applied sciences

MDPI

Article

Stealthy Messaging: Leveraging Message Queuing Telemetry Transport for Covert Communication Channels

Sara Lazzaro and Francesco Buccafurri *

Department DIIES, University Mediterranea of Reggio Calabria, Via Università 25, 89122 Reggio Calabria, Italy; sara.lazzaro@unirc.it
* Correspondence: bucca@unirc.it

Abstract: Covert channel methods are techniques for improving privacy and security in network communications. These methods consist of embedding secret data within normal network channels, making it more difficult for unauthorized parties to detect such data. This paper presents a new approach for creating covert channels using the Message Queuing Telemetry Transport (MQTT) protocol, widely used in the context of the Internet of Things (IoT). The proposed method exploits storage channels by altering the field length of MQTT messages. Our solution leverages well-known one-way mathematical functions to ensure that data remain hidden from third parties observing the MQTT stream. In this way, we ensure that not only the content of the communication is preserved but also that the communication itself takes place. We conducted a security analysis to show that our solution offers the above-mentioned property even against severe threats, such as an adversary being able to observe all the messages exchanged in the network (even in the clear). Finally, we conducted an overhead analysis of our solution both in terms of the time required to perform the required operations and of the bytes to send. Our study shows that our solution adds no significant time overhead, and the additional overhead in terms of transmitted bytes remains within acceptable limits.

Keywords: MQTT; cover channel; censorship

Citation: Lazzaro, S.; Buccafurri, F. Stealthy Messaging: Leveraging Message Queuing Telemetry Transport for Covert Communication Channels. *Appl. Sci.* **2024**, *14*, 8874. https://doi.org/10.3390/app14198874

Academic Editor: Pedro Couto

Received: 6 August 2024
Revised: 24 September 2024
Accepted: 30 September 2024
Published: 2 October 2024

1. Introduction

The way information is shared has significantly changed due to the advancement of communication technologies. These technologies enable interaction across several platforms and devices [1–4]. However, there are increasing concerns about individual privacy. Additionally, in some contexts, communications can be restricted due to the presence of a censor that is able to monitor the entire network traffic. In such contexts, ensuring secure and undetectable communication is crucial, as traditional methods may expose sensitive information [5–7].

For example, standard encryption-based solutions only conceal the exchanged content but not the act of communication itself; i.e., an adversary knows the communicating parties. Additionally, an adversary can analyze metadata such as the timing, frequency, and patterns of communication to infer sensitive information. Alternative solutions consist of anonymous communication protocols [8–11] whose aim is to hide the identities of the communicating parties and metadata. However, these approaches often require a complex network setup involving a high number of communicating parties. Additionally, the use of such protocols may act as a red flag to censors, who are capable of detecting their distinct traffic patterns, making it easier for them to block or restrict the use of these methods [5–7].

In recent years, covert channel methods [12,13] have gained increasing attention as a means to enhance privacy and security. These methods enable the hiding of messages within regular communication channels. This makes it difficult for third parties to detect the presence of any secret communications.

Covert channels can be classified into two types: storage channels and timing channels [14]. Storage channels hide data within network packets, such as the headers or the payload [15,16]. For example, unused header fields can be manipulated to carry hidden messages. Instead, timing channels alter the timing of packet transmissions to encode information [17,18]. For instance, the presence or absence of a packet at a specific time can be encoded by a binary value, thus creating a hidden communication channel within the normal flow of traffic.

This paper proposes a novel approach to create covert communication by leveraging the construction of a storage channel within MQTT messages. Specifically, our solution leverages the field length within these messages to exfiltrate data, which are in turn related to the actual data through known one-way mathematical functions. Through this method, we aim to allow for the secret exchange of information between parties while obfuscating the communication itself. This approach enables safe information exchange in high-risk environments. For instance, it can bypass censorship activities since the information is completely hidden in covert channels carried by commonly adopted protocols. In this work, we adopt the MQTT protocol as a carrier protocol since it is commonly used in the IoT context. Therefore, with the increased spread of MQTT, covert communications can be embedded within regular MQTT traffic without raising any suspicion.

Our overhead analysis shows that our solution does not impose any significant time cost for the required operations. However, there is some overhead in terms of the number of bytes transmitted. Nevertheless, this overhead appears to be acceptable.

Finally, our security analysis shows that our solution is effective in protecting the transmission of information in high-risk environments, where attackers can observe all network traffic, even in the clear. In addition, it is effective against more powerful adversaries in the middle of the communication that may attempt to tamper with the messages exchanged between publishers and subscribers. This also includes compromised brokers natively in the middle of the communication.

To conclude, we summarize the main contributions of this paper.

- We propose a novel covert communication method by utilizing MQTT messages to build a storage channel for secure data exfiltration.
- We provide an analysis of the overhead introduced by our solution both in terms of time and bytes sent. Overall, the overhead required appears to be acceptable.
- We provide a security analysis of the proposed approach. This analysis shows that our solution offers security guarantees in scenarios with network monitoring and censorship without raising suspicion due to the common usage of the MQTT protocol in IoT environments.

The remainder of this document is structured as follows. In Section 2, we review the relevant literature on covert communication methods. In Section 3, we provide background information on MQTT. Section 4 describes the motivations and potential use cases for our approach. Section 5 presents our approach. Then, we illustrate our approach by referring to a numerical example in Section 6, while in Section 7, we provide an analysis of the overhead required by our solution. In Section 8, we analyze the security of the proposed approach. Finally, Section 9 concludes the paper.

2. Related Works

In this section, we review the relevant literature on covert communication methods that can be embedded in several protocols for the IoT context.

Several works have shown how out-of-band covert channels can be implemented in the IoT context [19]. For instance, light can be considered an effective carrier for exfiltrating data [20,21] since it is possible to exploit the inability of the human eye to detect low-contrast and fast flickering images. Through this technique, plain LEDs, which are ubiquitous in IoT nodes and industrial equipment [22–25], can also be used to build covert channels [26]. Instead, [27,28] exploit smart bulbs to transmit covert channels by either modulating color, brightness, or power utilization.

Different techniques are reported in [29,30], where the authors investigate thermal covert channels that exploit heat emissions from components. Specifically, [29] demonstrate that by exploiting the core design principles of the BLE link-layer protocol combined with well-timed intensive CPU calculations, it is possible to mount a thermal covert channel between two devices.

Additionally, [31,32] focus on acoustic covert channels where devices use inaudible sound waves (for humans) to transmit data to microphones in the vicinity.

Other covert channel techniques exploit the physical communication layer, i.e., they exploit the hardware and physical properties of a system to transmit hidden data outside the normal communication channels [33–35]. An example of these techniques can be found in [36,37], where the authors show that radio frequency signals can be used to carry covert information. Similarly, in [38], the authors show that IEEE 802.15.4 links can also be used as steganographic carriers by hiding information in the direct spread spectrum sequence modulation.

We observe that the limitation of these techniques lies in the fact that the transmission occurs over a short range; i.e., physical proximity to the device emitting the channel is required.

A commonly adopted technique to create covert channels that can be exploited at the network level consists in manipulating packet timing and network traffic patterns to stealthily transmit data [39]. For instance, a covert channel can be built by altering the order of packets in a network stream, e.g., a given permutation of packet order could represent a binary value. This method exploits the fact that many network protocols are resilient to minor reordering, making the changes less noticeable [40]. On the other hand, the transmission rate of packets can be modulated to encode information. By switching between different data rates, a sender can embed covert signals within the timing of the packet flow [41]. For instance, a high transmission rate might represent a binary '1', while a low transmission rate represents a binary '0'. Additionally, intentional packet loss or manipulated data re-transmission of packets can be used as a means to build covert channels [42]. For instance, packets can be selectively dropped according to a predetermined pattern that in turn can be later decoded by the receiver. Alternatively, by triggering re-transmissions, a sender can embed information in the timing and frequency of these events.

Alternative techniques to create covert channels consist of hiding data within the content of network packets [15,16]. These techniques commonly leverage unused data fields in protocol headers, as well as other packet components, to embed hidden messages. For instance, at the IP level [43,44], the identification field that is used for fragmenting packets can be used to transmit covert data without affecting the packet's delivery. Similarly, the type of service field can be used to embed covert data, since this field is rarely used in modern networks. Alternatively, TCP sequence numbers are used for tracking the order of packets [45,46].

All these methods take advantage of common practices in network protocols which help covert channels to remain undetected. However, these techniques are difficult to be adopted in MQTT. This is due to the fact that, as explained in Section 3, in this protocol, data are not directly transmitted from the sender to the recipient, but the communication is always mediated by a broker. Therefore, ad hoc solutions tailored for MQTT are needed to obfuscate data transmission.

3. Background

MQTT is a client–server publish/subscribe messaging transport protocol [47]. The message exchange involves two types of agents: MQTT clients and MQTT brokers. MQTT clients can be either publishers (information producers) or subscribers (information consumers), and they can perform both roles simultaneously. The architecture of MQTT is depicted in Figure 1.

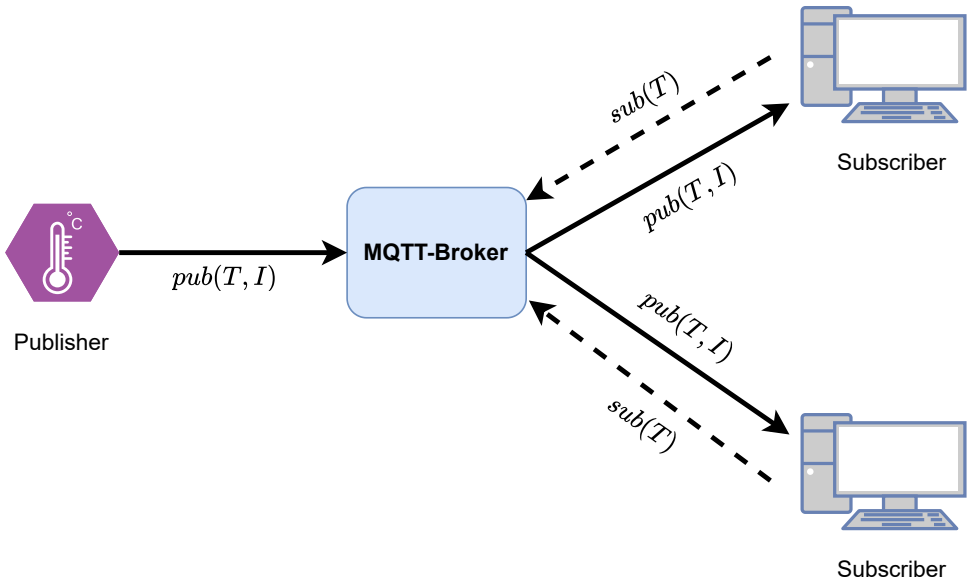

Figure 1. MQTT architecture.

The MQTT protocol requires that information provided by a publisher be associated with a topic that categorizes the information. Subscribers express interest in receiving certain information by specifying its topic.

Unlike traditional client–server architectures, MQTT uses a broker to mediate communication between publishers and subscribers rather than direct communication. This process is illustrated in Figure 1. Subscribers interested in a given topic (e.g., T) send a subscribe message $sub(T)$ to the broker, specifying the topic. When a publisher sends a message to the broker, it includes the information I and the topic T to which the information should be published. The broker then forwards the message to all subscribers interested in that topic.

MQTT messages have minimal overhead, with a protocol header of only 2 bytes, and the payload size can be up to 268,435,456 bytes. These characteristics make MQTT suitable for low-bandwidth networks and resource-constrained clients, such as IoT devices. To conclude this section, we provide some details on the structure of an MQTT packet. This packet, depicted in Figure 2, consists of a fixed header, variable header, and payload (which contains the actual message content being transmitted).

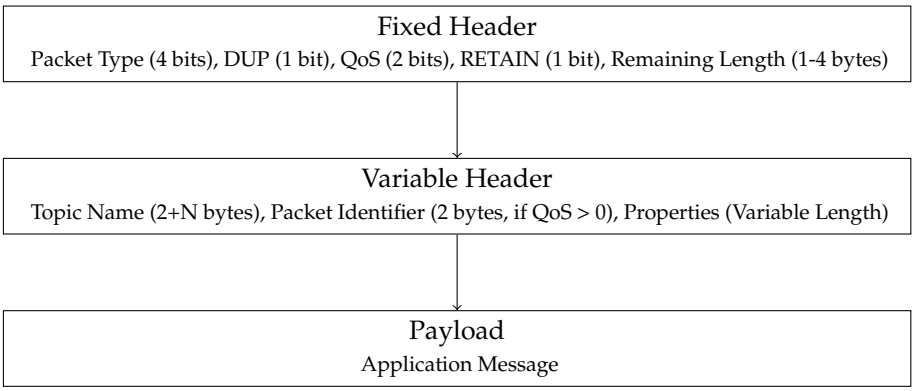

Figure 2. General MQTT packet structure.

The main fields of the header are as follows:

- Packet type: Identifies the type of packet, e.g., whether it is a publish packet.
- Remaining length: Length of the variable header and payload.
- Topic name: UTF-8 string that indicates the topic to which the message is published.

- Properties: User-defined properties that add metadata to MQTT messages in order to transmit additional user-defined information.

4. Motivations

In this section, we outline the motivations behind our work and the potential application domains where it can be adopted. We recall that this work aims to obfuscate the communication between two parties so that no external eavesdropper can intercept the communication between two parties or even determine whether communication is happening.

There are several potential use cases for our approach. For instance, it can be adopted in IoT environments where devices need to communicate very sensitive information, such as information revealing personal preferences and user behavior. Additionally, companies may protect their proprietary information and internal communications by utilizing our solution. This ensures that even if communication channels are being monitored, the data remain secure and undetectable.

Generally speaking, achieving this goal is fundamental in contexts where high privacy levels are required, particularly in environments subject to censorship and surveillance, where secure and undetectable communication is crucial. Indeed, in many regions, governments and authorities impose strict controls on internet use and actively monitor communication channels to suppress dissent and control the flow of information. These regimes often employ surveillance techniques by acting as global adversaries able to observe all traffic through the network. They implement extensive filtering systems, deep packet inspection (DPI), and other monitoring tools [48] to detect and block unwanted communications.

Standard encryption-based solutions, while protecting data content, can still be identified by such systems. Indeed, these solutions do not conceal the act of communication itself but just the exchanged context. On the other hand, in several environments, it is crucial not only to protect the content of communication but also to hide the parties that are communicating.

Alternative solutions involve the use of anonymous communication protocols [49] that are designed to hide the identities of the communicating parties and the metadata associated with the communication. These protocols are effective in hiding the communication between parties, even against global adversaries capable of observing all network traffic.

However, the use of anonymous communication protocols can itself be a red flag. Censors are often capable of detecting the presence of such protocols due to their distinct traffic patterns and signature behaviors. The fact that an anonymous protocol is being used can alert the censor to potentially suspicious activity. As a result, censors can block or restrict the use of these protocols, thereby preventing parties from utilizing them for secure communication.

This highlights the need for communication methods that blend seamlessly with regular, non-sensitive traffic. Our proposal is to leverage the MQTT protocol since it is commonly used in IoT applications. Specifically, our solution allows parties to exchange information through covert channels blended into MQTT packets.

The MQTT protocol is commonly used in IoT applications due to its lightweight nature and efficiency in handling devices with limited resources. This ubiquity acts as a cloak for secret communications, as MQTT traffic is generally considered benign and routine. Therefore, by inserting hidden messages in MQTT traffic, we can effectively hide the existence of communication between two parties, i.e., a publisher and a subscriber. This method also benefits from the architecture of MQTT, where publishers and subscribers never communicate directly, but the communication is always mediated by a broker. Additionally, in MQTT, communication on a topic can occur between multiple parties.Indeed, when a client publishes on a given topic, the broker broadcasts the messages to all the subscribers interested in that topic. Additionally, on each topic, there can be multiple publishers. This mechanism is crucial to obscure the direct connection between two parties [50,51].

In Table 1, we summarize the security features offered by the three approaches discussed in this section, i.e., encryption-based, anonymous communication, and covert

channel. We denote by data content the protection of the message, by sender identity the inability to identify the sender of a message (among other potential senders), and by sender activity the inability to detect that the sender is secretly sharing some information.

Table 1. Security feature comparison of encryption-based, anonymous communication, and covert channel approaches.

Approach	Data Content	Sender Identity	Sender Activity
Encryption-based	yes	no	no
Anonymous communication	yes	yes	no
Covert channel	yes	yes	yes

5. The Proposed Approach

In this section, we first provide an overview of the design principles of our solution. Then, we formally describe our solution.

5.1. Design Overview

Consider a publisher and a set of subscribers who wish to communicate secretly. The publisher and the subscribers share a list of public brokers and a list of topics on which they want to publish. Realistically, both brokers and topics can be obtained from Shodan [52,53]. We do not have specific requirements for topics and brokers, except that topics should be chosen to accommodate payloads of varying sizes (for example, topics such as "temperature", in which data are generally very short and fixed in size, should be excluded).

Suppose M is the secret message being transmitted to the subscribers. Let d be its length. The idea is to design a covert channel between the publisher and the authorized subscribers. Indeed, the covert channel can be detected only by those subscribers who own a given secret key. We assume that this key is pre-shared among the authorized parties.

The covert channel is based on a combination of the transmission of k messages generated on the basis of M, which we call *parts*, with the reverse of k modular exponentiations, feasible only with the knowledge of the secret key.

Specifically, the k parts are obtained first by generating $k-1$ random messages of the same size as M. Then, the k^{th} part is obtained as the bit-wise exclusive or between the $k-1$ random parts and the message M. Consider that, thanks to the properties of the exclusive or (denoted by $\oplus$), for which $A \oplus A = 0$ and 0 is the null element for the operator, if a subscriber receives the k parts, they can easily reconstruct M simply by computing the exclusive or between all the k parts. But, this would be a very weak covert channel. Therefore, the k parts cannot be sent clearly. They are sent in an indirect way. For each them, first, a modular exponentiation, whose exponent is the secret key, is applied. Then, a random message whose length is equal to the above exponentiation is sent. Thus, the subscriber can detect a part among all the received messages by identifying the MQTT message x for which there exists a string y of length d such that the modular exponentiation of this string with the secret key has a length equal to the length of x. The identified part is just such a string, y. Once the subscribers have received k parts, they can compute M, then the hidden transmission is achieved. Observe that only the authorized subscribers (owing the secret key) are able to identify the k parts and then to receive M. For the other subscribers, there is no way to obtain y, because they do not know the exponent. To make this infeasible, it suffices to set d to a small value and the length of the key to a large value. This way, the authorized subscribers have to perform only 2^d attempts (in the worst case) per received message. d should be fixed in such a way that the computation is feasible. Instead, an unauthorized subscriber should perform $2^d \cdot 2^s$ attempts, where s is the length of the key. Therefore, the standard size of cryptographic keys (for example, 128 bits) is enough to make the above operation infeasible, which is also burdened by the multiplicative factor 2^d.

5.2. Formal Description

In the following, we formally describe our protocol.

Whenever the publisher wants to send a message M, it generates k parts $P_1, \ldots, P_k$ as follows:

- $P_1, \ldots, P_{k-1}$ are randomly extracted of a size equal to the size of M.
- $P_k = M \oplus P_1 \oplus \ldots \oplus P_{k-1}$, where $\oplus$ denotes the XOR (exclusive or) operator.

We observe that the introduction of the XOR operator causes P_k to appear random.

Next, the publisher and subscribers must choose the broker and topic on which to publish each part. We denote as B the list of brokers whose size is $|B|$ and by $|T_B|$ the number of topics for each broker. We denote by $max_B(|T_B|)$ the maximum number of topics among all the brokers.

The topics and the brokers are extracted as follows. The publisher and the subscribers share a seed of a Pseudo-Random Number Generator (PRNG). For each extracted value of the PRNG, the first $k \log_2(|B|)$ bits index the k brokers to which the parts will be sent. The subsequent $k \log_2(max_B(|T_B|))$ bits index the k topics of the previously chosen brokers on which the parts will be published.

We focus now on a single part P_i to be sent to a single broker on a specific topic. It will be sent over a covert channel that exploits the length field of the MQTT header. We assume that the parts are small and therefore that the message size is small.

The publisher and the subscribers also share a prime modulus n and an exponent a. Consider the group $\mathbb{Z}_n^*$. The publisher calculates the following:

$$L = ({P_i}^a \mod n) \mod l$$

where l is the maximum number of bytes of an MQTT message (i.e., 268,435,456). This value L will be inserted into the length field of an MQTT packet header. The payload of the MQTT packet will be dummy data satisfying this length.

To publish the other parts, the publisher will repeat the above operations by extracting the subsequent values from the PRNG.

Now let us focus on the operations of a subscriber. The subscriber extracts the same value from the PRNG (since it shares the same secret with the publisher). This allows the subscriber to subscribe to the same brokers and topics as the publisher. However, on these topics, external publishers may also be publishing, and the subscriber (in the standard MQTT protocol) cannot identify which publisher owns each message. Therefore, to retrieve the parts $P_1, \ldots, P_k$, the subscriber performs the following operations for each packet received on that topic and broker.

Specifically, using the same a and n, the subscriber verifies all possible values of P_i' such that:

$$L' = \left({P_i'}^a \mod n\right) \mod l$$

where L' is the length of the current packet. In other words, the subscriber performs brute force on all possible values of P_i'. As mentioned above, the length of P_i' is sufficiently small to make such brute force feasible. If this equality does not hold for any P_i', then that packet does not contain any covert channel and can be ignored. This operation is performed for each broker and topic extracted from the current PRNG value. In this way, the subscriber can retrieve k parts. This allows them to reconstruct the original message by leveraging the XOR properties as follows:

$$M = P_1 \oplus P_2 \oplus \ldots \oplus P_k$$

We conclude this section by observing that since a and n are known, the subscriber can precompute all possible values of P_i' in advance. However no rainbow table attacks can be performed, i.e., the transmission of the same message M is not always related to

the same length L. This is because each time the publisher can choose different parties to represent the same message M.

6. Numerical Example

To illustrate the approach, we consider the publisher transmitting the letter "H" as a message.

6.1. Step 1: ASCII Representation and Binary Conversion:

The letter "H" in ASCII is 72. The binary representation of 72 is $M = 01001000$.

6.2. Step 2: Split the Binary Message into k Parts:

Assume $k = 3$.
The publisher computes the parts P_1, P_2, P_3 as follows.

- P_1 and P_2 are randomly chosen.
- $P_3 = M \oplus P_1 \oplus P_2$.

For example, we consider:

- $P_1 = 00101101$ (45 in decimal).
- $P_2 = 11010011$ (211 in decimal).

Then, the publisher computes P_3:

$$P_3 = 01001000 \oplus 00101101 \oplus 11010011 = 10110110$$

6.3. Step 3: Publish Each Part Using a Covert Channel

We recall the maximum MQTT message size: $l = 268,435,456$.
To be concrete, we consider that the publisher knows 256-bit values for n and a:

- Prime modulus $n = 898026010364896354127008642528464072795849174768208717087728218521211266225 33$
- Exponent $a = 1861836248542572424$

Then, it calculates the length L for each part:
For P_1:

$$L_1 = ({P_1}^a \mod n) \mod l$$

$$L_1 = (45^a \mod n) \mod 268,435,456$$

Assume $L_1 = 264313832$.
For P_2:

$$L_2 = ({P_2}^a \mod n) \mod l$$

$$L_2 = (211^a \mod n) \mod 268,435,456$$

Assume $L_2 = 89686678$.
For P_3:

$$L_3 = ({P_3}^a \mod n) \mod l$$

$$L_3 = (139^a \mod n) \mod 268,435,456$$

Assume $L_3 = 144660972$.
Therefore, the lengths L_1, L_2, L_3 will be 264313832, 89686678, and 144660972, respectively.

6.4. Step 4: Publish to Brokers and Topics

The publisher sends these values to different brokers and topics extracted according to the PRNG using the lengths in the MQTT header to transmit the parts covertly.

6.5. Step 5: Subscriber Retrieves the Parts

The subscriber, knowing n and a, retrieves the parts by examining the lengths of incoming MQTT packets:

- For the packet with length 264313832, the subscriber brute-forces 8 bits to find $P_1 = 45$ (00101101).
- For the packet with length 89686678, the subscriber brute-forces 8 bits to find $P_2 = 211$ (11010011).
- For the packet with length 144660972, the subscriber brute-forces 8 bits to find $P_3 = 139$ (10001011).

6.6. Step 6: Reconstruct the Original Message

The subscriber reconstructs the original message as follows:

$$M = P_1 \oplus P_2 \oplus P_3$$

$$M = 00101101 \oplus 11010011 \oplus 10110110 = 01001000$$

Converting 01001000 back to ASCII, we obtain the letter "H".

7. Overhead Analysis

To evaluate the efficiency of the proposed approach, we conducted several simulations, varying the key parameters that influence performance. The overall performance of the system was measured by tracking the execution time for the main operations required by our approach. The measurements were performed by running a Python simulation on a personal computer equipped with a 2.8 GHz Intel i7-1165G7 CPU and 16 GB of RAM. The obtained results are reported in Table 2.

Table 2. Execution time of the main operations required for our approach.

	Operations	**Execution Time**
Publisher side	Selection of k parts	$0.908\ \mu s * k$
	Length computation	$1.997\ \mu s * k$
Subscriber side	Setup phase	27.8 s (for 20 bits)
	Message reconstruction	$0.823\ \mu s * k$

We now discuss the timings resulting from our simulations by examining the operations executed on the publisher side and the subscriber side.

On the publisher side, the overhead required by our solution consists first of the selection of k parties and then of the computation of the length for each part.

It is observed that the operations performed by the publisher typically require times on the order of tens of microseconds, multiplied by the number of parts k. Specifically, the selection of the k parts involves randomly extracting $k - 1$ parts and then computing the k-th part as the XOR of all the other parts and the message to be transmitted. In contrast, the length computation operation involves calculating k message lengths to be transmitted (through exponentiation operations).

Moreover, the number of parts k determines the number of brokers where the parts are published. The higher this number, the more brokers would need to collude to reconstruct the final message. However, selecting a value of k in the range of about ten parts is typically sufficient to ensure that these randomly selected brokers do not all collude. Therefore, it can be concluded that the operations on the publisher's side are efficient, with negligible overhead.

On the subscriber's side, two main operations are involved: the setup phase and the message reconstruction. Message reconstruction is highly efficient, typically requiring times on the order of tens of microseconds multiplied by the number of parts k, as it consists only of XORing the parts recovered from the k brokers. On the other hand, the setup operation (which involves generating the lengths corresponding to all possible parts—essentially a brute-force operation over all possible parts) has times that grow exponentially with the

number of bits in the message. As shown in the table, if the message has 20 bits, the time required is on the order of seconds.

It is important to note that while the setup operation takes longer, it is a one-time process, as it is used to precompute the possible message lengths.

In conclusion, we discuss the overhead introduced by our solution in terms of the transmitted bytes. Recall that M represents the message to be reconstructed, whereas the actual transmitted messages are those whose lengths depend on the values of the selected parts. In the example shown in Section 6, for transmitting the message $M = "H"$ with $k = 3$, three messages of lengths 264, 313, 832, 89, 686, 678, and 144, 660, 972 bytes are transmitted.

To estimate the average length of the messages sent, we conducted a simulation where, over 10,000 runs, we varied the modulus n and the exponent a while keeping the message length $|M| = 8$ bits. Figure 3 shows how the computed lengths are distributed based on the simulation. It can be observed that the distribution of lengths is uniform in the range from 0 to 268, 435, 456 bytes. Due to the properties of a uniform distribution, the average length of the transmitted messages is approximately 134, 217, 728 bytes.

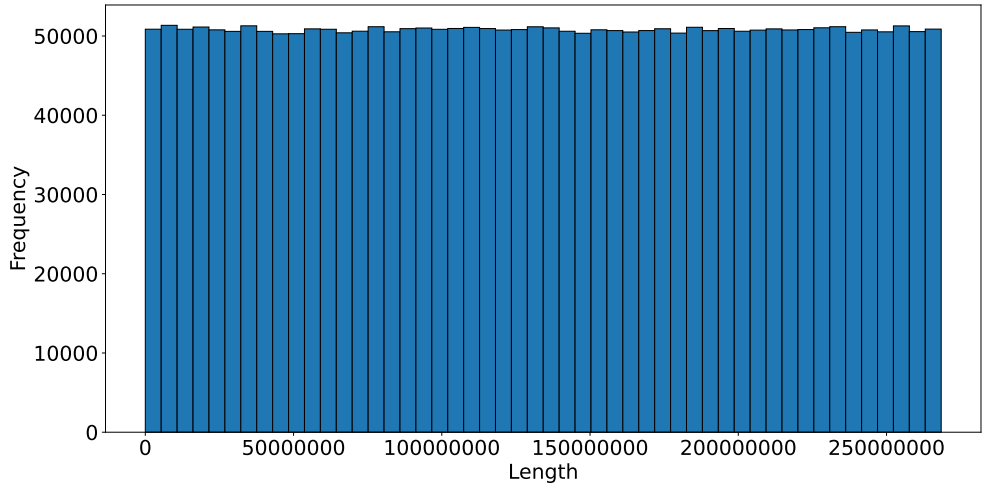

Figure 3. Length distribution.

Finally, although this overhead is acceptable for practical purposes, the total bytes transmitted can be reduced by decreasing the value of k, i.e., the number of actual messages sent.

8. Security Analysis

In this section, we discuss the security features offered by our solution.

The security of our solution is based on three components:

1. Message splitting: Each message to be sent is split in k parts. An adversary cannot reconstruct M without knowing all the parts.
2. Covert channel: Each part is sent through a cover channel exploiting the length field of the MQTT packet. This makes it harder to distinguish between a publisher transmitting actual data and dummy data.
3. PRNG: The use of a shared seed for a PRNG ensures synchronized and unpredictable selection of brokers and topics for communication. Moreover, secure and unpredictable parameters for the execution of the protocol are extracted through the PRNG.

Our threat model includes three possible threats:

- Eavesdropping and traffic analysis: An adversary intercepts communication to gain information about the transmitted message.
- Man-in-the-M = middle attacks [54]: An adversary intercepts, modifies, or injects messages between the publisher and subscribers.
- Broker compromise [55]: An adversary gains control over one or more brokers.

In the following, we show how our proposal addresses this threat model.

Eavesdropping and traffic analysis:

Message splitting makes traffic analysis less effective. Indeed, the original message M cannot be reconstructed when the adversary can retrieve up to $k-1$ parts.

In addition, the attacker cannot identify meaningful information within intercepted packets due to the covert channel. The use of the length field to encode parts of the message means that the actual payload appears as random or dummy data. This reduces the likelihood of detection by intermediaries or adversaries monitoring the traffic.

Finally, the adoption of PRNG for broker and topic selection mitigates the risk of traffic analysis. Indeed, the publisher and subscribers share the seed of the PRNG and are able to independently determine the same brokers and topics for communication. On the other hand, without such a seed, the brokers and topics are unpredictable for an adversary.

Man-in-the middle attacks:

The adoption of a hlcovert channel prevents data interception and tampering. Indeed, any modification to the length field or payload would render the part unusable, as it would fail the brute force verification performed by the subscriber.

The only way for the adversary to compute a valid length field should be to recover the parameters n and a. However, an adversary cannot do so, since there are no feasible means to guess or compute these two parameters.

Broker compromise:

The brokers adopted to transmit parts are randomly chosen through the PRNG. Then, it is difficult for the adversary to detect the right brokers to compromise. In addition, message splitting requires the adversary to break at least k brokers to recover the original message, M.

9. Conclusions

in this work, we developed a covert channel specifically for the MQTT protocol with the goal of enabling the secure transmission of sensitive information in situations that demand high levels of privacy. Our approach leverages the construction of MQTT messages with chosen lengths as a mean for covert data transmission. Through a security analysis, we show that our solution effectively protects data transmission in high-risk environments where adversaries are capable of monitoring all network traffic, including unencrypted information. Not only does our technique obscure the message content, but it also conceals the very presence of secret communication, thereby reinforcing both security and privacy. The overhead analysis also shows the feasibility of our approach and traces the way for real-life implementation in future work.

Author Contributions: Conceptualization, F.B. and S.L.; methodology, F.B. and S.L.; software, S.L.; validation, S.L.; formal analysis S.L.; investigation, F.B. and S.L.; resources S.L.; data curation, S.L.; writing—original draft preparation S.L.; writing—review and editing, S.L.; visualization, S.L.; supervision, F.B.; project administration, F.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

MQTT	Message Queuing Telemetry Transport
IoT	Internet of Things
DPI	Deep Packet Inspection
PRNG	Pseudo-Random Number Generation
QoS	Quality of Service

References

1. Kim, J.; Lee, J.; Kim, J.; Yun, J. M2M Service Platforms: Survey, Issues, and Enabling Technologies. *IEEE Commun. Surv. Tutor.* **2014**, *16*, 61–76. [CrossRef]
2. Buccafurri, F.; De Angelis, V.; Lazzaro, S.; Pugliese, A. Enforcing security policies on interacting authentication systems. *Comput. Secur.* **2024**, *140*, 103771. [CrossRef]
3. Hofmann, R.; Boano, C.A.; Römer, K. X-Burst: Enabling Multi-Platform Cross-Technology Communication between Constrained IoT Devices. In Proceedings of the 2019 16th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON), Boston, MA, USA, 10–13 June 2019; pp. 1–9. [CrossRef]
4. Lee, C.T.; Chen, L.B.; Chu, H.M.; Hsieh, C.J.; Liang, W.C. An Internet of Things (IoT)-Based Master-Slave Regionalized Intelligent LED-Light-Controlling System. *Appl. Sci.* **2022**, *12*, 420. [CrossRef]
5. Celik, Z.B.; Babun, L.; Sikder, A.K.; Aksu, H.; Tan, G.; McDaniel, P.; Uluagac, A.S. Sensitive information tracking in commodity {IoT}. In Proceedings of the 27th USENIX Security Symposium (USENIX Security 18), Baltimore, MD, USA, 15–17 August 2018; pp. 1687–1704.
6. Buccafurri, F.; De Angelis, V.; Labrini, C. A Privacy-Preserving Solution for Proximity Tracing Avoiding Identifier Exchanging. In Proceedings of the 2020 International Conference on Cyberworlds (CW), Caen, France, 29 September –1 October 2020; pp. 235–242. [CrossRef]
7. Buccafurri, F.; De Angelis, V.; Idone, M.F.; Labrini, C. A Distributed Location Trusted Service Achieving k-Anonymity against the Global Adversary. In Proceedings of the 2021 22nd IEEE International Conference on Mobile Data Management (MDM), Toronto, ON, Canada, 15–18 June 2021; pp. 133–138. [CrossRef]
8. Piotrowska, A.M.; Hayes, J.; Elahi, T.; Meiser, S.; Danezis, G. The Loopix Anonymity System. In *Proceedings of the 26th USENIX Security Symposium (USENIX Security 17)*; USENIX Association: Vancouver, BC, Canada, 2017; pp. 1199–1216.
9. Buccafurri, F.; De Angelis, V.; Idone, M.F.; Labrini, C. A protocol for anonymous short communications in social networks and its application to proximity-based services. *Online Soc. Netw. Media* **2022**, *31*, 100221. [CrossRef]
10. van den Hooff, J.; Lazar, D.; Zaharia, M.; Zeldovich, N. Vuvuzela: Scalable private messaging resistant to traffic analysis. In *Proceedings of the 25th Symposium on Operating Systems Principles*; Association for Computing Machinery: New York, NY, USA, 2015; SOSP'15, pp. 137–152. [CrossRef]
11. Young, A.L.; Yung, M. The drunk motorcyclist protocol for anonymous communication. In Proceedings of the 2014 IEEE Conference on Communications and Network Security, San Francisco, CA, USA, 29–31 October 2014; pp. 157–165. [CrossRef]
12. Wendzel, S.; Zander, S.; Fechner, B.; Herdin, C. Pattern-Based Survey and Categorization of Network Covert Channel Techniques. *ACM Comput. Surv.* **2015**, *47*, 1–26. [CrossRef]
13. Tian, J.; Xiong, G.; Li, Z.; Gou, G. A Survey of Key Technologies for Constructing Network Covert Channel. *Secur. Commun. Netw.* **2020**, *2020*, 8892896. [CrossRef]
14. Kemmerer, R. A practical approach to identifying storage and timing channels: twenty years later. In Proceedings of the 18th Annual Computer Security Applications Conference, Las Vegas, NV, USA, 9–13 December 2002; pp. 109–118. [CrossRef]
15. Tsai, C.R.; Gligor, V.; Chandersekaran, C. On the identification of covert storage channels in secure systems. *IEEE Trans. Softw. Eng.* **1990**, *16*, 569–580. [CrossRef]
16. Tsai, C.R.; Gligor, V.D.; Chandersekaran, C.S. A Formal Method for the Identification of Covert Storage Channels in Source Code. In Proceedings of the 1987 IEEE Symposium on Security and Privacy, Oakland, CA, USA, 27–29 April 1987; pp. 74–74. [CrossRef]
17. Cock, D.; Ge, Q.; Murray, T.; Heiser, G. The Last Mile: An Empirical Study of Timing Channels on seL4. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*; Association for Computing Machinery: New York, NY, USA, 2014; CCS'14, pp. 570–581. [CrossRef]
18. Gianvecchio, S.; Wang, H. Detecting covert timing channels: an entropy-based approach. In *Proceedings of the 14th ACM Conference on Computer and Communications Security*; Association for Computing Machinery: New York, NY, USA, 2007; CCS'07, pp. 307–316. [CrossRef]
19. Caviglione, L.; Merlo, A.; Migliardi, M. Covert Channels in IoT Deployments Through Data Hiding Techniques. In Proceedings of the 2018 32nd International Conference on Advanced Information Networking and Applications Workshops (WAINA), Krakow, Poland, 16–18 May 2018; pp. 559–563. [CrossRef]
20. Guri, M.; Hasson, O.; Kedma, G.; Elovici, Y. An optical covert-channel to leak data through an air-gap. In Proceedings of the 2016 14th Annual Conference on Privacy, Security and Trust (PST), Auckland, New Zealand, 12–14 December 2016; pp. 642–649. [CrossRef]

21. Loughry, J.; Umphress, D.A. Information leakage from optical emanations. *ACM Trans. Inf. Syst. Secur.* **2002**, *5*, 262–289. [CrossRef]
22. Lupia, F.; Lucchese, M.; Merro, M.; Zannone, N. ICS Honeypot Interactions: A Latitudinal Study. In Proceedings of the 2023 IEEE International Conference on Big Data (BigData), Sorrento, Italy, 15–18 December 2023; pp. 3025–3034. [CrossRef]
23. Liu, S.F.; Fan, Y.J.; Luh, D.B.; Teng, P.S. Organizational Culture: The Key to Improving Service Management in Industry 4.0. *Appl. Sci.* **2022**, *12*, 437. [CrossRef]
24. Longo, G.; Lupia, F.; Pugliese, A.; Russo, E. Physics-aware targeted attacks against maritime industrial control systems. *J. Inf. Secur. Appl.* **2024**, *82*, 103724. [CrossRef]
25. Lazzaro, A.; D'Addona, D.M.; Merenda, M. Comparison of Machine Learning Models for Predictive Maintenance Applications. In *Proceedings of the Advances in System-Integrated Intelligence*; Valle, M., Lehmhus, D., Gianoglio, C., Ragusa, E., Seminara, L., Bosse, S., Ibrahim, A., Thoben, K.D., Eds.; Springer: Cham, Switzerland, 2023; pp. 657–666.
26. Carrara, B.; Adams, C. Out-of-Band Covert Channels—A Survey. *ACM Comput. Surv.* **2016**, *49*, 1–36. [CrossRef]
27. Cronin, P.; Gouert, C.; Mouris, D.; Tsoutsos, N.G.; Yang, C. Covert Data Exfiltration Using Light and Power Channels. In Proceedings of the 2019 IEEE 37th International Conference on Computer Design (ICCD), Abu Dhabi, United Arab Emirates, 17–20 November 2019; pp. 301–304. [CrossRef]
28. Ronen, E.; Shamir, A. Extended Functionality Attacks on IoT Devices: The Case of Smart Lights. In Proceedings of the 2016 IEEE European Symposium on Security and Privacy (EuroS&P), Saarbruecken, Germany, 21–24 March 2016; pp. 3–12. [CrossRef]
29. Claeys, T.; Rousseau, F.; Simunovic, B.; Tourancheau, B. Thermal covert channel in bluetooth low energy networks. In *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*; Association for Computing Machinery: New York, NY, USA, 2019; WiSec'19, pp. 267–276. [CrossRef]
30. Chen, S.; Xiong, W.; Xu, Y.; Li, B.; Szefer, J. Thermal Covert Channels Leveraging Package-on-Package DRAM. In Proceedings of the 2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), Rotorua, New Zealand, 5–8 August 2019; pp. 319–326. [CrossRef]
31. Roy, N.; Hassanieh, H.; Roy Choudhury, R. BackDoor: Making Microphones Hear Inaudible Sounds. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*; Association for Computing Machinery: New York, NY, USA, 2017; MobiSys'17, pp. 2–14. [CrossRef]
32. Al Faruque, M.A.; Chhetri, S.R.; Canedo, A.; Wan, J. Acoustic Side-Channel Attacks on Additive Manufacturing Systems. In Proceedings of the 2016 ACM/IEEE 7th International Conference on Cyber-Physical Systems (ICCPS), Vienna, Austria, 11–14 April 2016; pp. 1–10. [CrossRef]
33. Lee, K.S.; Wang, H.; Weatherspoon, H. PHY Covert Channels: Can you see the Idles? In Proceedings of the 11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14), Seattle, WA, USA, 2–4 April 2014; USENIX Association: Seattle, WA, USA, 2014; pp. 173–185.
34. Classen, J.; Schulz, M.; Hollick, M. Practical covert channels for WiFi systems. In Proceedings of the 2015 IEEE Conference on Communications and Network Security (CNS), Florence, Italy, 28–30 September 2015; pp. 209–217. [CrossRef]
35. Jiang, Y.; Wang, L.; Chen, H.H.; Shen, X. Physical Layer Covert Communication in B5G Wireless Networks—Its Research, Applications, and Challenges. *Proc. IEEE* **2024**, *112*, 47–82. [CrossRef]
36. Li, Y.; Zhao, R.; Deng, Y.; Shu, F.; Nie, Z.; Aghvami, A.H. Harvest-and-Opportunistically-Relay: Analyses on Transmission Outage and Covertness. *IEEE Trans. Wirel. Commun.* **2020**, *19*, 7779–7795. [CrossRef]
37. Li, Y.; Aghvami, A.H. Covertness-Aware Trajectory Design for UAV: A Multi-Step TD3-PER Solution. In Proceedings of the ICC 2022—IEEE International Conference on Communications, Seoul, Republic of Korea, 16–20 May 2022, pp. 7–12. [CrossRef]
38. Nain, A.K.; Rajalakshmi, P. A reliable covert channel over IEEE 802.15.4 using steganography. In Proceedings of the 2016 IEEE 3rd World Forum on Internet of Things (WF-IoT), Reston, VA, USA, 12–14 December 2016; pp. 711–716. [CrossRef]
39. Tan, Y.a.; Zhang, X.; Sharif, K.; Liang, C.; Zhang, Q.; Li, Y. Covert Timing Channels for IoT over Mobile Networks. *IEEE Wirel. Commun.* **2018**, *25*, 38–44. [CrossRef]
40. Ahsan, K.; Kundur, D. Practical data hiding in TCP/IP. In Proceedings of the Workshop on Multimedia Security at ACM Multimedia, Juan-les-Pins, France, 6 December 2002; ACM Press: New York, NY, USA, 2002, Volume 2, pp. 1–8.
41. Cabuk, S.; Brodley, C.E.; Shields, C. IP covert timing channels: Design and detection. In *Proceedings of the 11th ACM Conference on Computer and Communications Security*; Association for Computing Machinery: New York, NY, USA, 2004; CCS'04, pp. 178–187. [CrossRef]
42. Servetto, S.; Vetterli, M. Communication using phantoms: Covert channels in the Internet. In Proceedings of the 2001 IEEE International Symposium on Information Theory (IEEE Cat. No.01CH37252), Washington, DC, USA, 29 June 2001; p. 229. [CrossRef]
43. Cabuk, S.; Brodley, C.E.; Shields, C. IP Covert Channel Detection. *ACM Trans. Inf. Syst. Secur.* **2009**, *12*, 1–29. [CrossRef]
44. Zander, S.; Armitage, G.; Branch, P. Covert channels in the IP time to live field. In Proceedings of the Australian Telecommunication Networks and Applications Conference (ATNAC), Melbourne, Australia, 4–6 December 2006 ; Volume 13, pp. 16–53.
45. Murdoch, S.J.; Lewis, S. Embedding Covert Channels into TCP/IP. In *Proceedings of the Information Hiding*; Barni, M., Herrera-Joancomartí, J., Katzenbeisser, S., Pérez-González, F., Eds.; Springer: Berlin/Heidelberg, Germany, 2005; pp. 247–261.

46. Sohn, T.; Seo, J.; Moon, J. A Study on the Covert Channel Detection of TCP/IP Header Using Support Vector Machine. In *Proceedings of the Information and Communications Security*; Qing, S., Gollmann, D., Zhou, J., Eds.; Springer: Berlin/Heidelberg, Germany, 2003; pp. 313–324.
47. OASIS. *MQTT*, Version 5.0; OASIS Standard, OASIS Open: Burlington, MA, USA, 2019.
48. Anselmi, G.; Mandalari, A.M.; Lazzaro, S.; De Angelis, V. COPSEC: Compliance-Oriented IoT Security and Privacy Evaluation Framework. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*; Association for Computing Machinery: New York, NY, USA, 2023.
49. Buccafurri, F.; Angelis, V.D.; Francesca Idone, M.; Labrini, C. WIP: An Onion-Based Routing Protocol Strengthening Anonymity. In Proceedings of the 2021 IEEE 22nd International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), Pisa, Italy, 7–11 June 2021; pp. 231–235. [CrossRef]
50. Eugster, P.T.; Felber, P.A.; Guerraoui, R.; Kermarrec, A.M. The many faces of publish/subscribe. *ACM Comput. Surv.* **2003**, *35*, 114–131. [CrossRef]
51. Greco, G.; Lupia, F.; Scarcello, F. The Tractability of the Shapley Value over Bounded Treewidth Matching Games. In Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, VIC, Australia, 19–25 August 2017; pp. 1046–1052. [CrossRef]
52. Andy, S.; Rahardjo, B.; Hanindhito, B. Attack scenarios and security analysis of MQTT communication protocol in IoT system. In Proceedings of the 4th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI), Yogyakarta, Indonesia, 19–21 September 2017; pp. 1–6. [CrossRef]
53. Lucchese, M.; Lupia, F.; Merro, M.; Paci, F.; Zannone, N.; Furfaro, A. HoneyICS: A High-interaction Physics-aware Honeynet for Industrial Control Systems. In *Proceedings of the 18th International Conference on Availability, Reliability and Security*; Association for Computing Machinery: New York, NY, USA, 2023; ARES '23. [CrossRef]
54. Lazzaro, S.; De Angelis, V.; Mandalari, A.M.; Buccafurri, F. Is Your Kettle Smarter Than a Hacker? A Scalable Tool for Assessing Replay Attack Vulnerabilities on Consumer IoT Devices. In Proceedings of the 2024 IEEE International Conference on Pervasive Computing and Communications (PerCom), Biarritz, France, 11–15 March 2024; pp. 114–124. [CrossRef]
55. Buccafurri, F.; De Angelis, V.; Lazzaro, S. MQTT-I: Achieving End-to-End Data Flow Integrity in MQTT. *IEEE Trans. Dependable Secur. Comput.* **2024**, *21*, 4717–4734. [CrossRef]