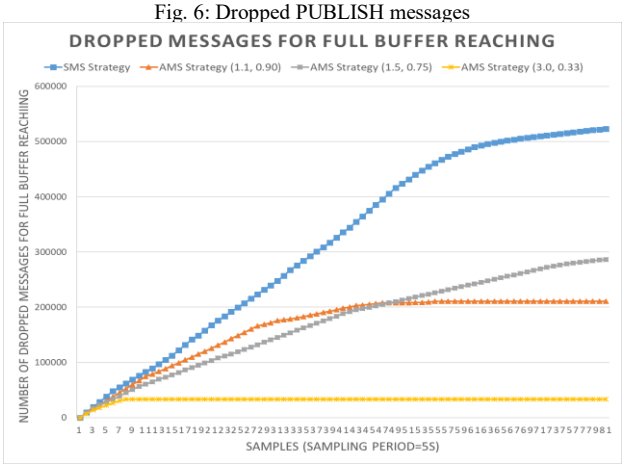

Fig. 7: Dropped messages for full buffer reaching

C. *Timeout messages*

In the following simulations we analyze the number of timeout occurred in the lightweight nodes in relations to their type.

In Fig. 8 it is shown the number of CONNECT messages sent by the lightweight nodes for which a CONNACK response was not received within WAITING_TIMEOUT for the various strategies. We can see that the number of timeout CONNACK messages for the SMS strategy is higher than the AMS strategy. This happens because the SMS strategy continues to send messages ever with the same frequency, also if the fog node is congested, while the AMS strategy changes its frequency in relation of the sensed congestion. In this case the difference is not significant because the CONNECT messages are sent principally at the start of the simulation when all lightweight nodes send messages at high frequency. Later, the AMS strategy adapts its frequency on the basis of the INCR_COEF and DECR_COEF. Then, the steepness of the AMS curve depends of these coefficients. In fact, AMS Strategy (3.0, 0.33) adapts rapidly its frequency, causing a lower number of timeout CONNACK messages.

In Fig. 9 it is shown the number of PUBLISH messages sent by the lightweight nodes for which a PUBACK response was not received within WAITING_TIMEOUT for the various strategies. We can see that the number of timeout PUBACK messages in the SMS strategy is much higher than the AMS ones. This happens because the SMS strategy continues to send messages ever with the same frequency, also if the fog node is congested, while the AMS ones change their frequency in relations of the sensed congestion. In this case the difference is significant because when the lightweight nodes with the AMS strategy start to send PUBLISH messages they have already a good frequency which was adjusted in the first step at connection establishment (Key exchange and CONNECT messages).

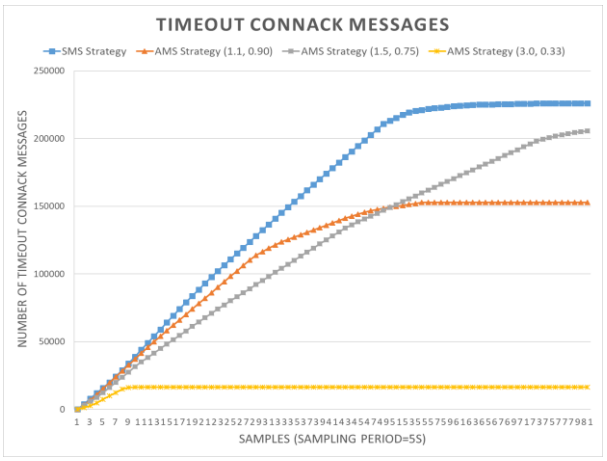

Fig. 8: Timeout CONNACK messages

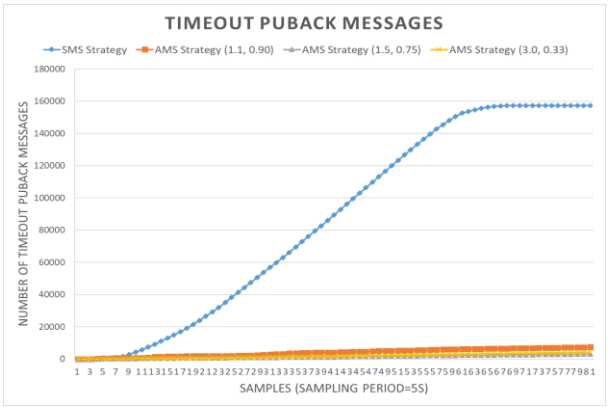

Fig. 9: Timeout PUBACK messages

VI. CONCLUSIONS

This work has permitted to propose a distributed mitigation strategy for DoS attacks in an edge computing

context in which there are lightweight nodes exchanging data through a secured MQTT protocol. The security system is based on ECC for mitigating data tampering and eavesdropping. The fog node uses a Host Intrusion Detection system for dropping messages until the reaching of buffer fullness for granting messages priorities. Lightweight nodes use an Adaptive Message Sending strategy for helping the fog node in the decongestion process because it performs better than the Static Message Sending one.

The proposed security system was validated by the implementation of an event driven simulator able to collect data that can be used for generating some graphs. The effectuated simulations show that the proposed AMS is more suitable to a fog network context for mitigating DoS attacks to the fog node. In fact, if this mitigation is made in a distributed manner it can be more scalable. Moreover, higher is the used dynamicity and lower is the adaptation time in case of congestion. But a very high dynamicity can become a problem because it causes repeatedly jumps from optimal point causing system malfunctions.

ACKNOWLEDGMENT

This work was supported by "POR Calabria FSE/FESR 2014/2020 – International mobility of PhD students and Research Grants/Type A Researchers" - Actions 10.5.6 and 10.5.12 actuated by Regione Calabria, Italy.

REFERENCES

[1] Orlando, «Gartner Identifies the Top 10 Strategic Technology Trends for 2013», *Gartner,* 2012.

[2] UL, «An introduction to the internet of things,» *White paper,* 2016, https://library.ul.com/wp-content/uploads/sites/40/2016/02/Internet-of-Things-white-paper_final.pdf.pdf.

[3] Alexia Mourtou, Anastasios Kyranas, Panagiotis Yannakopoulos, « Internet of Things», 2014.

[4] Vala Afshar, « Cisco: Enterprises Are Leading The Internet of Things Innovation,» *Huffpost,* 2017, https://www.huffingtonpost.com/entry/cisco-enterprises-are-leading-the-internet-of-things_us_59a41fcee4b0a62d0987b0c6?guccounter=1.

[5] ICT & Electronics, « Internet of Things (IoT) Market : Global Demand, Growth Analysis & Opportunity Outlook 2023,», https://www.researchnester.com/reports/internet-of-things-iot-market-global-demand-growth-analysis-opportunity-outlook-2023/216.

[6] Mritunjay Kumar, Km Annoo e Raman Kumar Mandal, «The Internet of Things Applications for Challenges and Related Future Technologies & Development,» *International Research Journal of Engineering and Technology (IRJET) ,* vol. 5, n. 1, Jan-2018.

[7] N. e. a. Abbas, «Mobile Edge computing: A Survey,» *IEEE Internet of Things,* vol. 5, n. 1, pp. 450-65, 2018.

[8] F. A. Hany, B. Gary e B. W. Gary, «Fog Computing and the Internet of Things: A review,» *Big Data Cognitive Computing,* 2018.

[9] S. Barber, P. Mahalle, A. Stango e N. Prasad, «Proposed security model and threat taxonomy for the internet of things,» *Recent Trends in Network Security and Applications, Springer Heidelberg,* pp. 420-429, 2010.

[10] P. R. Egli, «MQTT MQ Telemetry Transport - AN INTRODUCTION TO MQTT, A PROTOCOL FOR M2M AND IoT APPLICATIONS,» *indigoo.com,* 2016.

[11] E. Rescorla, «SSL and TLS: Designing and Building Secure Systems,» *Addison-Wesley Reading,* 2001.

[12] K. N., «Elliptic Curve Cryptosystems,» *Mathematics of Computation,* vol. 48, pp. 203-209, 1987.

[13] D. H. e. al., «Guide to Elliptic Curve Cryptography».

[14] M. V. S., «Use of Elliptic Curves in Cryptography,» *Springer-Verlang Berlin Heidelberg,* 1986.

[15] H. Farhoud, V. A. Payam, P. Juha, H. Timo e T. Hanu, «An Intrusion Detection System for Fog Computing and IoT based Logistic Systems using a Smart Data Approach,» *International Journal of digital Content Technology and its Applications (JDCTA),* vol. 10, n. 5, 2016.

[16] Bace & Rebecca, «An introduction to Intrudison Detection & Assessment,» *Infidel Inc. prepared for ICSA Inc,* 1998.

[17] Oscar Garcia-Morchon e H. Renè, «Security consideration in the IP-based Internet of Things,» *IETF Internet-Draft,* 2013.

[18] A. Mektoubi, H. L. Hassani, H. Belhadaout e M. Rifi, «New approach for securing communication over MQTT protocol: A comparison between RSA and Elliptic Curve,» *IEEE,* 2016.

[19] A. A. Diro, N. Chilamkurti e N. Kumar, «Lightweight Cybersecurity Schemes Using Elliptic Curve Cryptography in Publish-Subscribe fog Computing,» *Springer Science + Business Media ,* New York 2017.

[20] R. M. S. V. a. B. P. Meena Singh, «Secure MQTT for Internet of Things (IoT),» *Fifth International Conference on Communication Systems and Network Technologies,* 2015.

[21] P. Victor-Valeriu, C. Bogdan-Cosmin e B. Ion, «Mitigating DoS attacks in publish-subscribe IoT networks,» *ECAI 2017-International Conference-9th edition,* 2017.

[22] «The Legion of the Bouncy Castle», Available: https://www.bouncycastle.org/latest_releases.htm

[23] F De Rango, DC Lentini, S Marano, «Static and dynamic 4-way handshake solutions to avoid denial of service attack in Wi-Fi protected access and IEEE 802.11i,» *in EURASIP Journal on Wireless Communications and Networking*, Vo. (1), 2006.

[24] A.F. Santamaria, F. De Rango, A. Serianni, P. Raimondo, «A real IoT device deployment for e-Health applications under lightweight communication protocols, activity classifier and edge data filtering, » in *Computer Communications*, 2018, Vo. 128, pp. 60-73.

[25] A.F. Santamaria, P. Raimondo, F. De Rango, A. Serianni, «A two stages fuzzy logic approach for Internet of Things (IoT) wearable devices, » *in IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications*, 2016, pp.1-6.

[26] A. Lupia, CA Kerrache, F De Rango, C.T. Calafate, J.C. Cano, P. Manzoni, «TEEM: Trust-based Energy-Efficient Distributed Monitoring for Mobile Ad-hoc Networks, » in *Wireless Days* (WD), 2017, pp.133-135.

[27] A Lupia, F De Rango, «Evaluation of the energy consumption introduced by a trust management scheme on mobile ad-hoc networks, » in *Journal of Networks* (JNW), Vol. 10 (4), 2015, pp.240-252.

[28] Frnda, J., Voznak, M., Sevcik, L. «Impact of packet loss and delay variation on the quality of real-time video streaming», *Telecommunication Systems*, 2016, 62 (2), pp. 265-275. DOI: 10.1007/s11235-015-0037-2

[29] Voznak, M., Kovac, A., Halas, M. «Effective packet loss estimation on VoIP jitter buffer», *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2012, 7291 LNCS, pp. 157-162. DOI: 10.1007/978-3-642-30039-4_21