Towards Covert-Channel based IoT Authentication

João Alves[1,2] and Ricardo Severino[1][0000−1111−2222−3333]

[1] PORTIC - Porto Research, Technology and Innovation Center, Polytechnic Institute of Porto, Portugal
sev@portic.ipp.pt
[2] ISEP/IPP Instituto Superior de Engenharia do Porto, Portugal
1171575@isep.ipp.pt

Abstract. Network covert channels are increasingly being used to support malware with stealthy behaviours in IoT deployments, aiming at exfiltrating data or to secretly orchestrating nodes of a botnet. Although concerning, such covert communication strategies can also be effectively leveraged in a beneficial way, to support lightweight message authentication mechanisms which can not only remain undetected to the prying eye, but also mitigate any kind of ill intended exploitation of such covert techniques. In this work, we design and implement a new authentication mechanism relying on timing covert channels and analyze its performance over the IEEE 802.15.4-DSME protocol. Such extra layer of security can effectively contribute to improve the current state-of-art of cybersecurity on IoT systems which rely on such, or similar underlying communication technologies.

Keywords: IoT · Covert-Channel · IEEE 802.15.4. · Timing · DSME

1 Introduction

The advancements in information and communication technology in the past decades have been converging into a new communication paradigm in which everything is expected to be interconnected with the heightened pervasiveness and ubiquity of the Internet of Things (IoT). Increasingly becoming a pervasive reality, enabling applications in multiple domains such as medical care [1], agriculture [2], supply chains [3], transportation [4] and smart cities [5], IoT is increasingly finding their way into the industrial domain, to support what is now dubbed as the Industry 4.0, converging IoT, Cyber Physical Systems (CPS) and Cloud technologies into the factory floor. However, IoT technologies present a severe set of challenges, enhanced by the IoT scale, heterogeneity, and its fast adoption. Indeed, despite opening exciting opportunities, it also unlocks a variety of new security threats [6], [7], [8], with heightened security and privacy risks, as 70% of IoT devices are found to have serious security vulnerabilities [9]. The problem is further exacerbated as bad actors exploit this weakness to conduct attacks against IoT infrastructure [10], [11], [12], even taking down large swaths

of the Internet by leveraging D-DoS attacks such as the Mirai botnet [13], [14], which relied upon illegitimate usage of 400 000 IoT devices.

In such a troublesome scenario, particularly when physical access to the IoT infrastructure is possible, a very important element in any attack is thus represented by the capability of the attacker to exploit a covert channel and information hiding techniques. In fact, no matter if the goal is exfiltration of critical information or if it is to induce unintended behavior of a node, there is the need for communicating with the compromised node without disclosing the fact that it has been compromised. In this line, network covert channels are increasingly being used to support malware with stealthy behaviours (stegomalware), for instance to exfiltrate data or to orchestrate nodes of a botnet in a cloaked fashion [15]. However, the detection of such attacks is difficult as it is unknown in advance where the secret information has been hidden, and on the other hand, network covert channels usually feature low data-rates which complicates the detection. Consequently, countermeasures are tightly coupled to specific channel architectures, leading to poorly generalized and often scarcely scalable approaches.

However, such covert communication channels can also be explored in a beneficial way to improve the security of these systems. One of such examples relies in the implementation of a cloaked authentication mechanism, which can guarantee a legitimate origin of the IoT message in a lightweight fashion. Importantly, with such mechanism in place, besides achieving another layer of security, we can also mitigate any exploitation of such covert channel with any malicious intent, as such deliberate usage would disrupt the covert authentication protocol in place.

Sadly, such approaches have not been adequately explored in the literature regarding IoT communication systems. In this work, we design and implement a new authentication mechanism relying on timing covert channels and analyze its performance over the IEEE 802.15.4 protocol, well known for its wide deployment among several IoT applications. Such extra layer of security can effectively contribute to improve the current state-of-art of cybersecurity on IoT systems which rely on such, or similar underlying communication technologies.

2 Research Background

In network covert channels, data is hidden over legitimate communications by relying upon the network protocols as carriers. This concept of covert channels, first introduced in [16], postulates that, in general, covert channels can be divided into storage and timing channels. In storage covert channels, processes interact via a shared resource, directly or indirectly, using read and write operations. In the context of network steganography, storage covert channels hide data by storing them in the protocol header and/or in the Protocol Data Unit (PDU). On the other hand, timing channels hide data by manipulating some event timing, such as the packet inter-arrival time, or by changing the packet order.

The first evidence of an exploit of the timing features of network packets was developed in [17], by analysing several LAN protocols vulnerabilities to covert channel techniques. Since this date, the evolution regarding covert channels has

been considerable, and several kinds have been implemented in a plethora of widespread network protocols [18], [19], [20].

However, just a few works address the quite pervasive IEEE 802.15.4 protocol. At the physical layer, [21] developed a covert channel by exploiting the Direct Spread Spectrum Sequence (DSSS), using a steganography technique. They also propose a secret acknowledgement and error detection mechanism to ensure reliable communication in the covert channel. In [22] the authors proposed to develop a covert channel by manipulating the link quality as provided by the Link Quality Indication (LQI). The disadvantage of such approaches is that access and control of the node's physical layer is mandatory, something that is often not possible since these features are often deeply embedded at the radio transceiver's firmware. In this line, covert techniques that can function at the MAC sub-layer are much more attractive, particularly since these are usually implemented by a software stack, to which an application can much easily gain access to. In this regard, [23] explored several steganography options of the 802.15.4 protocol, in its 2006 version. Several fields such as the Frame Control, Sequence Number, Address Info, were explored for potential usage.

Timing covert channels implementations, on the other hand, only recently, in [24], were analyzed to understand the significant risks of timing covert channels implementations over such communication protocols. The conclusion is that such timing covert channels are very capable of being implemented in the DSME MAC behaviour of the 802.15.4 protocol, particularly over its DSME-GTS protocol, in different forms, some with great effectiveness, achieving a throughput capacity of 5.67 Bytes per second at moderate traffic rates. Such results encouraged us to find ways of mitigating such risk while re-purposing such covert strategies in novel directions, such as IIoT message authentication over this protocol. Indeed, although covert channels have multiple applications, one of their primary advantages is their secrecy. This characteristic makes it ideal for a possible implementation of an authentication mechanism.

For CAN networks, [25] developed a system to authenticate nodes through an insecure CAN bus. TACAN (Transmitter Authentication in CAN) provides secure authentication of ECUs by relying on covert channels without modifying the CAN protocol. They use keys and hashing algorithms to generate authentication messages transmitted over the covert channel and paired between sender and receiver to verify each other's identities. Also, [26] designed and implemented a lightweight authentication method in the FTP protocol in which node authentication keys are shared using a binary timing covert channel technique.

We believe similar approaches can and should be devised for the IEEE 802.15.4 protocol. In the next sections we overview the IEEE 802.15.4 protocol and introduce our implementation towards such goal.

3 Overview of the IEEE 802.15.4 DSME

The Deterministic Synchronous Multi-channel Extension (DSME) of IEEE 802.15.4 stands out from the remaining MAC behaviours, because of its features such as

multi-channel access and support for both contention-based and contention-free traffic, that increases the overall robustness, flexibility and scalability, while still providing deterministic communication. These properties also make it an interesting target for deployment of hidden Covert Timing Channels (CTC). The DSME network is time-synchronized by the Multisuperframe structure (Fig. 1). The rows that span across the Multisuperframe indicate the channels and the columns represent the timeslots. Every superframe within a Multisuperframe consists of Contention Access Period (CAP), that uses CSMA/CA for data transmission and Contention Free Period (CFP), that uses Guaranteed Timeslots (GTS). Every GTS slot accommodates the transmission of data and an eventual acknowledgment.

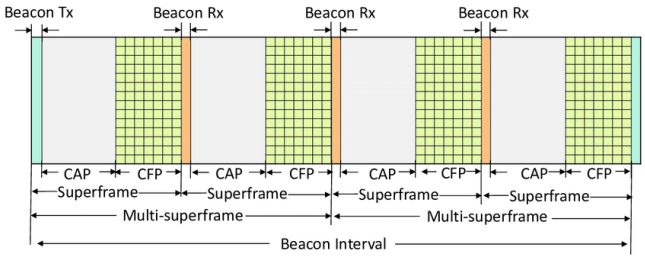

Fig. 1. IEEE 802.15.4 DSME multi superframe structure

The standard defines the structure of the superframe by the values of Superframe Duration (SD), Multi-superframe Duration (MD) and the Beacon Interval (BI). The Multi-superframe Duration is a new parameter introduced in DSME as compared to the native IEEE 802.15.4 superframe structure. These parameters are defined in the following equations:

$$MD = aBaseSuperframeDuration \times 2^{MO} symbols \quad (1)$$
$$for\ 0 \leq SO \leq MO \leq BO \leq 14$$
$$BI = aBaseSuperframeDuration \times 2^{BO} symbols \quad (2)$$
$$for\ 0 \leq BO \leq 14$$
$$SD = aBaseSuperframeDuration \times 2^{SO} symbols \quad (3)$$
$$for\ 0 \leq SO \leq BO \leq 14$$

where $aBaseSuperframeDuration$ denotes the minimum duration of a superframe corresponding to Superframe Order (SO) $SO = 0$. This duration is fixed to 960 symbols, corresponding to 15.36 ms, assuming 250 kbps in the 2.4 GHz frequency band. The total number of superframes and multi-superframes in a DSME superframe structure can be determined by $2^{(BO-SO)}$ and $2^{(BO-MO)}$, respectively. The PAN coordinator sets the values of BO, SO, and MO upon

network initialization and are then conveyed to the nodes via Enhanced Beacon at the beginning of each Multisuperframe. The superframe is defined by *BO*, the *Beacon Order* which specifies the transmission interval of a beacon which delimits the beginning of a superframe. *MO* is the *Multisuperframe Order* that impacts the *Enhanced Beacon* (EB) interval of a multi-superframe, and *SO* is the *Superframe Order* that specifies the size of a superframe.

4 System Design and Implementation

4.1 TimeCloaked Design

The work in [8] presents several possible implementations of timing covert channels. From these, the Time Replay technique seems quite suitable to support the intended authentication mechanism due to its flexibility in terms of data encoding capabilities and confidentiality. The technique works by manipulating the packet inter-arrival times between DSME-GTS transmissions. At setup time, one can establish an encoding matrix, assigning packet inter-arrival delays to covert data. We support the encoding of 2, 4 and 8 bit covert data. In addition, the Time Replay technique adds redundancy to the encoding delays, meaning that there are more than one inter-arrival delay to encode the same piece of covert data (Fig. 2). For instance, if we consider a single bit, the bit value 0 can be conveyed using a delay of 2, 4 or 6 symbols while the bit 1 can be encoded using an interval of 3, 5 or 7 symbols in length. This property was considered as very interesting as it adds an additional layer of security to the covert transmissions, making it mode difficult to detect patterns and decode the information in case of an adversary.

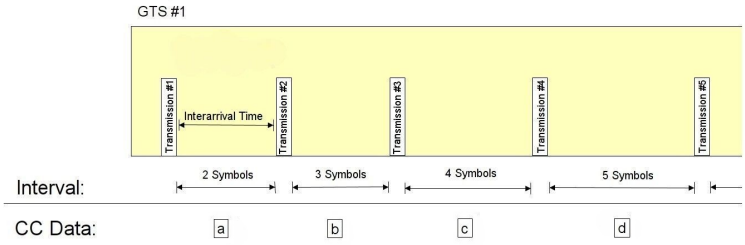

Fig. 2. Time Slot diagram of a Time Replay covert channel

For our architectural model, let us suppose Alice wants to communicate with Bob's robot arm (3), but Charlie, a vicious industrial hacker, is there to carry out a spoofing attack, by simply replaying the saved data at any time. As both Alice and Charlie would be sending data packets to Bob, Bob's robot would not be able to differentiate which were legitimate. Obviously, encryption could harden security, but keys can be compromised and often, a simple replay attack could

defeat it, if the application was not carefully crafted. The objective of *Time-Cloaked* is to enable a secondary and secret message authentication mechanism, of which Charlie is not aware. Hence, even if the attacker replays the data packet, TimeCloaked authentication will fail, thus alerting the receiver that illegitimate packets are being transmitted, and a probable attack is in progress.

As you can see from figure 3, Alice contains 2 keys, her own key and Bob's key, which identify sender and receiver. When transmitting, Alice inserts delays between each packet, thus covertly encoding the binary transmission key. Bob, who has the same mechanism as Alice, will store the packets Alice transmitted and record the delays until the key is complete. When the key is complete, the whole message (composed of multiple packets) is processed by the higher layer of the robot arm. As for the packets sent by Charlie, they will be discarded, because he does not have Bob's key and has no knowledge of the technique being used.

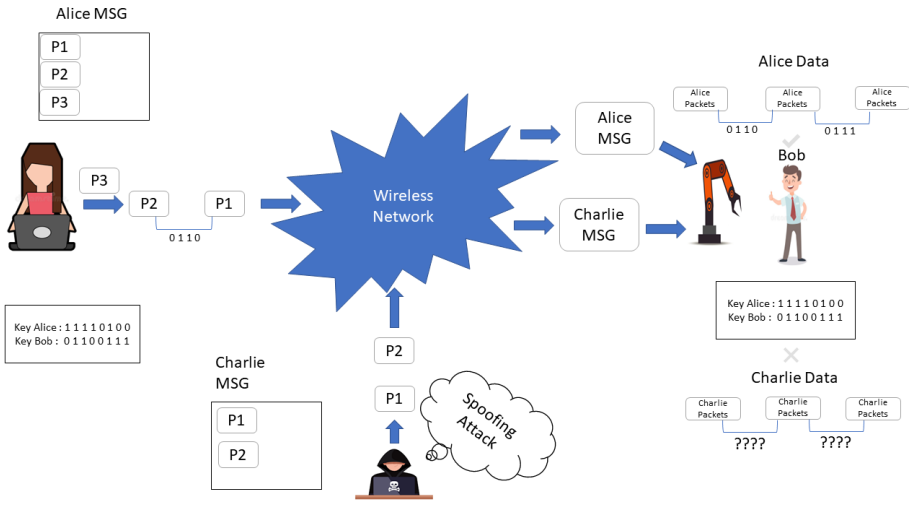

Fig. 3. Covert Channel Authentication Animation

4.2 Implementation

We implement the Time Replay covert channel over the DSME-GTS mechanism of the IEEE 802.15.4. Such mechanism supports contention-free traffic, therefore providing improved delivery guarantees and time-bounded communication. The implementation and assessment of the covert channels relies on the IEEE 802.15.4 openDSME simulation model [27], which was deployed over the OMNeT++ 6.0 event simulation framework, supported by the INET platform v4.3.7. This model is composed by two fundamental layers integrated into the MAC link layer, the DSMELayer and the DSMEAdaptionLayer (Fig. 4). The

first is responsible for implementing the newly released DSME MAC behaviour and all its features, while the adaption layer implements the base IEEE 802.15.4 functions required to perform a cohesive link with the rest of the OSI layers. The higher layer communicates with the lower DSME layer through two interfaces known roughly as Service Access Points (SAP). The DSMEPlatform module is responsible for harmonizing the whole model, interconnecting both the DSME-Layer and the DSMEAdaptionLayer. The model [3] is described in more detail in [27] and overviewed in Fig. 4. To fully encapsulate the covert channel implementation in the openDSME model, two new modules were created: the Covert Helper, responsible for the transmission of the messages with covert information and, therefore, inserting time delays into the network and the Covert Helper Receiver, in charge of interpreting the information being transmitted in the subliminal channel, by calculating the timing interval between received packets and, with that value, decoding the information being covertly transmitted.

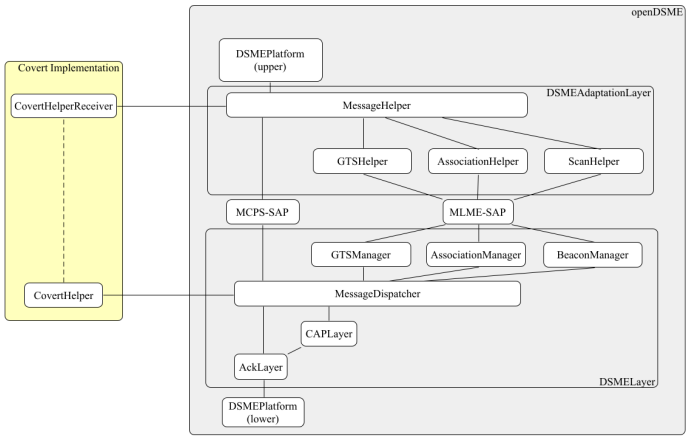

Fig. 4. openDSME architecture and covert modules implementation

Covert Helper At the beginning of each new covert message transmission, a start delay is introduced. The algorithm must then deconstruct the covert message into sets of bits, which are then translated into delays, multiple of the protocol symbol duration.

This correlation between the covert information to be transmitted and their associated delay must be known by transmitting and receiving side and is established in a encoding/decoding lookup table. For instance, a set of two bits (00) can be assigned to a delay of 2 symbols, set (01) to 3 symbols, set (11) to 4 symbols, etc. Because in this technique there are multiple encoding intervals

[3] Available in https://bitbucket.org/sev_portic/154_gts_ctc/src/master/

for the same data, a Random Number Generator is also implemented in order to select the interval to be used from a set of three.

From this association, the algorithm will find the next bits to transmit from the full message, get their corresponding delay and insert it into the packet's timing befor the new transmission. On the receiving side, the inserted delay will be obtained and decoded using the pre-arranged lookup table containing the association between delays and covert bits.

This algorithm is injected into the MessageDispatcher module of the DSME-Layer, of the OpenDSME stack, more particularly, in the *sendDoneGTS* method. In here, the transmitter node will pop messages from the send queue and, in each one, will check the time slot for available remaining time for packet transmission. This native method was also tweaked in order to account for the covert delay, upon checking if the time slot has space for a whole packet transmission. A few changes to this algorithm are introduced in the case of the other covert channel techniques. This process is discussed in greater detail in [24].

Covert Helper Receiver On the receiver's side, the packet inter-arrival interval is computed. However, to accurately compute it, one must consider the protocol wireless transmission time. Hence, the algorithm computes it, considering the specific packet payload and header size ($TxTime$). The remaining is then converted from milliseconds into multiples of protocol symbols (Equation 4).

$$CovertDelay = \frac{(CurrentInterval - TxTime) \times 960}{15.36} \tag{4}$$

where, *CurrentInterval* is computed as current time minus the last covert packet recorded time (in milliseconds), and $TxTime$ represents the wireless transmission time of the packet (Header+Payload). With this value, it is then possible to compute the covert delay inserted between the received packets. Afterwards, a simple comparison with the known encoding values is sufficient to determine the bit being covertly transmitted in the channel, or the initialization of a new transmission, which resets the storing variables. After a correct reception, the algorithm saves the received covert information and proceeds to next packet, restarting the process. Depending on the Timing Covert Channel (TCC) technique used, the algorithm will search different delay lists in order to decode the received covert bit (or set of bits).

4.3 Authentication Key Design

There are multiple possibilities to design the authentication key to carry out the TimeCloaked authentication. We implemented it in the following manner.

As already mentioned there are two keys, the transmitter key, and the receiver key. The receiver key will be modulated with the last delay of the transmitter key and the transmitter key will be modulated with the last delay of the receiver key. This modulation is done with the delay and a XOR of a number (unique to

each node) previously defined in the setup. The two keys together give rise to the authentication key which can be of different lengths, as selected by the user.

5 Validation and Results

For the validation we consider a star network topology with 11 nodes, organized in a circular shape, where the distance between the exterior nodes and the node[0], i.e. the PAN Coordinator, is approximately similar. This node, being placed at the center, acts as sink. All the data sender nodes will be scheduled according to an incrementing schedule (node[1] gets slot 1, node[2] gets slot 2, etc.). This data begins being generated in the 30th second of the simulation to ensure the already flowing state. From this second, the simulation will run for 200 seconds, recording several metrics of interest described below.

To evaluate *TimeCloaked* performance we carried out multiple simulation with different networks settings, particularly for different Superframe Order values, authentication key sizes and covert encoding lengths (1, 2, 4 and 8 bits) using the implemented Time Replay covert channel. Regarding SO, we used the settings from table 1. These parameters are sufficient to schedule all node's transmissions in a single multi-superframe, with one GTS Slot per node.

	SO	MO	BO
A	4	5	6
B	7	8	9
C	10	11	12

Table 1. Network DSME configurations

Figure 5 presents the results for the authentication delay for the different encoding techniques and different key lengths (covert encoding lengths/key length), by fixing SO = 7. It can be observed that in general, as expected, the delay increases as the key size increases as it takes more packets to transmit the lengthier key. Interestingly, the larger encoding capabilities is not significantly effective for smaller key lengths. This is because, smaller keys authentication can be achieved during the same slot, not increasing significantly the authentication delay. However, as the key length increases, at this SO, one slot is not sufficient to complete an authentication, which leads to a great increase in authentication delay for lower encoding techniques. Higher encoding techniques on the other hand can effectively keep the delay much lower, managing to complete an authentication in less slots.

This is visible in Figure 6 which presents the number authentications per DSME-GTS slot, for the different techniques and key sizes. Naturally, the need for more slots to complete and authentication lead to a great increase in delay, particularly for larger SO. It is also visible that using Time Replay encoding

techniques of 4 or 8 bits results in many more authentications per slot, which maximizes their utilization.

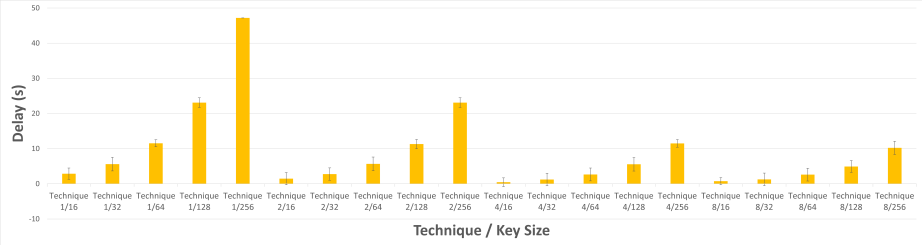

Fig. 5. Authentication Delay for SO7

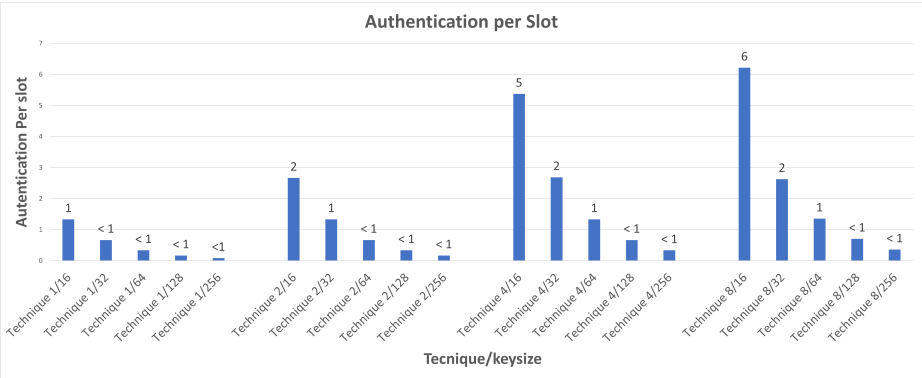

Fig. 6. Authentication per slot SO7

To better understand the performance of such techniques at different SO, we varied the SO from 4 to 10 and present the results of the number of authentications per DSME-GTS slot in Figure 7, with different key sizes (16, 64 and 256 bits) and different encoding techniques (1 and 8 bits).

The technique that presents the best results is the one that encodes more bits (8 bits) and that the authentication key size was smaller (16 bits). This is because of the larger encoding capacity of covert data per inter-arrival interval.

Interestingly, lower SO cannot support the larger 8-bit encoding techniques. This is because the available DSME-GTS slot length is so reduced, that the large interval set needed to encode data does not fit inside the slot. Hence, for lower SO, one must find a compromise and go for lower encoding techniques instead.

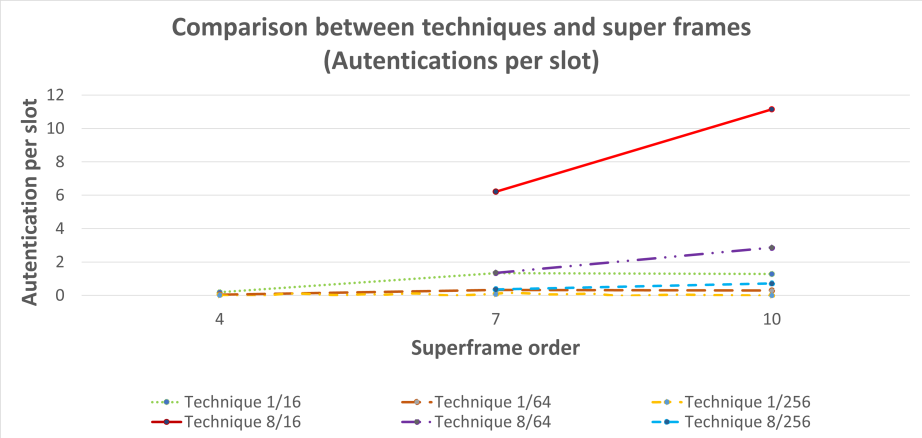

Fig. 7. Authentication per slot between SO with different key sizes

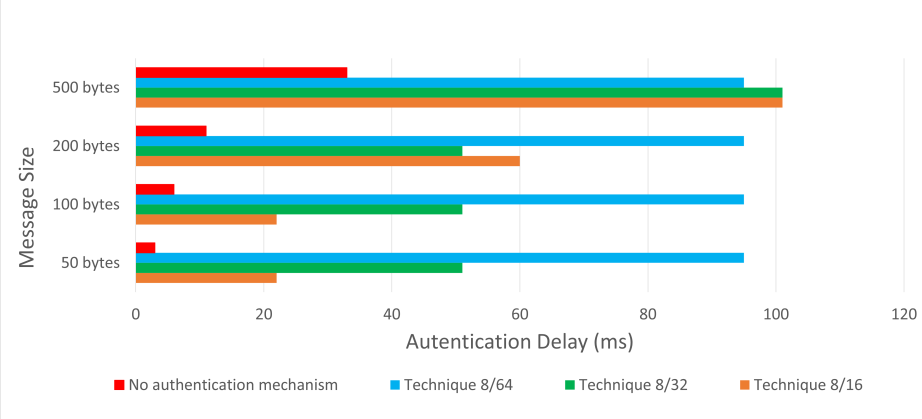

Fig. 8. Message reception relay with and without the authentication mechanism for different techniques (Payload:75B)

To understand the performance of TimeCloaked in authenticating different message sizes, we setup a set o simulation maximizing the packet payload to 75 bytes to maximize the amount of overt information and relied on SO = 7 and different authentication key sizes. We rely on 8 bit encoding for the covert channel. As presented in Figure 8, TimeCLoaked is able to authenticate up to 200 byte messages within 100 ms, by relying on 64 bit authentication keys. If one can decrease the key size to 32 or 16 bits, one can achieve less delay, up to 60 ms. That is because with small size keys, we need less slots to authenticate

the data. For 500 bytes messages however, even if using lower key sizes it takes approximately the same time, as the key overhead lies on the transmission of the big chunk of overt data which occupies several DSME-GTS slots, and are sufficient to support all authentication key sizes.

6 Conclusions

With the results obtained it was possible to conclude, that it is feasible to use timing covert channels to support an authentication mechanism for IoT, implemented over the DSME MAC behavior of the 802.15.4 protocol. With the carried out performance analysis it was possible to achieve very satisfactory results and understand the inner mechanisms of the solution.

Regarding the key size, it is possible to observe that the larger the key size, the longer the authentication will take, as expected. However this time can be minimized with the use of larger encoding techniques. An 8-bit encoding will be much faster than a 1-bit encoding. Such techniques are however limited to the SO constraints of the network. For instance, the 8 bit encoding technique cannot be supported by small SO due to the small GTS slot duration. Although TimeCloaked introduces some overhead in terms of delay, such mechanism can improve the security of IoT systems by supporting an additional cloaked authentication channel. In addition, the deployment of such covert channel technique prevents any illicit usage of such technique to exfiltrate data or with any other purpose.

We believe this mechanism, if further explored, holds the possibility to greatly increase the security of IoT networks. Although we have implemented the mechanism over a real stack, which can be deployed into IoT nodes, we plan to increase the modularity and flexibility of the software, to ease its deployment and setup.

References

1. Fadi Al-Turjman, Muhammad Hassan Nawaz, and Umit Deniz Ulusar. Intelligence in the Internet of Medical Things era: A systematic review of current and future trends. *Computer Communications*, 150:644–660, January 2020.
2. Akash Sinha, Gulshan Shrivastava, and Prabhat Kumar. Architecting user-centric internet of things for smart agriculture. *Sustainable Computing: Informatics and Systems*, 23:88–102, September 2019.
3. Jesús Muñuzuri, Luis Onieva, Pablo Cortés, and José Guadix. Using IoT data and applications to improve port-based intermodal supply chains. *Computers & Industrial Engineering*, 139:105668, January 2020.
4. Jia-huai Wang. T cell receptors, mechanosensors, catch bonds and immunotherapy. *Progress in Biophysics and Molecular Biology*, 153:23–27, July 2020.
5. Kun Liu, YunRui Bi, and Di Liu. Internet of Things based acquisition system of industrial intelligent bar code for smart city applications. *Computer Communications*, 150:325–333, January 2020.
6. Jie Lin, Wei Yu, Nan Zhang, Xinyu Yang, Hanlin Zhang, and Wei Zhao. A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications. *IEEE Internet of Things Journal*, 4(5):1125–1142, October 2017.

7. Mario Frustaci, Pasquale Pace, Gianluca Aloi, and Giancarlo Fortino. Evaluating Critical Security Issues of the IoT World: Present and Future Challenges. *IEEE Internet of Things Journal*, 5(4):2483–2495, August 2018.
8. Shoupu Lu, Zhifeng Chen, Guangxin Fu, and Qingbao Li. A novel timing-based network covert channel detection method. *Journal of Physics: Conference Series*, 1325:012050, 10 2019.
9. Kristi Rawlinson. HP Study Reveals 70 Percent of Internet of Things Devices Vulnerable to Attack, 2014.
10. Selena Larson. A smart fish tank left a casino vulnerable to hackers, July 2017.
11. Mark Stanislav and Tod Beardsley. HACKING IoT: A Case Study on Baby Monitor Exposures and Vulnerabilities. page 17, 2015.
12. Scott Simon. 'Internet Of Things' Hacking Attack Led To Widespread Outage Of Popular Websites. *NPR*, October 2016.
13. Brian Krebs. Mirai IoT Botnet Co-Authors Plead Guilty – Krebs on Security, 2017.
14. Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Jeffrey Voas. DDoS in the IoT: Mirai and other botnets. *Computer*, 50:80–84, January 2017.
15. Luca Caviglione. Trends and Challenges in Network Covert Channels Countermeasures. *Applied Sciences*, 11(4):1641, January 2021. Number: 4 Publisher: Multidisciplinary Digital Publishing Institute.
16. Butler W. Lampson. A note on the confinement problem. *Communications of the ACM*, 16(10):613–615, October 1973.
17. Manfred Wolf. Covert channels in LAN protocols. In Thomas A. Berson and Thomas Beth, editors, *Local Area Network Security*, Lecture Notes in Computer Science, pages 89–101, Berlin, Heidelberg, 1989. Springer.
18. Ningning Hou and Yuanqing Zheng. CloakLoRa: A Covert Channel over LoRa PHY. In *2020 IEEE 28th International Conference on Network Protocols (ICNP)*, pages 1–11, Madrid, Spain, October 2020. IEEE.
19. Yu an Tan, Xiaosong Zhang, Kashif Sharif, Chen Liang, Quanxin Zhang, and Yuanzhang Li. Covert timing channels for iot over mobile networks. *IEEE Wireless Communications*, 25:38–44, 12 2018.
20. Arnab Kumar Biswas, Dipak Ghosal, and Shishir Nagaraja. A survey of timing channels and countermeasures. *ACM Comput. Surv.*, 50(1), mar 2017.
21. Ajay Kumar Nain and P. Rajalakshmi. A reliable covert channel over ieee 802.15.4 using steganography. *2016 IEEE 3rd World Forum on Internet of Things, WF-IoT 2016*, pages 711–716, 2017.
22. Nilufer Tuptuk and Stephen Hailes. Covert channel attacks in pervasive computing. In *2015 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 236–242, March 2015.
23. David Martins and Hervé Guyennet. Steganography in mac layers of 802.15.4 protocol for securing wireless sensor networks. *Proceedings - 2010 2nd International Conference on Multimedia Information Networking and Security, MINES 2010*, pages 824–828, 2010.
24. Ricardo Severino, João Rodrigues, and Luis Lino Ferreira. Exploring timing covert channel performance over the ieee 802.15.4. In *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–8, 2022.
25. Xuhang Ying, Giuseppe Bernieri, Mauro Conti, and Radha Poovendran. Tacan: Transmitter authentication through covert channels in controller area networks. *Proceedings of the 10th ACM/IEEE International Conference on Cyber-Physical Systems*, pages 23–34, 3 2019.

26. Haijiang Xie and Jizhong Zhao. A lightweight identity authentication method by exploiting network covert channel. page 10.
27. Maximilian Köstler, Florian Kauer, Tobias Lübkert, and Volker Turau. Towards an open source implementation of the ieee 802.15.4 dsme link layer. In Juergen Scholz and Alexander von Bodisco, editors, *Proceedings of the 15. GI/ITG KuVS Fachgespräch Sensornetze*, page 4. University of Applied Sciences Augsburg, Dept. of Computer Science, September 2016.