Covert Channels-based Stealth Attacks in Industry 4.0

C. Alcaraz, G. Bernieri[2], F. Pascucci[3]
J. Lopez[1], and R. Setola[4]
[1]Computer Science Department, University of Malaga,
Campus de Teatinos sn, 29071, Malaga,
[2]Department of Mathematics, University of Padua, Padua, Italy
[3]Department of Engineering, Roma Tre University, Rome, Italy
[4]Complex System & Security Lab, University Campus Bio-Medico, Rome, Italy
alcaraz@lcc.uma.es, bernieri@math.unipd.it, federica.pascucci@uniroma3.it
jlm@lcc.uma.es, r.setola@unicampus.it

Abstract

Industry 4.0 advent opens several cyber-threats scenarios originally designed for classic information technology, drawing the attention to serious risks for the modern industrial control networks. To cope with this problem, in this paper we address the security issues related to covert channels applied to industrial networks, identifying the new vulnerability points when information technologies converge with operational technologies such as edge computing infrastructures. Specifically, we define two signaling strategies where we exploit the Modbus/TCP protocol as target to set up a covert channel. Once the threat channel is established, passive and active offensive methodologies are further exploited by implementing and testing them on a real Industrial Internet of Things testbed. The experimental results highlight the potential damage of such specific threats and the easy extrapolation of the attacks to other types of channels in order to show the new risks for the Industry 4.0. Related to this, we discuss some countermeasures offering an overview of possible mitigation and defensive measures.

Keywords: Stealth attacks, covert channel, Industry 4.0, data exfiltration, and command and control.

1 Introduction

In recent years, the Industrial Control Systems (ICSs) have been exposed to the massive integration of Information and Communications Technologies (ICTs). With the appearance of the first Supervisory Control And Data Acquisition (SCADA) monolithic systems, conceived in the 60s, the networked generation was born [1]. Since then, industrial networks have been undergone numerous technical transformations to segment and protect the operational and manufacturing processes, leading to a new industrial revolution today. The new industrial paradigm, known as Industry 4.0 [2], foresees the use of ICTs for remote monitoring and control critical infrastructures, unifying the Operational Technologies (OTs) with the new Information Technologies (ITs) [3] such as edge computing infrastructures composed of cloud and fog servers [2].

Although this technical evolution is relevant for the modernization of control processes, new security challenges and risks arise within this new digital transformation era [2, 4]. Many of these challenges generally come from typical vulnerabilities of the cyber domains (e.g., accessible critical ports and services, irregular access control, lacks of isolation measures or uncontrolled network sections, lacks of auditing and accountability, irregularity in the governance processes, incompatibilities) [4], and they need to be properly managed 24/7 through security controls as specified by standards (e.g., the NISTIR 8183 [5]) or specific cybersecurity frameworks (e.g., [6]). Clear examples (Stuxnet, TRISIS, Flame, Night Dragon, BlackEnergy or Ex-Petr, Duqu [7, 8, 9]) have already shown the weak nature of the current ICSs to detect furtiveness attacks [9]. This paper envisages how OT networks may be compromised through specific IT techniques based on covert channels. According to Lampson, a covert channel consists of *"a channel of communications that is neither designed nor intended to transfer information"*, or particularly *"a process to share information in a manner that violates the system's security policies"* as clearly specified in [10].

Given this, the **main contribution** of this paper consists in (i) introducing several signaling strategies to create covert channels in Modbus/TCP[1] client-server-based networks, and (ii) leading several attacks to demonstrate the attack influence on the control routines. Concretely, the research aims to show how the success of the threat may unfortunately trigger a serious impact in the performance of critical underlying infrastructures and cause significant damages during the provision of their services. These risks can even become greater when IT-OT networks needs converge towards Industry 4.0 [4]. For that reason, Figure 1 illustrates the penetration points of today's industry and how the technological evolution may bring numerous security issues. For example, attackers may create covert channels between malicious peers, and exfilter or manipulate critical data (measurements, alarms or commands) managed by sensors working in 6LowPAN, gateways, edge computing servers [11], controllers or actuators.

Therefore, the remainder of this paper is structured as follows: Section 2 highlights the related work and the differences with the current work. Section 3 details the threat model along with attack techniques on covert channels. To validate the threat model, a practical experimentation is discussed in Section 4 considering the implication of a real Industrial Internet of Thing (IIoT) testbed. Finally, countermeasures are identified in Section 5, and conclusions and future work are widely described in Section 6.

2 Related work

Covert channels have attracted a lot of interest in the research community as testified by the large amount of literature on this topic [12]–[13]. Currently, different taxonomies

[1]Note that the choice of Modbus/TCP protocol is motivated in part by its large usage in industry and scientific community [1].

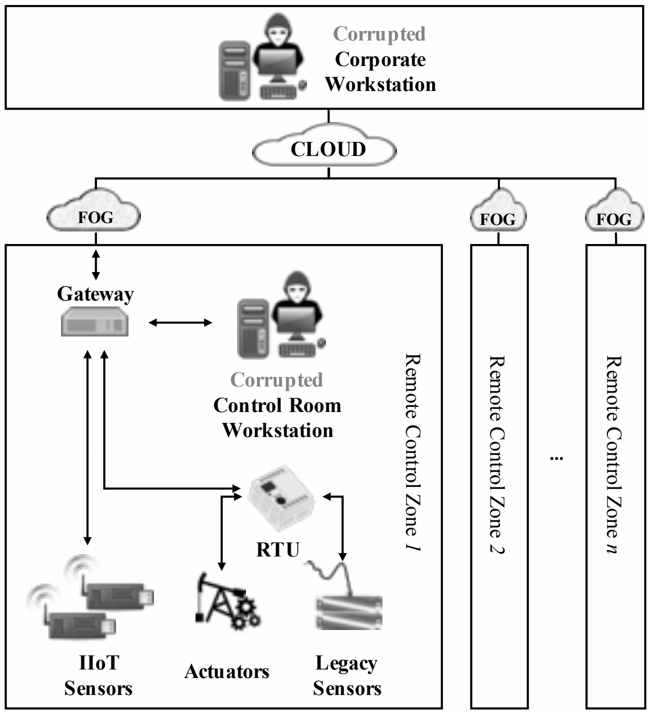


Figure 1: Attackers addressing Modbus/TCP covert channel in OT networks.

can be drawn, however, the most common way is to classify them by type. Three different types have been widely recognized [14]: storage, timing-based, and behavior-based covert channels.

- *Storage-based* covert channels hide data in a medium that is shared between the endpoints. The shared resource is not designed to carry data. In a networked system, a covert channel can be implemented by exploiting the reserved fields in various packet headers/footers or by concealing data in the payload;

- *Timing-based* covert channels modify the event-timing to share data between the endpoints. The key advantage is that the covert channel is set up without altering the communication data stream to hide information. To this end, timing-based channels can be applied to several protocols: they are independent of syntax and/or semantics of the network's packet;

- *Behavior-based* covert channels operate by intentionally changing the behavior of an application. This change is used by the endpoints to communicate. Behavior-based covert channels address the application level, so they do not depend on a specific network protocol or synchronization. As a consequence, they are more difficult to prevent and detect: the complete knowledge of the application is needed to identify the message in such a covert channels.

The design and the implementation of covert channels over communication networks represent the most prolific field. According to [15], all the protocols in the TCP/IP stack have been considered in setting up covert channels. In [12, 16], vulnerabilities in the headers of the network protocols are used to set up storage-based covert channels. The information hiding depends on the vulnerabilities detected and the channel bandwidth can be considerably low. In [17, 18], the concealed data are embedded in the payload of the packets. Specifically, TCP data are encoded in the payload of ICMP or DNS queries using tunneling techniques. Both types of communication alter the information properties of the channel they use. This approach can be easily discovered since it introduces a large amount of information inside.

In [19], the Modbus protocol is exploited to perform an attack using a storage-based covert channel. Specifically, a state machine is adopted to model the server and the client of a Modbus/TCP communication. Master sends malicious commands on the covert channel that is implemented, using holding registers and coils. The covert channel is used to run multiple transfers of compressed files from the server to the client, that corresponds to exfiltrating a compressed data archive. The approach is tested using a simulated environment and the throughput of the channel is analyzed. The tests are made using virtual machines on the same virtual network: one Windows workstation image is running the client and another is running the server. The Modbus/TCP Command and Control (denoted here as C&C) channels have been implemented using Python and specific Modbus/TCP libraries.

The same protocol is applied in [20] to set up a storage and behavior-based covert channel. Specifically, it considers the read-only request from client to server. The server interprets the number of the requested objects as an ASCII character, according to a shared alphabet, so that a data transfer can be executed. It is worth noting that the

Table 1: Differences between approaches and contribution.

	Storage based	**Timing based**	**Behavior based**	**Data Exfilt.**	**C & C**	**Simulation**
[19]	✓			✓		Virtual
[20]	✓		✓	✓		Virtual
[21]	✓	✓				Concept
Proposed	✓	✓	✓	✓	✓	Testbed

concealed data is not contained as a payload within the Modbus request itself, but it is extracted depending on how the request is interpreted, thus implementing a behavior-based covert channel. The performance of the approach is evaluated in a simulated environment: client and server are separate hosts with a software-based Modbus/TCP implementation. In this work, the authors also used Python with a software library to develop the client and server scripts.

Covert channels have been generally conceived for malicious purposes, however, in [21] the authors developed timing and storage channels to provide a methodology able to implement a secure authentication for Modbus/TCP communication. Concretely, the authors introduce a new concept of Modbus Covert Channel Integrity Check (MC-CIC) to send out-of-band messages and to verify the data integrity. But despite all these progresses, there is no complete contribution yet that meets all covert channel characteristics (timing, storage and behaviour). For this reason, this paper focuses on providing two signaling strategies based on the timing and storage properties, with the possibility of changing the behaviour of an application by simply executing malicious commands from remote locations. To do this, we assume that attackers are able to exploit both the real-time constraints and manipulate the information at application level in order to modify the normal behaviour of the field devices (actuators, controllers, servers, etc.). In other words, we present two approaches based on: (i) *timing* where insignificant "delays" are injected in the TCP/IP channels − depending on the delay time, the covert channel is created or closed −, and (ii) *storage* by the inclusion of hidden data in specific fields of the Modbus/TCP packets, such as Unit Identifier. Once the covert channel is created using one of these two strategies, attackers can not only exfiltrate information to external entities but also execute active C&C instructions in particular devices of the system (*Behaviour*). Table 1 summarizes the differences between approaches and stresses the ways how they have been validated.

3 Threat model and covert channel techniques

As mentioned above, in this section we model a specific covert channel for Modbus/TCP communication channels following the traditional client-server communication model. Legitimate servers or Master Terminal Units (MTUs) are able to connect to controllers (e.g., RTUs (Remote Terminal Units)/PLCs (Programmable Logic Controllers)) to receive back information from a determined context (e.g., pressure, humidity) or modify the state of the context through actions. Controllers serve as Modbus/TCP servers in the perspective of the client/server model, also illustrated in Figure 1.

To simplify the attack process, it is widely assumed that the adversary applies advanced attack techniques (e.g., social engineering) for the penetration and applies lateral movements, passive traffic analysis to map network topologies, and injection of malware code to manage the network traffic, navigate and control [4, 7]. These capacities normally belong to advanced and persistent attacks (based on diverse attack vectors: *Reconnaissance, Delivery, Compromise, Command and Control, Lateral Movement, Execution*) [7], whose complexity is out of the scope of this paper but for its stealthy nature may involve a covert channel. For the design of a specific covert channel in Modbus/TCP, a set of assumptions are required together with some notations:

Assumption 1. *Let A be an expert attacker, targeting a critical control environment. At this point, we assume A knows the network topology and its resources, such as location of RTU, MTU, gateways, sensors, actuators, etc., with high mobility capacity to sublimate nodes within the network. This characteristic is also represented in Figure 1.*

Assumption 2. *Based on Assumption 1, A is able to compromise a legitimate Modbus/TCP server and a rightful client so as to establish the covert channel through them.*

This last supposition, in turn, entails to assume that A is able to install in the legitimate server (e.g., RTU/PLC) and in the legitimate client (e.g., Human Interface Machine (HMI) or SCADA server) a piece of malicious code capable of interpreting any traffic transferred by a covert channel, without implying the participation of a man-in-the-middle. This malicious code is denoted in this paper as S^m and C^m, such that m refers to a malicious node, S the server and C the client node.

Figure 2 shows the general features of the threat, which may integrate collateral C&C^m operations executed from a remote node and through the covert channel. These actions are classified as follows:

- *passive command and control instructions* (C&C$_p^m$) over Modbus/TCP traffic to read evidence without causing any distortion to its real value. Generally, this action is mainly addressed by C^m, such that the communication of C&C$_p^m$ takes the following direction: $C^m \rightarrow S^m$.
- *Active command and control operations* (C&C$_a^m$) over Modbus/TCP traffic or sensitive variables of the system (e.g., discrete input register values) to deliberately modify the natural behavior of field devices. In this case, the manipulation may be led by both S^m and C^m.

3.1 Covert channel: signaling techniques and modeling

Considering the previous assumptions, we define in this section the attack model. As mentioned, A requires that two malicious pieces related to S^m and C^m are introduced in the target nodes. Once they have been installed, both S^m and C^m must be subject to a

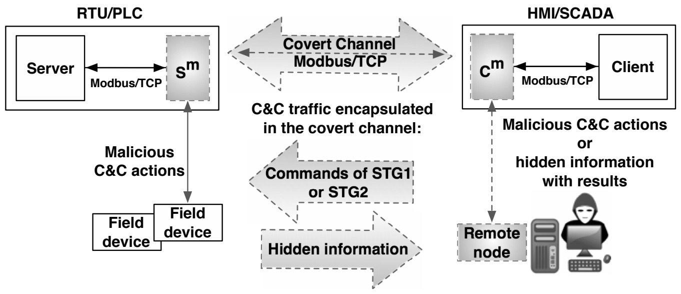


Figure 2: General attack model based on malicious C&C actions.

common actuation policy that only they must know. Taking as reference the work done in [20] and its capacity to transfer secret information (e.g., an input register data of a particular sensor) in Modbus/TCP packets (henceforth as s^m), we define two possible signaling policies from C^m to S^m:

[STG1]: C^m applies a pre-established C&C$_p^m$ based on a delay δ_t (*timing*), which represents the signaling mode to deliver a secret between S^m to C^m. To go unnoticed throughout the threat, it is essential that A predefines the delay value within a prudent time period such that $t(q_{i-1}(\cdot)) - t(q_i(\cdot)) \geq \delta_t$, where $q(\cdot)$ represents the query launched by a C^m.

[STG2]: as in **STG1**, the signaling pattern is determined by a particular code sequence in a reserved field of C&C$_p^m$ corresponding to the Modbus/TCP packet, such as Unit Identifier, Protocol or Function Code (*storage*). The use of one field or another will depend on how they will be applied within an application scenario.

Taking into account these two strategies (**STG1** and **STG2**) and the notation given in Table 2, we now formalize the communication in the side of the client, i.e., $C^m \rightarrow S^m$. In this case, C^m needs to notify the kind of secret information to be transferred to S^m, indicating the signaling mode: either **STG 1** or **STG 2**:

$$C^m(*)_{STG1} = \begin{cases} q(t,\cdot) & if\ no\ attack \\ q(t+\delta_t,\cdot) & if\ attack \\ & through\ a\ C\&C_p^m \end{cases} \tag{1}$$

where $*$ contains $q(t,\cdot)$ (for reasons of space).

$$C^m(*)_{STG2} = \begin{cases} q(field_{Modbus/TCP},\cdot) & if\ no\ attack \\ q(field_{Modbus/TCP_T},\cdot) & if\ attack \\ & through\ a\ C\&C_p^m \end{cases} \tag{2}$$

where $*$ contains $q(field_{Modbus/TCP},\cdot)$ (for reasons of space).

Once the starting notification has been received by S^m, it needs to execute the actions requested by C^m through C&C$_p^m$. If C&C$_p^m$ entails the emission of a particular type of information to C^m, S^m has to include this information in the least significant bit

Table 2: Notation of the strategies and their algorithms

Notation	Meaning
S and C	Legitimate servers and clients
S^m and C^m	Malicious piece installed in S and C
C&C$_p^m$	Passive C&C actions
C&C$_a^m$	Active C&C actions
$q(t/field_{Modbus/TCP_T}, \cdot)$	Modbus/TCP query dependent of a C&C$_p^m$
$r(\cdot)$	Modbus/TCP response packet
$r(\cdot)^m$	Modbus/TCP response packet modified by A
ϕ	Function code
ϕ_T	Target Function code
t	Time
δ_t	Malicious time delay
Rg	Register value
Rg_T	Target register value
Rg^m	Register value modified by A
field$_{Modbus/TCP}$	Modbus/TCP field
field$_{Modbus/TCP_T}$	Modbus/TCP field targeted by A
$bin(\cdot)_i$, $i \in \{1,...,n\}$, where i is the bin value index	Conversion to binary value
s^m	Secret message between S^m and C^m
γ	Initializer corresponding to r^m
μ	Terminator related to r^m
$covert_{ch}$	Covert channel

of the Modbus/TCP response packet and in such a way that:

$$S^m(r(\phi, Rg_k, \cdot)) = \begin{cases} r(\phi, Rg_k, \cdot) & if\ ((\phi \neq \phi_T)\ and \\ & (Rg_k \neq Rg_T)\ and\ *^1 \\ & or\ no\ attack \\ r^m(\phi_T, Rg^m_{k_T}, \cdot) & if\ (\phi = \phi_T)\ and \\ & (Rg_k = Rg_T)\ and\ *^2 \end{cases} \tag{3}$$

where $*^1$ contains $bin(r(\phi_T, Rg_k, \cdot))_n = bin(sm)_i)$, and $*^2$ includes $bin(r(\phi_T, Rg_{k_T}, \cdot))_n \neq bin(sm)_i$. Additionally, $r^m(\phi_T, Rg^m_k, \cdot)$ embodies the construction of the packet, which can be compounded of a list of k registers of the style $\{r(\phi_T, Rg_k, \cdot) + \alpha_i\}$, where Rg_{k_T} and ϕ_T refers to the register and Function Code targeted by A, and α_i represents the change in the value of the response. The value of α can generically be computed as follows:

$$\alpha = min(value(\cdot))\ such\ that\ bin(r(\phi_T, Rg_k, \cdot)) + \alpha = bin(s^m)_i \tag{4}$$

In this way, and regardless of the type of the value of the registers, α adds the minimum value to be considered within the type of data itself. Thus, if the least significant value of the demanded register, $bin(r(\phi_T, Rg_k, \cdot)_n$, is equal to the most significant i-th bit of the secret message s^m being transferred, S^m sends the same packet received, $r(\phi, Rg_k, \cdot)$, and increases the index value i for the reading of the next secret. Otherwise, S^m transmits back a register value increasing its value to α.

3.2 Actions in OT networks: passive and active C&C

With the operative covert channel, A may lead multiple types of threats. Here, we specify two malicious actions, which can trigger collateral effects in or from S^m:

- data exfiltration (measurements, alarms, logs, state variables or system parameters) from S^m to C^m and through C&C^m_p (Figure 3, above-hand side), and
- distortion in parameters, variables or natural states of the underlying system (Figure 3, bottom-hand side).

For the *data exfiltration*, the communication between S^m and C^m must be subject to a common secret that only they must know. The secret is predefined by two keywords related to the beginning and end of the covert channel (i.e., γ and μ, respectively), and according to a binary/ASCII sequence code associated with a determined value to mark off the real size of s^m. The resulting message consists of a concatenation of values, of the style: $(\gamma \mid\mid s^m \mid\mid \mu)$. In this way, when S^m receives the signal δ_t, it interprets the need to transfer the required information following a specific action policy, which is described in more detail below. When S^m ends the transference, it adds the μ code so that C^m knows when to stop the reading of s^m. When C^m receives the secret, it needs to interpret all the information flow related to $(\gamma \mid\mid s^m \mid\mid \mu)$. In this case, the malicious client computes for each received r^m, the binary conversion $bin(r^m(\phi_T, Rg^m_{k_T}, \cdot))$ in order to extract from each k target input register, the meaning of the least significant bit in binary. For the re-composition of the final secret, the client has to temporally store the information flow in a buffer until that r^m contains the real value related to μ.

Algorithm 3.1 and Algorithm 3.2 summarize the behavior of the threat; both assuming that C^m sends regular Modbus/TCP requests based on standardized function codes to S^m. As shown in the algorithms, the process is normal until the client adds a Modbus/TCP query and the signaling order (either of class **STG1** or **STG2**) together with a target attack sequence specifying a function code (ϕ_T) and a register value (Rg_{k_T}). Once the server S^m processes the threat, this one has to exfilter and hide the secret following the structure $(\gamma \mid\mid s^m \mid\mid \mu)$ in $r^m(\phi_T, Rg_{k_T}, \cdot)$ as defined in Algorithm 3.2.

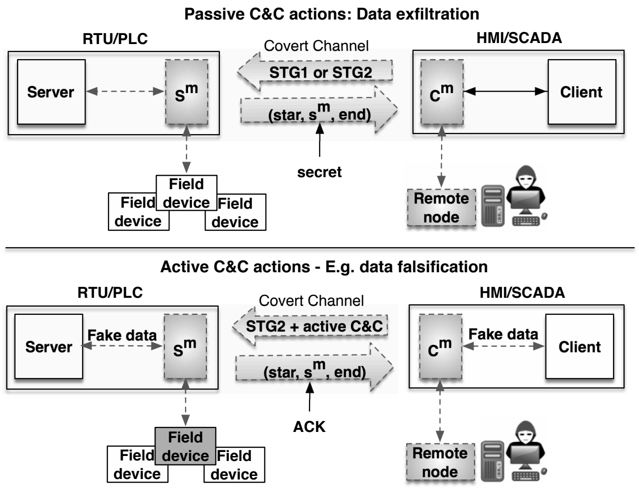


Figure 3: Passive and active actions in OT networks.

Algorithm 3.1: MALICIOUS S^m $(q(t/field_{Modbus/TCP}, \phi, Rg_k, \cdot))$

```
local $i \leftarrow 0$; $s^m \leftarrow secret; covert_{ch} \leftarrow$ false;
output $(r(\phi, Rg_k, \cdot))$;

if $((t(q_{i-1}(.)) - t(q_i(.)) \geq \delta_t)$ or
$q(field_{Modbus/TCP} = field_{Modbus/TCP_T}, \cdot))$
 and $(covert_{ch} = false)$
  then $\{covert_{ch} \leftarrow$ true;
  else if $((\phi = \phi_T)$ and $(Rg_k = Rg_{k_T})$ and $(covert_{ch} =$ true$))$
  then
$\begin{cases} r(\phi, Rg_k, \cdot), i, covert_{ch}\} \leftarrow \\ \text{DATA EXFILTRATION TO } C^m(r(\phi_T, Rg_{k_T}, \cdot), s^m, i, covert_{ch}); \end{cases}$
return $(r(\phi, Rg_k, \cdot))$
```

Algorithm 3.2: DATA EXFILTRATION TO C^m $(r(\phi_T, Rg_{k_T}, \cdot),$
$s^m, i, covert_{ch})$

```
local $s^m \leftarrow (\gamma \,||\, s^m \,||\, \mu)$;
output $(r^m(\phi_T, Rg_{k_T}, \cdot))$;

if $((i < length(bin(s^m)))$ and $(covert_{ch} =$ true$))$
  then
$\begin{cases} \textbf{if } (bin(r(\phi_T, Rg_{k_T}, \cdot))_n \neq bin(s^m)_i \\ \quad \textbf{then } \{bin(r^m(\phi_T, Rg_k, \cdot))_n \leftarrow bin(r(\phi_T, Rg_{k_T}, \cdot))_n + \alpha; \\ i \leftarrow i + 1; \end{cases}$
  else $\begin{cases} covert_{ch} \leftarrow \textbf{false}; \\ i \leftarrow 0; \end{cases}$
return $(r^m(\phi_T, Rg_{k_T}, \cdot), i, covert_{ch})$
```

Another way to address attacks in the covert channel and within a complex infrastructure is through *active actions* on the side of the server (PLC/RTU). Attackers may, for example, corrupt the normal states of a system by subtly varying their relative values, either of measurements, commands or alarms. To make the threat effective and prevent the legitimate nodes (both the client and the server $-$ C and S, respectively) from noticing the changes produced in the parameters or variables of the system, the actions taken by S^m and C^m must be addressed in a particular way. In first place, the actions taken by the legitimate devices (S and C) must be masked so as to show normality; and secondly, any deliberate C&C$_a^m$ action must be made in the covert channel created between S^m and C^m.

Figure 3 (above-hand side) illustrates an active attack under C&C$_a^m$ orders. Particularly, the example represents the manner to falsify data to legitimate entities while active actions are addressed against the correct performance of the system and its operative states. To formalize the process, Algorithm 3.3 reflects the threat making use of the covert channel. Through this channel, A is able to send C&C$_a^m$ actions to take the control in the field while injecting fake data from the sensors in S.

Algorithm 3.3: DATA FALSIFICATION TO S $(q(field_{Modbus/TCP}, \phi, Rg_k, \cdot))$

```
local $i \leftarrow 0$; $s^m \leftarrow (\gamma \,||\, ACK \,||\, \mu)$; $covert_{ch} \leftarrow$ false;
output $(r(\phi, Rg_k, \cdot);)$

procedure MODIFY DATA SENSOR$(r(\phi, Rg_k, \cdot))$
  $r^m(\phi, Rg_k^m, \cdot) \leftarrow bin(r(\phi, Rg_k, \cdot)) + \alpha$;
  return $(r^m(\phi, Rg_k^m, \cdot))$

main
  $r(\phi, Rg_k, \cdot) \leftarrow$ OBTAIN SENSOR DATA$(\phi, Rg_k, \cdot)$;
  if $(q(field_{Modbus/TCP} = field_{Modbus/TCP_T}, \cdot)$
    then $\{covert_{ch} \leftarrow$ true;
    else if $((\phi = \phi_T)$ and $(Rg_k = Rg_{k_T})$ and $(covert_{ch} =$ true$))$
    then
      $r(\phi, Rg_k, \cdot) \leftarrow$ MODIFY DATA SENSOR$(r(\phi, Rg_k, \cdot))$;
      comment: $S^m$ may concurrently send to $C^m$ a success notification
      $r(\phi, Rg_k, \cdot), i, covert\_channel\} \leftarrow$
      DATA EXFILTRATION TO $C^m(r^m(\phi_T, Rg_{k_T}, \cdot), s^m), i, covert_{ch})$;
    else
      $covert_{ch} \leftarrow$ false;
      $i \leftarrow 0$;
  return $(r(\phi, Rg^k, \cdot))$
```

4 Practical validation based on IT-OT networks

For the practical validation of **STG1** and **STG2**, we consider the implementation of the approaches in a real testbed of IIoT belonging to the NICS Lab[2]. The testbed incorporates cyber and physical components of the industrial control such as: three

[2]NICS Lab research group, https://www.nics.uma.es, University of Malaga.

Arduino-certified development boards (concretely, three Intel Galileo boards) working as field devices, one Raspberry Pi 3 board supporting the logical of the PLC, one server including the services of HMI and SCADA system, and a dedicated server capable of obtaining and simulating evidence from the physical world. For these inputs, the server constantly connects to specific sources of real physical world capable of providing data streams in real time. In our case, we connect with [22] to inject the field devices with information related to temperature and humidity, creating, in this way, an IIoT testbed capable of relating the real physical world, corresponding to the energy sector, with the virtual physical world.

Considering the assumptions made in Section 3 and the attack model defined in Section 3.1, the covert channel and the malicious C&$C^m_{p/a}$ actions have been developed in Python together with the libraries NetfilterQueue and Scapy. With NetfilterQueue, it is possible to manage the access to Modbus/TCP traffic by means of iptables rules allowing S^m to be decoupled from S and C^m from C as expressed below:

```
iptables -A INPUT  -s <IP (server/client)> -p tcp -sport 502
                   -j NFQUEUE-queue-num 1
iptables -A OUTPUT -d <IP (server/client)> -p tcp -dport 502
                  -j NFQUEUE -queue-num 2
```

Both rules state that any Modbus/TCP traffic (with reserved port `502`) must first be redirected to S^m or C^m instead of S or C. Scapy, to the contrary, is a very effective network tool that permits A to capture and modify Modbus/TCP traffic in real time, and even return the perturbed packets to the communication channel in optimal times. Regarding the experiments, they have been addressed according to the stealthy command and control actions defined in this paper, and through the activation of the covert channel following the strategy **STG1** and **STG2**. For **STG1**, δ_t has been pre-established for 2 seconds; whereas for **STG2**, we have applied the reserved Modbus/TCP field, `Unit Identifier` of one byte of size.

Unit Identifier is part of the MBAP header added at the start of the Modbus/TCP PDU containing slave address, function code, registers and the cyclic redundancy check. This field is always initialized by the client, and it is used for intra-system routing purposes where the communication is based on the interconnection of Ethernet TCP/IP networks with Modbus serial networks. In this sense, the bridge devices or gateways use the Unit Identifier to forward the request to the right slave device. This feature is, to the contrary, not applicable on TCP/IP networks since any cyber-physical device can be directly reached through their own IP addresses. This also means that the Unit Identifier field is also useless for our proof of concept, the value of which is generally predetermined to `0x00`. However, attackers may take advantage of this field to modify its value and add relevant information that allows to hide actions in the OT network, such as: the activation of the covert channel or the execution of malicious actions in field devices. In our validation tests, the indicator `0xFF` is applied to establish the start signal of the covert channel.

For the responses from S^m to C^m, we consider the least significant bit of those queries of type float, resulting in one bit for every two Modbus/TCP registers. This also means that the number of registers applied for the storage corresponds the half of the Modbus/TCP registers. In this case, the communication frequency of the covert channel depends on the number of float type values that are requested by C^m, as well as the updating frequency of these values. If we suppose, for example, that C^m requests ten values of type float and the period of update is one second, then the communication frequency would be 10 bits/second. Therefore, the complexity of our approach is found in the communication channel, either for **STG1** or for **STG2**. In both, the costs to activate the covert channel is practically linear, but the communication overhead involved in the response will depend on the size of the response and the updating frequency to later reconstruct the message. As this is a related factor to stealthy nature of the threat, any approach will present similar costs: straightforward in the computation to create the covert channel, but considerable for the communication. For example, when C^m requests to read a specific input register in [20], the covert channel needs to transfer a camouflaged ASCII character to S^m, presenting a similar overhead to our approach. Moreover, future approaches combining diverse strategies for the covert channel activation (**STG1-STG2**, combinations of delays or several keywords), will present similar behavior where the complexity will be concentrated in the communication channel.

Figure 4 shows the feasibility of the two signaling approaches. In both, it is easy to see how A is able to activate the covert channel taking advantage of the characteristics of the communication channel or the predetermined structures of the communication protocols. In relation to this figure, the picture on the left-hand side of Figure 5 illustrates the effects of any C&C$^m_{p/a}$ once the covert channel has been activated; in this case, it corresponds to the transference of a secret between $S^m \rightarrow C^m$ according to α value. The deviations between the original data and their modifications are strictly minimal, highlighting the stealthy nature of the threat. This characteristic is also noted in the picture to the right-side of Figure 5. The effect of the threat is mainly marked by slight deviations in the temperature threshold ($[25-27]$) and the humidity threshold ($[40-42]$), the impacts of which are due to α. More specifically, the simulations have been planned for an $\alpha = 0.25$ in order to keep the new variations within the predetermined temperature and humidity thresholds. In this sense, an attacker needs to know the real threshold values to find a way to go unnoticed. The subtler the value of alpha is, the stealthier the threat is; this also means that the value of α constitutes a challenge for A.

To evaluate a covert channel, it is also possible to consider the throughput and the robustness of the channel as stated in [20]. The throughput measures the amount of data that the channel is able to transmit over a given period of time. This feature is related to the capability of the channel to remain undiscovered: a high throughput, implies a higher probability of being detected by advanced IDSes. The threat robustness concerns the capability to persist when different security measures are crossed, but it depends largely on the protection level offered by the system.

5 Countermeasures

Security-by-design using secure protocols in OT networks is certainly the best practice to reduce the cyber-attack surface in an effective way. The best solution to effectively protect innovative IIoT architectures from cyber threats is represented by a complete secure reengineering of the systems, starting with a well-defined segregation of IT and OT networks. Unfortunately, legacy systems are very difficult to renew due to complex configuration of multiple vendors for the OT physical processes functioning. This coexistence problem is also more invasive in IIoT scenarios where old devices are coupled with the latest generation of systems. Consequently, security implementations for detection of cyber threats have a central role in IIoT architectures.

In scenarios where IDSes are considered priority mechanisms, the covert channel detection challenge can still be complex. The stealthy nature of the threat involves the need to trigger casual slight anomalies over long time periods to avoid raising intrusion suspicion [23]. Therefore, the persistence of the threat over time with slight deviations can be keys to determine a stealth attack. Apart from this, and considering the two threat specifications given in Section 3.1, we also note that **STG1** can be less detectable by advanced anomaly-based IDSes than **STG2** through signature/specification-based IDSes. Casual delays δ_t in the Modbus/TCP traffic with insignificant delays in the communication channels (e.g., $\delta_t \approx [0,5-1]$ second/s) can have less repercussion during the monitoring processes than an intentional exploitation of Modbus/TCP fields.

A wide range of IDSes have been proposed for cyber-physical environments [24]. However, the greater majority of them are designed for specific classes of attacks without exploring possible intelligent capacities to detect multiple attack patterns of covert channel with variable capacities to modify their attack vectors in real time [24]. In the case of data exfiltration, attackers may, for example, customize their attack vectors to optimize modus operandi and simplify their activity traces. In our case, this could be associated with the inclusion of one bit of the secret s^m in each less significant bit of the k input data registers involved in the request r^m, instead of using one register in each transmission. In this way, A can enhance the covert channel bandwidth and reduce sent frequency of secret messages via the covert channel. This feature was also considered by Gao and Morris in [25], which also underlined that industrial control traffic generally follow regular communication patterns. This regularity permits an easier integration of signature-based IDSes in industrial control systems by simply including specific rules related to format and length of packets and policies defined by the communication protocols. However, signature-based IDSes in constrained devices such as RTUs/PLCs or sensors can be more complicated. As stated in [26], their monitoring systems generally demand complex databases with pre-defined attack patterns or known undesirable states forcing the constrained resources to depend on large databases. These databases, in turn, require frequent maintenance processes to keep the databases up-to-date. Similarly, specification-based IDSes also add computational and storage costs since the attack models are specified according to the technical specification of the devices.

Anomaly-based systems, to the contrary, allow to detect unforeseen changes to be detected without relying on predefined attack patterns [27]. However, the incorporation of advanced techniques in independent devices (e.g., decision trees, time series, rule learners, clustering, support vector machines, Markov models, artificial neural networks, and so on) could also have a significant effect on the resources of the system [26]. Generally, their methods demand computational and storage capacities to invest in processing, training and assessment, without a guarantee of accuracy and a suitable tuning of their own models. This feature is also contemplated by Jokar in [28] who clearly states that anomaly-based systems generally present a high false positive rate with respect to signature/specification-based IDSes. This also means that the probability of detecting a covert channel modeled through **STG2** and using a signature/specification-based IDS can be greater than applying anomaly-based detection for covert channels created through **STG1**. On the other hand, and thanks to the routing capacities of the new IoT paradigms, the use of redundant wireless communication channels should be applied to not only favor deployment costs but also resilience. Human operators are also called to apply countermeasures based on their own experience or with the support of decision support systems suggesting possible countermeasures according to the situation [29]. Likewise, any new IT incorporation (e.g., fog/cloud servers [11], digital twins) implies to review security levels, security policies and regulatory frameworks in order to verify the well usage and performance of components, in addition to applying accountability and audit measures.

6 Conclusions and future work

Since ICSs opened their systems to adapt the new ICTs, multiple and potential attacks have not stopped appearing [7, 8, 9]. A simple way to address these of attacks is by means of covert channels, through which attackers might exfiltrate data and/or remotely execute commands in order to falsify or modify critical states or variables. To do this, two signaling strategies (**STG1** and **STG2**) have been presented so as to create Modbus/TCP-based covert channels and lead multiple types of subsequent attack vectors. As future work, we plan to show the feasibility of the strategies for other industrial protocols, such as Ethernet/IP, S7 and OPC-UA, and provide defense measures based on machine learning and opinion dynamics to prevent these types of threats. We also foresee that the migration can become straightforward as the signaling **STG1** heavily depends on the communication delay, δ_t; whereas with **STG2**, it is only required to consider the implicit characteristics of the protocols and their packets (header, payload, trailer) to hide information.

Acknowledgments

Part of this work was done while the first author was visiting UCBM in 2017 under the Visiting Programme funded by this University. Additionally, the work was partially supported by the projects: SealedGRID (8.06.UE/47.8035) financed by the EC, SADCIP (RTC-2016-4847-8) financed by the Spanish Ministry of Economy, Industry and Competitiveness, HANSEL (LR132008 FILASRU20141193) financed by Lazio Region, and DISS-IIoT financed by the University of Malaga (I Plan Propio de Investigación y Transferencia). The work was also supported by a grant of the Italian Presidency of the Council of Ministers. The authors also thank J. Rodriguez (NICS Lab.) and Matteo Langone (Coserity Lab.) for their valuable support and incredible help.

References

[1] C. Alcaraz, G. Fernandez, and F. Carvajal. Security aspects of scada and dcs environments. In *Critical Infrastructure Protection: Information Infrastructure*

Models, Analysis, and Defense, volume 7130 of *LNCS*, pages 120–149. Springer, September 2012 2012.

[2] B. Chen, J. Wan, L. Shu, P. Li, M. Mukherjee, and B. Yin. Smart factory of Industry 4.0: Key technologies, application case, and challenges. *IEEE Access*, 6:6505–6519, 2018.

[3] Y. Lu. Industry 4.0: A survey on technologies, applications and open research issues. *Journal of Industrial Information Integration*, 6:1 – 10, 2017.

[4] J. E. Rubio, C. Alcaraz, R. Roman, and J. Lopez. Analysis of intrusion detection systems in industrial ecosystems. In *14th International Conference on Security and Cryptography (SECRYPT)*, pages 116–128. SciTePress, 2017.

[5] K. Stouffer, T. Zimmerman, C. Tang, J. Lubell, J. Cichonski, and J. Mccarthy. NISTIR 8183, Cybersecurity Framework Manufacturing Profile. 2018.

[6] NIST. Framework for Improving Critical Infrastructure Cybersecurity, Version 1.1. 2017.

[7] J. E. Rubio, R. Roman, C. Alcaraz, and Y. Zhang. Tracking advanced persistent threats in critical infrastructures through opinion dynamics. In *23rd European Symposium on Research in Computer Security (ESORICS)*, pages 555–574. Springer, 2018.

[8] C. Alcaraz and J. Lopez. Wide-area situational awareness for critical infrastructure protection. *Computer*, 46(4):30–37, April 2013.

[9] L. Cazorla, C. Alcaraz, and J. Lopez. Cyber stealth attacks in critical information infrastructures. *IEEE Systems Journal*, pages 1–15, 2016.

[10] V. D. Gligor. A guide to understanding covert channel analysis of trusted systems. Technical Report NCSC-TG-030, Homeland Security, Department of Defense, 11 1993.

[11] J. Betz, D. Westhoff, and G. Müller. Survey on covert channels in virtual machines and cloud computing. *Emerging Telecommunications Technologies*, 28(6), 2017.

[12] S. Zander, G. Armitage, and P. Branch. A survey of covert channels and countermeasures in computer network protocols. *IEEE Communications Surveys Tutorials*, 9(3):44–57, Third 2007.

[13] B. Carrara and C. Adams. Out-of-band covert channels&mdash;a survey. *ACM Comput. Surv.*, 49(2):23:1–23:36, 2016.

[14] D. Johnson, P. Lutz, and B. Yuan. Behavior-based covert channel in cyberspace. In *Intelligent Decision Making Systems*. World Scientific, oct 2009.

[15] A. Ameri and D. Johnson. Covert channel over network time protocol. In *International Conference on Cryptography, Security and Privacy (ICCSP)*, pages 62–65, New York, NY, USA, 2017. ACM.

[16] R. W. Smith and G. S. Knight. Predictable design of network-based covert communication systems. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 311–321, May 2008.

[17] D. Stodle. Ping tunnel. `http://www.cs.uit.no/~daniels/PingTunnel/index.html`, last retrieved in June 2018, September 2011.

[18] D. Kaminsky. Black ops of DNS. `www.blackhat.com/presentations/bh-usa-04/bh-us-04-kaminsky/bh-us-04-kaminsky.ppt`, last retrieved in June 2018, July 2008.

[19] A. Lemay, J. M. Fernandez, and S. Knight. A modbus command and control channel. In *Annual IEEE Systems Conference (SysCon)*, pages 1–6, 2016.

[20] C. Leonardo and D. Johnson. MODBUS covert channel. In *International Conference on Security and Management (SAM)*, 2014.

[21] J. M Taylor and H. R Sharif. Enhancing integrity of modbus TCP through covert channels. In *11th International Conference on Signal Processing and Communication Systems*, pages 1–6. IEEE, 2017.

[22] Openweathermap. `https://openweathermap.org/`, last retrieved in June 2018, June 2018.

[23] J. E. Rubio, C. Alcaraz, and J. Lopez. Preventing advanced persistent threats in complex control networks. In *22nd European Symposium on Research in Computer Security (ESORICS)*, volume 10493, pages 402–418, 2017.

[24] L. Cazorla, C. Alcaraz, and J. Lopez. A three-stage analysis of IDS for critical infrastructures. *Computers & Security*, 55:235–250, 2015.

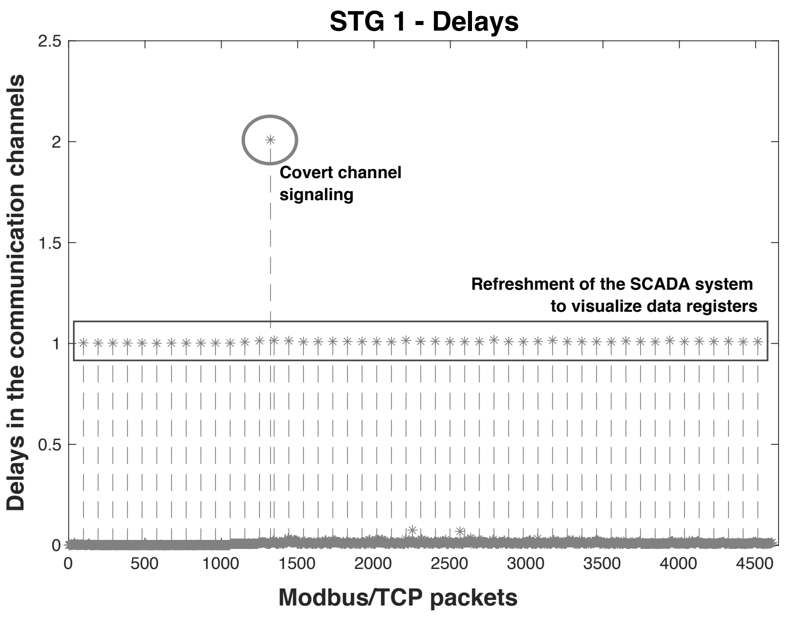
[25] W. Gao and T. H Morris. On cyber attacks and signature based intrusion detection for modbus based industrial control systems. *The Journal of Digital Forensics, Security and Law: JDFSL*, 9(1):37, 2014.

[26] C. Alcaraz, L. Cazorla, and G. Fernandez. Context-awareness using anomaly-based detectors for smart grid domains. In *9th International Conference on Risks and Security of Internet and Systems*, volume 8924, pages 17–34, Trento, 2015. Springer.

[27] E. Etchevés-Miciolino, R. Setola, G. Bernieri, S. Panzieri, F. Pascucci, and M. M Polycarpou. Fault diagnosis and network anomaly detection in water infrastructures. *IEEE Design & Test*, 34(4):44–51, 2017.

[28] P. Jokar. Model-based intrusion detection for home area networks in smart grids. *University of Bristol, Bristol*, pages 1–19, 2012.

[29] G. Bernieri, S. Damiani, F. Del Moro, L. Faramondi, F. Pascucci, and F. Tambone. A multiple-criteria decision making method as support for critical infrastructure protection and intrusion detection system. In *42nd Annual Conference Industrial Electronics Society (IECON)*, pages 4871–4876. IEEE, 2016.



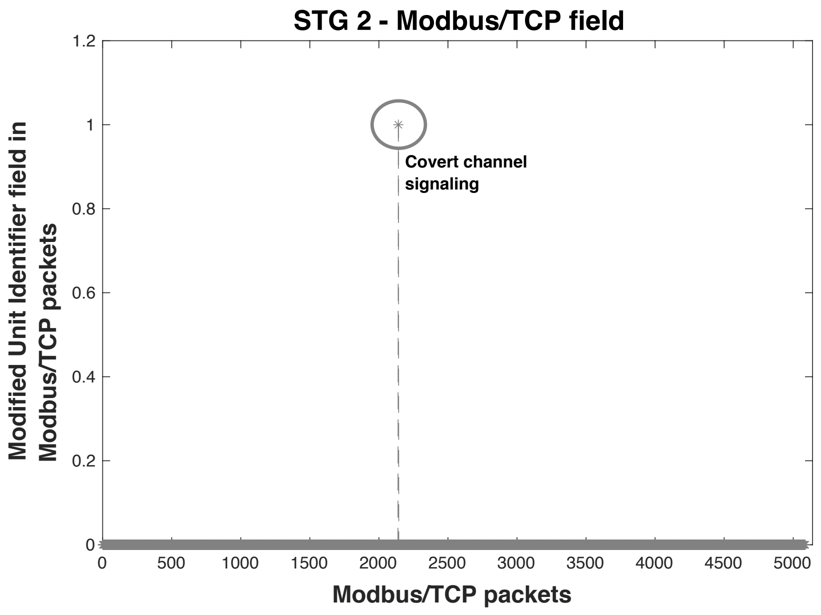
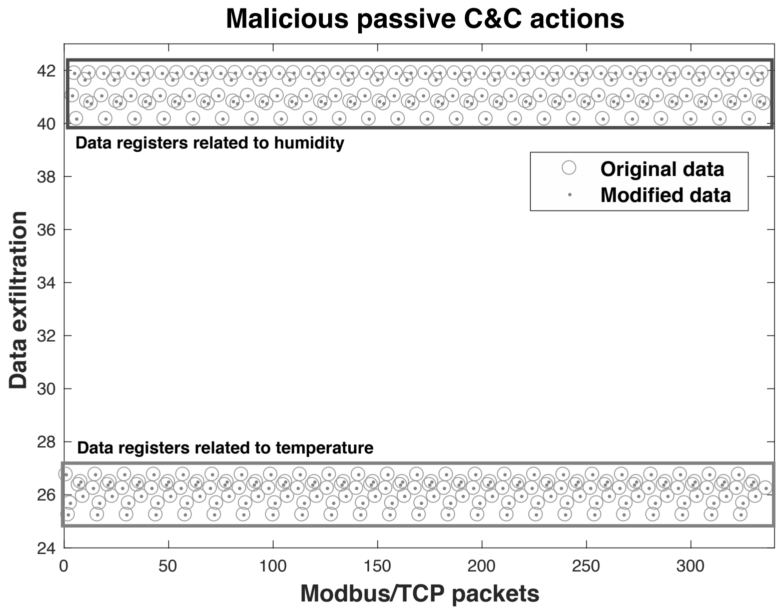


Figure 4: Covert channel using **STG-1** and **STG-2**.



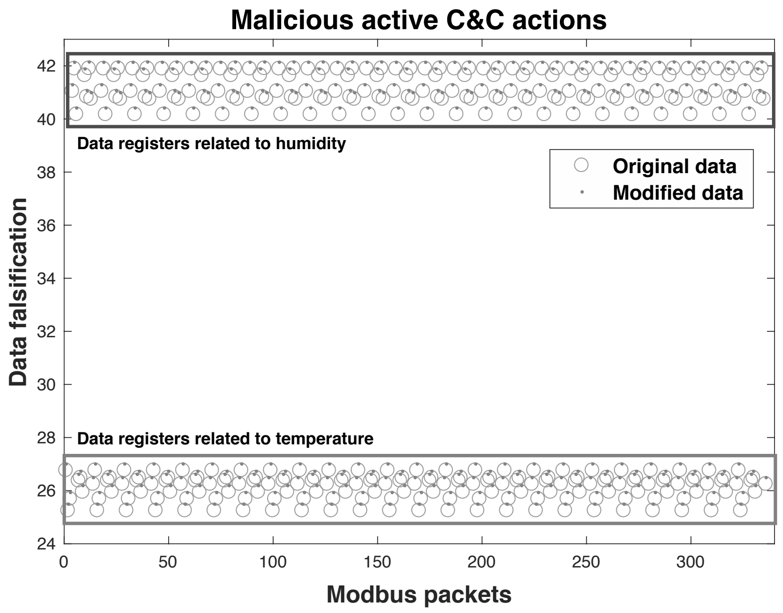


Figure 5: Data exfiltration in the communication between $C^m \rightarrow S^m$ (figure above) and data falsification to S from S^m (figure below).