Information Hiding in Cyber-Physical Systems: Selected Covert Channels and Threats at the Example of Industrial Control Systems

DISSERTATION

zur Erlangung des akademischen Grades

Doktoringenieur (Dr.-Ing.)

angenommen durch die Fakultät für Informatik
der Otto-von-Guericke-Universität Magdeburg

von M.Sc. Kevin Lamshöft

geb. am 11.11.1991 in Grantham (UK)

Gutachterinnen/Gutachter

Prof. Dr. Jana Dittmann
Prof. Dr. Stefan Katzenbeisser
Prof. Dr. Martin Steinebach

Magdeburg, den 05.03.2025

Otto-von-Guericke University Magdeburg

Faculty of Computer Science

Dissertation

Information Hiding in Cyber-Physical Systems: Selected Covert Channels and Threats at the Example of Industrial Control Systems

Author:

Kevin Lamshöft

2025

Advisor:

Prof. Jana Dittmann

Department of Computer Science

Lamshöft, Kevin:
Information Hiding in Cyber-Physical Systems:
Selected Covert Channels and Threats at the Example of Industrial Control Systems
Dissertation, Otto-von-Guericke University Magdeburg, 2025.

Abstract

In the last couple of years, two concerning trends in cyber security are observable: On the one hand, targeted attacks on Operational Technology (OT), such as Industrial Control Systems (ICS) seem on the rise, for example in critical infrastructures like power generation or water supply. On the other hand, current malware generations seem to increasingly incorporate stealth mechanisms using techniques of Information Hiding (IH), e.g. Steganography and Covert Channels. The question arises what impact these two trends would employ when combined - how can Information Hiding-based principles and techniques be applied to the specifics of such Cyber-Physical Systems? What are plausible carriers and how would exemplary covert channels perform in terms of stealthiness and capacity or bandwidth? What are corresponding threat scenarios and what challenges can be encountered for detection and mitigation of this threat? This thesis aims to shrink this research gap and answer these open questions in a selected and exemplary manner. To answer these questions, at first, a plausible reference architecture for Industrial Control Systems is proposed, which allows for a systematic analysis of potential covert communicators, plausible carriers, covert channels, and the discussion of resulting threat scenarios and potential mitigations. Potential covert sender, -receiver, and corresponding carriers are discussed in the context of Industrial Control Systems. Based on the State-of-the-Art, own investigations and publications a classification for Information Hiding methods is proposed, allowing for a systematic description of the variety of methods applicable to the specifics of Cyber-Physical Systems (CPS). In the course of three selected, exemplary case studies, a detailed investigation of three distinctive types of carrier (Automation Protocols, Process Data, and Infrastructure Protocols) is performed with a detailed description and analysis of plausible covert channels, their performance, and resulting threat scenarios and mitigation. Based on these case studies, lessons learned are discussed covering insights on plausibility, challenging features of CPS, techniques for cover object selection, bandwidth modulation, and the persistence and location of cover objects. Corresponding threat scenarios are discussed and the adversarial use of IH-methods is classified into covert lateral and covert vertical communication at the example of defense-in-depth architectures of ICS. An extended mitigation and countermeasure model is proposed and potential design principles for covert channel-robust architectures are described. Methods of detection and mitigation are reflected, based on the insights from the investigated covert channels in case studies.

This thesis is partially based on results of the research projects *"STEALTH Scenarios" (final public report [LAD23]) and "SMARTEST2" (final public report [ALD23]), funded by the German Federal Ministry for the Environment, Nature Conservation, Nuclear Safety and Consumer Protection, BMUV, Grant No. 1501589A (STEALTH) and 1501600B (SMARTEST2).*

Kurzfassung

In den letzten Jahren sind zwei besorgniserregende Trends in der Cybersicherheit zu beobachten: Zum einen scheinen gezielte Angriffe auf Operational Technology (OT), wie Industrial Control Systems (ICS), zuzunehmen, beispielsweise in kritischen Infrastrukturen wie der Energieerzeugung oder Wasserversorgung. Andererseits scheinen aktuelle Malware-Generationen zunehmend Stealth-Mechanismen mit Techniken des Information Hiding (IH), z.B. Steganographie und Covert Channels, zu verwenden. Es stellt sich die Frage, welche Auswirkungen diese beiden Trends haben, wenn sie kombiniert werden - wie können auf Information Hiding basierende Prinzipien und Techniken auf die Besonderheiten solcher Cyber-Physical Systems (CPS) angewendet werden? Was sind plausible Trägermedien und wie würden beispielhafte sich verdeckte Kanäle in Bezug auf Verdecktheit, Kapazität und Bandbreite verhalten? Um diese Fragen zu beantworten, wird zunächst eine plausible Referenzarchitektur für industrielle Steuerungssysteme vorgeschlagen, die eine systematische Analyse von potentiellen verdeckten Kommunikatoren, plausiblen Trägermedien und verdeckten Kanälen sowie die Diskussion von daraus resultierenden Bedrohungsszenarien und möglichen Gegenmaßnahmen ermöglicht. Basierend auf dem Stand der Technik, eigenen Untersuchungen und Veröffentlichungen wird eine Klassifizierung für Information Hiding Methoden vorgeschlagen, die eine systematische Beschreibung der Methodenvielfalt für die Spezifika von Cyber-Physical Systems ermöglicht. Anhand dreier ausgewählter, exemplarischer Fallstudien erfolgt eine detaillierte Untersuchung von drei unterschiedlichen Trägertypen (Automatisierungsprotokolle, Prozessdaten und Infrastrukturprotokolle) mit einer ausführlichen Beschreibung und Analyse plausibler verdeckter Kanäle, ihrer Leistungsfähigkeit und daraus resultierender Bedrohungsszenarien und -abwehr. Auf der Grundlage dieser Fallstudien werden Erkenntnisse über Plausibilität, besondere Merkmale von CPS, Techniken zur Auswahl von Cover Objeketen, Bandbreitenmodulierung sowie die Persistenz und Lokalisierung von Cover Objekten diskutiert. Entsprechende Bedrohungsszenarien werden diskutiert und die Nutzung von IH-Methoden in verdeckte laterale und verdeckte vertikale Kommunikation am Beispiel von Defense-in-Depth-Architekturen von ICS klassifiziert. Es wird ein erweitertes Gegenmaßnahmen-Modell vorgestellt und mögliche Designprinzipien für Covert-Channel robuste Architekturen beschrieben. Methoden zur Entdeckung und Abwehr werden auf der Grundlage der Erkenntnisse aus den untersuchten verdeckten Kanälen in den Fallstudien reflektiert.

Diese Arbeit basiert zum Teil auf Ergebnissen der Forschungsprojekte ”STEALTH Scenarios” (Öffentlicher Endbericht [LAD23]) und ”SMARTEST2” (Öffentlicher Endbericht [ALD23]), die vom Bundesministerium für Umwelt, Naturschutz, nukleare Sicherheit und Verbraucherschutz unter den Förderkennzeichen 1501589A (STEALTH) und 1501600B (SMARTEST2) gefördert wurden.

Contents

List of Figures

List of Tables

List of Code Listings

List of Acronyms

AD	Active Directory
ADU	Application Data Unit
AEAD	Authenticated Encryption with Associated Data
AES	Advanced Encryption Standard
APT	Advanced Persistent Threat
BAS	Building Automation Systems
C2	Command-and-Control (Channel)
Cap	Capacity
CC	Covert Channel
CM	Cover Model
CPS	Cyber Phsical System(s)
CR	Covert Receiver (Bob)
CRM	Customer Relationship Management
CS	Covert Sender (Alice)
DHCP	Dynamic Host Configuration Protocol
DHS	Department of Homeland Seucrity
DMZ	Demilitarized Zone
DNS	Domain Name System
DS	Dataset
ECB	Electronic Code Book
er	Embedding Ratio
ERP	Enterprise Resource Planning
ES	Embedding Strategy
EWS	Engineering Workstation
FC	Function Code (Modbus/TCP)
FN	False-Negate
FP	False-Positive
FPR	False-Positive Rate
FW	Firewall
GNSS	Global Navigation Satellite System

HMI	Human-Machine-Interface
HTTP	Hyper Text Transfer Protocol
I/O	Input/Output
ICS	Industrial Control System(s)
ID	Identifier
IDS	Intrusion Detection System
IH	Information Hiding
IoT	Internet of Things
IP	Internet Protocol
IPS	Intrusion Prevention System
IT	Information Technology
LLDP	Link Layer Discovery Protocol
LMT	Logistic Model Tree
LSB	Least Significant Bit
MAC	Message Authentication Code
MBAP	Modbus Application Protocol
MES	Manufacturing Execution System
ML	Machine Learning
MLP	Multilayer Perceptron
MQTT	Message Queuing Telemetry Transport
NGFW	Next Generation Firewall
NTP	Network Time Protocol
NTS	Network Time Security
NTS-EF	Network Time Security Extension Fields
NTS-KE	Network Time Security Key Establishment
OFB	Output Feedback
OOB	Out-of-band
OPC	Open Platform Communications
OPC UA	Open Platform Communications Unified Architecture
OR	Overt Receiver (Orwell)
OS	Overt Sender (Oscar)
OSI	Open Systems Interconnection
OT	Operational Technology
PDU	Protocol Data Unit
PERA	Purdue Enterprise Reference Architecture
PLC	Programmable Logic Controller
PRNG	Pseudo-Random Number Generator
PTP	Precision Time Protocol

RPM	Rounds per minute
RTU	Remote Terminal Unit
SDR	Software-defined Radio
SIEM	Security Information and Event Management
SSL	Secure Sockets Layer
SVM	Support Vector Machine
SWAT	Secure Water Treatment Testbed
TCP	Transmission Control Protocol
TID	Tansaction Identifier (Modbus/TCP)
TLS	Transport Layer Security
TN	True-Negative
TP	True-Positive
TPR	True-Positive Rate
TTP	Tactics, Techniques and Procedures
UAV	Unmanned Aerial Vehicle
UDP	User Datagram Protocol
VPN	Virtual Private Network

1. Introduction, Motivation & Scope

"If you know the enemy and know yourself, you need not fear the result of a hundred battles." - Sun Tzu, The Art of War

While in the last couple of years the trend of ransomware-based attacks is continuously making headlines in the newspapers, two concerning trends in Cyber Security have gone partly unnoticed in public discussions: Since Stuxnet in 2011 [FMC11] first wreaked havoc on the assumptions of the security and safety of Cyber Phsical System(s) (CPS) or in these cases Industrial Control System(s) (ICS), targeted attacks on facilities seem on the rise: In 2013 the Havex (or Dragronfly) malware was distributed by a supply chain compromise and was targeted at corporations in the defense and aviation domain to exfiltrate crucial information on the used Operational Technology (OT) [Nel16]. In 2015 Blackenergy and 2016 Industroyer (or Crashoverride) were used in the context of the attacks on the Ukrainian power grid resulting in a large-scale power loss [RTM16]. In 2017 the Triton malware was used in a targeted attack on an oil refinery in Saudi Arabia resulting in an emergency shutdown [GBMF20] and reportedly hit another target in 2019 as well [Gil19]. In 2022 again Industroyer was used, yet in a more advanced fashion, and therefore named Industroyer2 [ESE22]. Moreover, in 2022, Dragos [Dra22] as well as Mandiant [NKK+22] reported finding a novel malware before it could be put to use. This malware has to this point unprecedented capabilities to infiltrate and manipulate Industrial Control Systems and was named PIPEDREAM by Dragos [Dra22] and INCONTROLLER by Mandiant [NKK+22].

What makes these incidents of CPS so concerning is their inherent impact on the physical world and resulting compromise of physical safety and their potential harm to people and the environment.

At the same time in common Information Technology (IT) a different trend is observable as new malware generations seem to increasingly incorporate stealth mechanisms [CCM+18, CM22].

Such called stegomalware or stegoware [CM22] makes use of steganographic techniques e.g., loading of further malware components hidden in images [Fal20a] or

using network covert channels for data exfiltration and Command and Control (C2) [Fal20b]. A trend that is also observable in public research with increasing publications especially in the domains of *Network Covert Channels* (e.g., [LY21, WCM+21]) and *Side Channels* (e.g., [Gur20, GB19, Gur18a, GE18]).

From observing these two trends, on the one hand, targeted attacks on Cyber-Physical Systems and on the other hand the increased use of Information Hiding-based techniques, the question arises, what impact such Information Hiding-based techniques have when used by adversaries to compromise Cyber-Physical Systems or more specifically Industrial Control Systems. This question is aimed to be answered.

1.1 Scope of this Thesis

As this question of Information Hiding-based attacks in CPS is too much of a complex question to be answered within the constraints of a thesis, the scope is narrowed down in the following.

1.1.1 Focus on Industrial Control Systems

The term *Cyber Physical System* is generally a broad term, defined on many occasions [KM15] and overlaps with other terms [DS21]. As a result, systems of many domains can be considered to be *cyber physical* e.g., smart buildings, automotive, and manufacturing systems. However, a thorough analysis of the threat of Information Hiding-based attacks for all of these domains would go far beyond the scope of this thesis. Therefore, the focus is set to a specific sub-form of CPS, namely *Industrial Control Systems (ICS)*, aiming to achieve no loss of generality. While ICS is also a general term encompassing other terms, such as *SCADA* and *DCS*, this thesis follows the definition of the National Institute of Standards and Technology (NIST, SP 800-82), who describe ICS as "*[...] combinations of control components (e.g., electrical, mechanical, hydraulic, pneumatic) that act together to achieve an* ***industrial objective*** *(e.g., manufacturing, transportation of matter or energy)*" and further a typical ICS "*contains numerous* ***control loops****, human interfaces, and remote diagnostics and maintenance tools built using an* ***array of network protocols on layered network architectures****. A control loop utilizes* ***sensors, actuators, and controllers*** *(e.g., PLCs) to* ***manipulate*** *some controlled* ***process***" [SPL+15].

The scope of this thesis is primarily set to ICS for four reasons:

Prevalence Industrial Control Systems are found in nearly every industry e.g., in manufacturing and critical infrastructure. Due to that, ICS play a major role in CPS security research, guidelines and attack models are specifically designed for ICS [ABS21].

System Complexity As described in the definition ICS are complex communication networks of sensors, actuators, and controllers using layered network architectures. Nowadays, modern ICS form complex, layered systems of OT (Operational Technology) and IT (Information Technology) [SPL+15, Wil92] increasing their attack surface.

Representativeness In their complexity and overall prevalence ICS are representative for a broad range of CPS. Therefore, it can be expected that research results regarding ICS are also at least in parts applicable to other domains of CPS or can be adapted with minor adjustments.

Criticality and Impact Industrial Control Systems represent the backbone of the economy, especially in critical infrastructure. Therefore, security incidents can have a high impact on security and safety of human lives.

While these brief definitions are given here to set the scope of this thesis, Chapter 2 provides more detailed definitions and distinctions for CPS and ICS.

1.1.2 Focus on Covert Channels and CPS Steganography

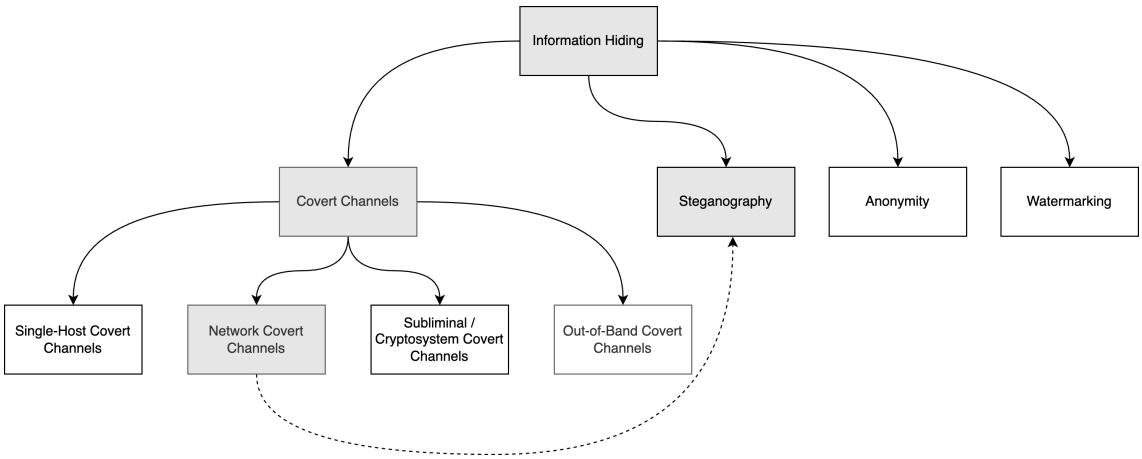


Figure 1.1: Classification of Information Hiding techniques, based on the initial classification by Petitcolas et al. [PAK99], with extensions from Wendzel et al. [WZFH15] (relationship between Network Covert Channels and Steganography), and from Carrara and Adams [CA16] (addition of Out-of-Band Covert Channels). Grey boxes indicate the focus of this thesis.

Terminology in the field of Information Hiding has blurred over time and terms are used interchangeably (see Section 2.1). Common ground for a description of the relationship of Information Hiding, Steganography, and Covert Channels is the classification of Information Hiding methods by Petitcolas et al. [PAK99], as illustrated in Figure 1.1. Wendzel et al. extended this classification to describe the relationship between Steganography and Network Covert Channels [WZFH15]. The classification illustrated in Figure 1.1 also includes the addition of *Out-of-Band* covert channels by Carrara and Adams [CA16], a sub-type of covert channels that use shared mediums without any overt traffic, similar to *side channels*. Side channels, however, do *not* belong to Information Hiding, as side channels are characterized (in contrast to Information Hiding-based techniques) by *unintentional* information leakage. Therefore, for this thesis side channels are out of scope. What seems missing is a classification of covert channels specific to Cyber-Physical Systems. For example, Alcaraz et al. [ABP+19] brought *Behaviour-based Covert Channels* over to Cyber-Physical Systems, a term originally coined by Johnson et al. in 2009

[JLY10]. Alacaraz et al. propose *Behaviour-based Covert Channels* as third category of Covert Channels in CPS, alongside Covert *Timing-* and *Storage- Channels* known from Network Steganography [WZFH15]. *Covert Timing Channels* embed information by modulating temporal aspects of the cover (e.g., time span between two consecutive packets). *Covert Storage Channels* embed information in the contents of a cover (e.g., writing information in unused fields of network packets). In contrast to such timing- and storage- covert channels, in which hidden communication is embedded in network traffic or communication in general, *behavior-based Covert Channels* change the behavior of a system on an application level [ABP+19], i.e. such are *protocol independent*. However, the common understanding seems to be, that using domain-specific steganographic methods, covert channels are established [MC15]. As pointed out in [WCM+22], domains of Steganography develop over time and do overlap. For example, in Cyber-Physical Systems covert channels might be established using Network Steganography, Text Steganography, or CPS Steganography. Therefore, as illustrated in Figure 1.2, a Covert Channel in Cyber-Physical Systems can be a *product of several domains of Steganography*.

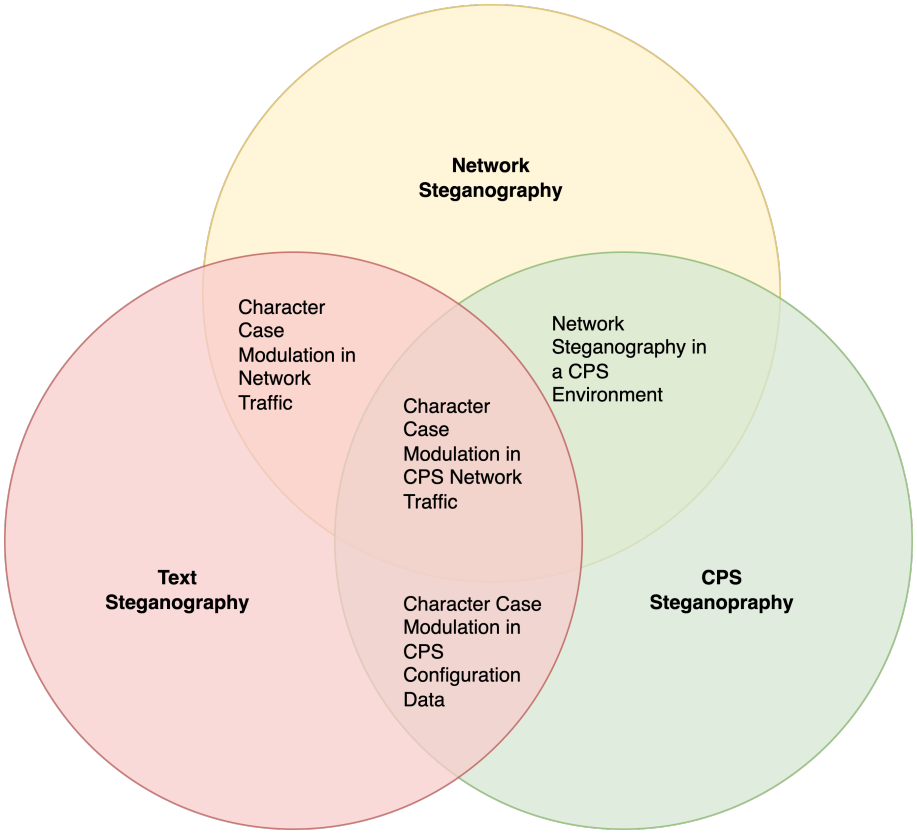


Figure 1.2: Example for overlapping domains of Steganography in the context of Cyber-Physical Systems. Example and figure based on [WCM+22].

Given this overlap of Steganography domains and the understanding that covert channels (i.e., hidden communication channels) are established using steganographic methods, the **scope** of this thesis is set to **Covert Channels** established by methods of **Cyber-Physical System Steganography** (which has overlaps with other domains of Steganography) with a focus on Communication/Network-based covert channels as this tends to be the main way of *covert* adversarial/malware-based communi-

cation [CCM+18, CM22, Cav21]. In the literature, Steganography itself is further differentiated into three types [Fri09]:

Steganography by *Cover Selection* A cover is selected from a large set of available covers to communicate a hidden message.

Steganography by *Cover Synthesis* The secret sender creates a cover that communicates the desired message.

Steganography by *Cover Modification* An existing cover is modified to encode a hidden message.

More recently, it appears especially the field of *Cover Synthesis* seems to attract a focus in research with the rise in use of *Generative Artificial Intelligence* e.g., see [dWSK+23] for reference.

However, the focus for this thesis is set to Steganography by *Cover Modification*, especially to investigate potential plausible cover/carrier[1] in the context of CPS systems.

1.1.3 Focus on Mitigation

The scope of the thesis is in most parts set by results of the research projects *"STEALTH Scenarios" and "SMARTEST2", funded by the German Federal Ministry for the Environment, Nature Conservation, Nuclear Safety and Consumer Protection, BMUV, Grant No. 1501589A (STEALTH) and 1501600B (SMARTEST2).* Therefore, the focus of the thesis is set to the evaluation of covert channels in ICS and resulting threat scenarios and models incorporating covert channel-based attack vectors, while methods and techniques for detection are investigated by project partners. However, as in the process of developing plausible threat scenarios detection plays a major role at certain points, results on detection performance are discussed in the context of the proposed covert channels.

Detection, or *Steganalysis* in general, plays a major role in Information Hiding research. In real-world deployments, detection or Steganalysis can be too much of an overhead as it requires certain knowledge and efforts and certain resources to be allocated. Whether distinctive detection methods are employed or not is part of a risk assessment by the operator. Still, in practice, many covert channels might be mitigated without prior detection if correct measures are taken during design, implementation, or configuration. For targeted countermeasures, detection remains the first step for further actions and is of importance for detection of cyber attacks. As research on covert channel detection is a rapidly emerging and changing field, methods develop rapidly and also outdate quickly as well. Therefore, in this thesis, the aim is to derive general principles and requirements for detection of such covert threat scenarios as well as to derive design principles making CPS-architecture more robust against such Information Hiding-based threats. For the case studies of this thesis, which discuss certain covert channels, detection methods will be discussed, and if data on detection performance is available by project partners these will be indicated as well.

[1] While in other work the terms *cover* and *carrier* might be further distinguished, here both terms are intentionally used *synonymously*.

Summary of Scope

In summary, the scope of this thesis is set to (Network-/ Communications-) Covert Channels in Industrial Control Systems, their corresponding threat scenarios and use cases as well as general measures for detection and mitigation.

1.2 Brief Summary of Related Work and Research Gap

Currently, public reports of attacks on facilities targeting Industrial Control Systems are still rare, undisclosed or lack important details which makes it difficult to get a better understanding of the actual threats to such systems [ABS21]. Taking into account the number of security advisories issued per year[2] by the United States Cybersecurity and Infrastructure Security Agency (CISA) the situation seems threatening, yet lacking the required data and visibility - as pointed out by Don C. Weber [Web20] in regards to the frequency of attacks on OT environments the United States FERC order 7063 mandates that "due to the lack of sufficient data available" organizations should assume that "an event will occur", and further that "Risk-based assessment methodology should focus on the consequences of an outage, not the likelihood of an outage". These statements mark the need for proper understanding and mitigation of threats to OT environments.

Many efforts were made over the last years to fill this gap and provide better understanding of Tactics, Techniques and Proceduress (TTPs) employed by adversaries. In the following, a selection of key contributions is briefly described. In 2015 the SANS Institute introduced an extended high-level model of the common Cyber Killchain by Lockheed Martin [HCA11] - modeling the different steps and phases an adversary takes when targeting Industrial Control Systems [AL15]. What stands out in this model is the insight that attacks on the Operational Technology (in most cases) require a prior full-scale attack on the Information Technology (IT) or conventional business environment respectively to acquire needed information and pivot points for breaching and compromising the associated OT environment and achieve the intended (potentially physical) impact. This is also reflected in a published case study [RMT16] by the SANS Institute. This marks the complexity of such attacks and the need for adversaries to stay undetected as long as possible and indicates the potential of Information Hiding-based techniques to increase the gap between compromise and detection.

Another major contribution to the understanding of the threat landscape of Industrial Control Systems is the ongoing work on the MITRE ATT&CK® *framework for Industrial Control Systems* (also called ATT&CK®ICS) [ABS21], a curated knowledge base and mid-level model for adversary behavior inspired by the MITRE ATT&CK® Framework for Enterprise [SAM+18] but specifically designed for the Industrial Control System domain and based on publicly available reports and security bulletins. The attack framework is under active development with regular updates. The aim of ATT&CK® for ICS is to reflect the behavior of adversaries in ICS and provide information on tactics and technical procedures, used malware, common assets and known adversary groups and Advanced-Persistent-Threats (APT). As the

[2] 204 advisories in 2019, 225 in 2020, 406 in 2021, 380 in 2022, over 400 in 2023

SANS ICS Killchain differentiates between the IT and OT domains so does MITRE, differentiating between MITRE ATT&CK® for Enterprise and Industrial Control Systems [ABS21], highlighting the difference between these two domains and their respective related adversarial behavior.

In MITRE ATT&CK® Framework *for Enterprise* [SAM+18], methods of Information Hiding and Steganopgrahy can be found in multiple tactics, namely *Defense Evasion* `[TA0005]`, *Command and Control* `[TA0011]` and *Exfiltration* `[TA0010]`.

However, in its current state[3], MITRE ATT&CK® for *Industrial Control Systems* includes methods of Information Hiding only in the tactic *Evasion* `[TA0103]`. A detailed comparison between the Enterprise and ICS versions follows in Section 2.2.5.2. This lack of details on Information Hiding in the ICS domain is also apparent in the description of malware in the current version of MITRE ATT&CK®, for example in the case of Stuxnet[4]. Stuxnet incorporates several techniques linked to the domain of Information Hiding, yet these are mainly reflected in the enterprise domain of ATT&CK® with the exception of `[T0849]` *Masquerading* in the ICS domain. These differences and especially the lack of (network-) covert channels in MITRE ATT&CK® for *Industrial Control Systems*, for example as part of Command-and-Control, highlight the need for further investigations.

The ICS Killchain and MITRE ATT&CK® show the difference in the threat landscape that operators of OT environments have to face and that networks of Industrial Control Systems are so much different from common networks that they result in new patterns of adversarial behavior and their use of previously unspecified or unknown tactics, techniques, and procedures.

While efforts to strengthen prevention, detection, mitigation, and recovery from attacks on OT environments are topics of current research projects and business developments, recent surveys show that such countermeasures in most cases are yet to be established in most businesses. A recent study with over 330 organizations illustrates this problem, in which less that 43% of the questioned organizations stated to be using an Intrusion-Detection-System (IDS) [FW19]. In addition to that recent research shows how such countermeasures, in this case Next Generation Firewalls (NGFW), could even be counter-productive as they might be leveraged to establish covert channels [MM20].

This marks the urgency to analyze what types of Information Hiding methods are applicable and in which threat scenarios they might be leveraged by adversaries to secretly infiltrate targeted CPS-Networks, to secretly exfiltrate sensible information or to remain undetected and establish covert command-and-control channels towards the adversary. From that point then, targeted measures and principles can be derived to define requirements for potential mitigation strategies.

As malware is increasingly using Information Hiding-based techniques, so does research on covert channels and network steganography seem on the rise [CCM+18, Cav21, WCM+21, CM22]. A major contribution to the understanding of network covert channels can be seen in the joint effort to create a pattern-based taxonomy by

[3] as of 2024/02/20

[4] https://attack.mitre.org/software/S0603/

Wendzel et al. [WZFH15, MWC18] intended for describing such covert communication. As the patterns are derived from real-world covert channels, they give insight into the core principles of network covert channels. However, these patterns describe network covert channels leveraging common protocols in IT-Systems, e.g. TCP/IP, DNS and HTTP which most certainly play a role in Cyber Physical Systems but are only a fraction of the whole picture. Due to different network architectures, assets, communication flows and in the control of underlying physical process by using sensors and actuators, Cyber Physical Systems are prone to other kinds of covert channels as well and not limited to IT network covert channels. Moreover, the spectrum of protocols found in CPS is quite different from common IT networks as the main communication takes places within automation protocols like Modbus [Mod21], OPC [OPC08], MQTT [OAS21] and even proprietary protocols, for example S7Comm(Plus) by Siemens [Sie19, BBC+19].
Therefore, one question is if and how known patterns and principles are applicable to Cyber Physical Systems and how covert channels in CPS might differ from common network covert channels, side channels and other types of Steganography as well as and how these facilitiate adversarial behavior.

Research explicitly on covert channels in Cyber Physical Systems seems to be still in its infancy and and focuses mainly on selected few covert channels for specific protocols or domain specific systems e.g., building automation [WKR12, WMH17a], automotive networks [YBCP19] or IoT [VMWM19]. While some of this work include attack scenarios and potential countermeasures, a broader understanding of core principles, such as a classification of techniques, participants, and potential cover for establishing covert channels and adversarial use cases in Industrial Control Systems leveraging such hidden communication is still missing. Another open question is how to counter the threat of Information Hiding-based threat scenarios in the context of Cyber-Physical Systems.

This thesis aims to provide insights helping towards such understanding and to provide initial steps for closing these gaps, shedding light on core principles how covert channels are applicable to the specifics of Industrial Control Systems, derive common methods leveraged by covert channels in that domain, and investigate how these techniques fit into the threat landscape and corresponding threat models. Such better understanding of adversarial behavior is vital to successful defending environments, and help to shrink the gap between compromise, detection and recovery.

1.3 Research Questions & Objectives

The introduction and previous section, which describes the research gaps, indicate at several points where open questions arise and investigations are needed for a better understanding of covert channels in Industrial Control Systems. This section summarizes these research gaps and formulates corresponding research questions. The research gaps from the previous Section 1.2 can be summarized as follows:

G_1 Discussion whether and how principles of Information Hiding might apply differently to the special features/circumstances and potential carriers of Cyber-Physical Systems seems missing.

G_2 A comprehensive listing or classification of potential carrier, sender/receiver and Information Hiding Methods *specific for CPS* environments seems missing.

G_3 While Information Hiding and Steganography is reflected in Adversarial Behavior Modeling (as part of TTPs) for IT, this is not the case for Operational Technology (OT). In the state of art, (adversarial-) use cases and threat scenarios of Information Hiding/Covert Channels in ICS are not investigated in depth.

G_4 Lack of clarity on which mitigation strategies against Information Hiding are available for ICS and whether they need to be adapted to the specifics of such CPS environments.

As shown in Figure 1.3 and with the aim of shrinking the identified researchs gaps, from these research gaps following research questions are derived to be answered in a selected and exemplary manner, by that illustrating the tendency for further investigations:

Q_1 How applicable are Information Hiding-based principles and techniques to CPS environments, i.e., what are plausible carrier and how do exemplary covert channels perform in terms of stealthiness and bandwidth?

Q_2 What are plausible threat scenarios for adversarial use of covert channels in ICS?

Q_3 What challenges can be encountered for detection and mitigation in Industrial Control Systems?

As shown in Figure 1.3, for each research question, a set of objectives is defined, aimed at partially answering selected aspects of the corresponding research question:

Q_1 *How applicable are Information Hiding-based principles and techniques to CPS environments, i.e., what are plausible carrier and how do exemplary covert channels perform in terms of stealthiness and bandwidth?*

- O_1 Derivation of a plausible reference architecture from the State-of-Art.

O_2 Derivation of potential covert sender and -receiver from the Plausible Reference Architecture.

O_3 Proposal for a classifcation of carrier and Information Hiding methods in CPS based on the state-of-art and own research.

O_4 Perform and analyze additional case studies on selected carrier to gain insights on plausible covert channels, parametrization and performance in terms of bandwidth and stealthiness.

Q_2 *What are plausible threat scenarios for adversarial use of covert channels in ICS?*

O_5 Discussion of plausible threat scenarios based on the covert channels covered in the case studies.

Q_3 *What challenges can be encountered for detection and mitigation in Industrial Control Systems?*

O_6 Discussion of identified challenges for potential wardens in terms detection and mitigation based on insigths from the case case studies.

In the following, these objectives are described in more detail how they aid in answering the research questions.

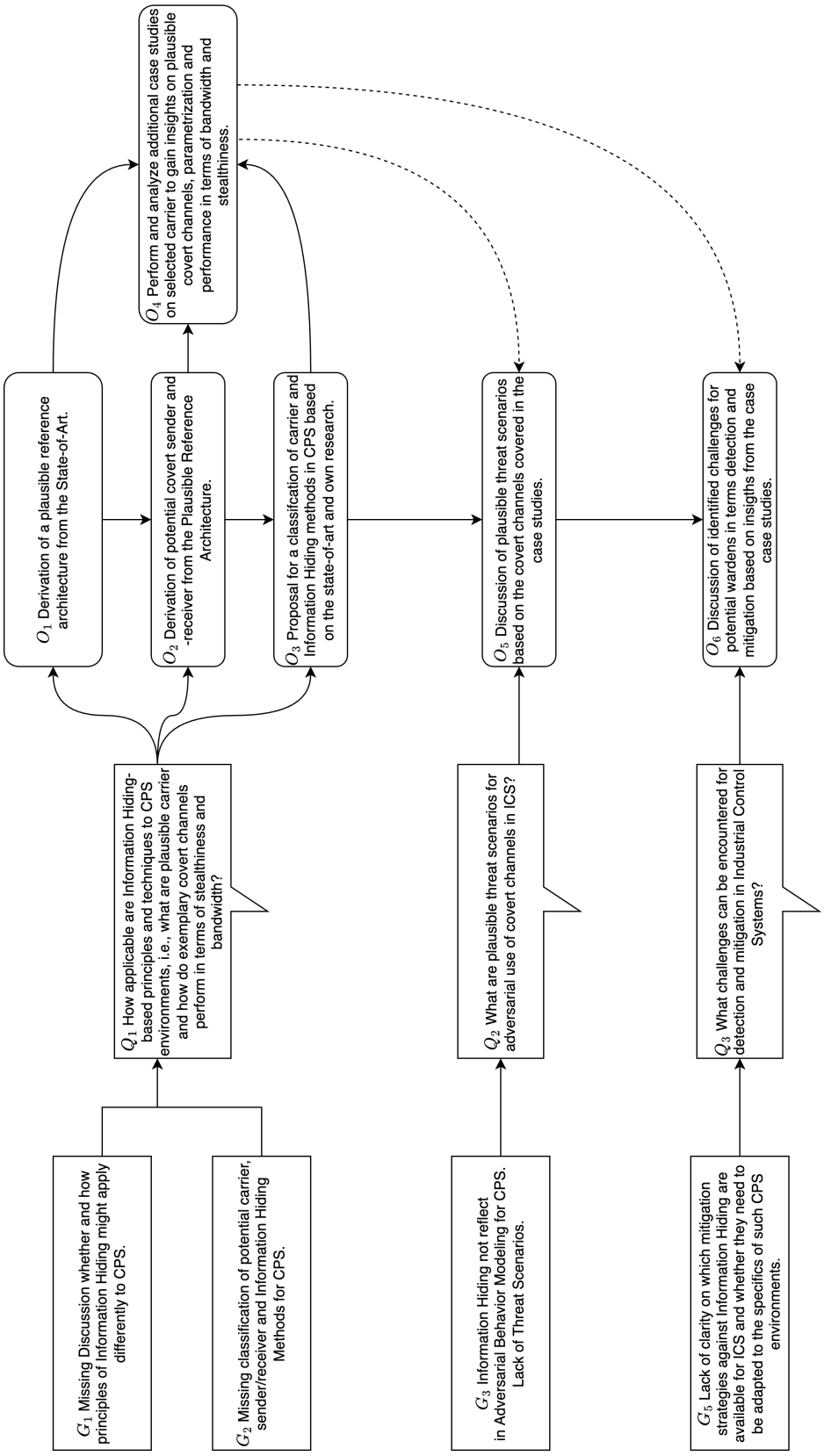


Figure 1.3: Relation of research Gaps, resulting research questions and objectives.

1.3.1 Objectives and Addressed Research Challenges

This thesis aims at providing a better understanding of the threat implications of covert channels in Industrial Control Systems and deriving insights on principles of Information Hiding in CPS, enabling and improving future mitigation. As described before, the thesis is built on three leading research questions *Q1, Q2, and Q3* filling the previously described research gaps. In the following, the objectives are described on how they aid answering the research questions and closing the corresponding research gaps.

O_1 **Derivation of a Plausible Reference Architecture** The aim of *Objective* O_1 is to provide a generalized yet representative reference architecture to be able to identify potential actors in hidden communication, plausible cover/carrier and to provide a basis for attack paths and scenarios for O_5 *Discussion of Threat Scenarios*. In Section 2.2.3 the Purdue Enterprise Reference Architecture, the NIST SP 800-82 alongside other guidelines and recommendations are described. The idea is to combine architectures, components and assets of these to achieve a plausible reference architecture representing the current state in terms of security-aware architectures for ICS. This also aids later to develop and describe potential threat scenarios.

O_2 **Derivation of Potential Covert Sender and -Receiver** From the Plausible Reference Architecture of O_1 components of common Industrial Control Systems are identified that might act as covert sender and -receiver for covert channels.

O_3 **Proposal for a Classifcation of carrier and Information Hiding methods in CPS** Based on the Plausible Reference Architecture (O_1), potential covert sender and -receiver (O_2) and selected publications from the State-of-Art a classification for potential carrier and their related Information-Hiding methods is proposed.

O_4 **Case study analysis** Perform and analyze case studies on selected carriers to gain additional insights on the plausibility of carriers, covert channel identification and -design, as well as parametrization, performance and migitigation. Practicical testing of selected carriers and covert channels in regard to performance metrics such as capacity, bandwidth, detectability and robustness. Beside these, another aim is to gain insights on different techniques for embedding, retrieval, synchronization, and key exchange in CPS and to derive insights on Information Hiding in Cyber-Physical Systems in general.

O_5 **Discussion of Threat Scenarios** At the example of the case studies of O_4, threat scenarios are derived and discussed. The idea is to aggregate the findings and put them into perspective within composite attack scenarios, i.e., to evaluate *how* an adversary might use the investigated covert channels to achieve certain goals.

O_6 **Discussion of identified Challenges for Mitigation** From the findings of the previous objectives, challenges for mitigation are discussed. This includes considerations on the placement of a warden as well as other security critical components, like data diodes, network taps, event aggregation system, IDS/IPS and

similar. The aim is to define a provide an understanding of methods and their limitations as well as providing first ideas, that can be used in the future to design CPS-architectures that are robust against covert channels and to provide a unified approach for prevention, detection, elimination and suppression of Information-Hiding based attack scenarios.

1.4 Publications contributing to this Thesis

As illustrated in Figure 1.4, this work rests on four pillars of publications:

1.4.1 Covert Channels in Automation Protocols

Modbus/TCP *"Assesment of Hidden Channel Attacks: Targetting Modbus/TCP"* [LD20]: This paper covers a systematic analysis of Modbus/TCP as carrier for hidden information and is part of Case Study CS_1.

OPC UA *"Information Hiding in Industrial Control Systems: An OPC UA based Supply Chain Attack and its Detection"* [HLD+20]: This paper features a covert channel in OPC UA within timestamp fields, which plays a major role in follow-up publications.

1.4.2 CPS Steganography

Process Data Information Hiding in Cyber Physical Systems: Challenges for Embedding, Retrieval and Detection using Sensor Data of the SWAT Dataset [LNK+21]: This paper discusses covert channels in process data transmitted towards historians. This work is the basis for the case study CS_2.

1.4.3 Covert Channels in Infrastructure Protocols

NTP *"A Systematic Analysis of Covert Channels in Network Time Protocol"* [HLKD21]: A systematic analysis of the NTP protocol in terms of covert channels. One of three publications contributing to Case Study CS_3.

NTS *"Covert Channels in Network Time Security"* [LD22]. Discussion of covert channels in NTS, a cryptographic extension for NTP. Second publication that is part of Case Study CS_3.

NTP/PTP *"The Threat of Covert Channels in Network Time Synchronization Protocols"* [LHKD22]: Discussion of threat scenarios in NTP and the Precision Time Protocol (PTP), the third publication included in the Case Study Case Study CS_3.

Syslog/Portscans *"Knock, Knock, Log: Threat Analaysis, Detection & Mitigation of Covert Channels in Syslog using Port Scans as Cover"* [LNH+22]: Using harmless-looking portscans to trigger logging-mechanisms and by that encoding hidden information.

1.4.4 Threat & Mitigation

Defense-in-Depth *"Resilience Against and Detection of Information Hiding in Nuclear Instrumentation and Control Systems within the Scope of NSS 17-T"* [LHA+22]: Discussion of Defense-in-Depth architectures (e.g., definition of security zones and -levels) as potential countermeasures.

Anomaly Detection *"Novel Challenges for Anomaly Detection in I&C Networks Strategic Preparation for the Advent of Information Hiding based Attacks"* [LNL+20]: Discussion of challenges for anomaly detection based detection schemes in OT networks.

Threat Scenarios *"Threat Analaysis of Steganographic and Covert Communication in Nuclear I&C systems" [HAL+20]*: Discussion of threat scenarios using covert channels in nuclear OT systems.

Event Aggregation *"Enhancing Safety and Security of Digital Instrumentation and Control System by Event Aggregation"* [AZL+21]: Discussion on how Event Aggregation might be employed in ICS for threat detection.

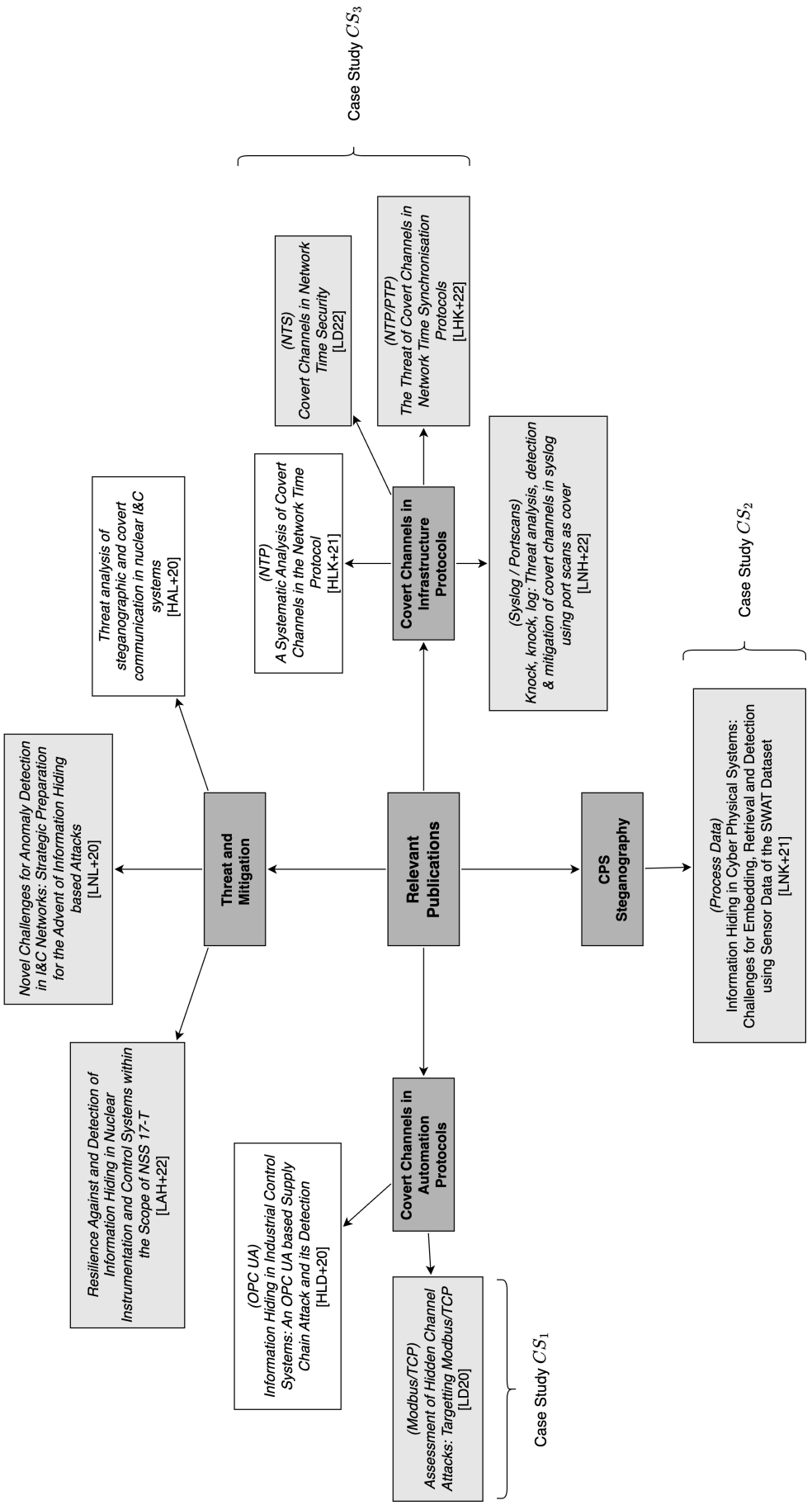


Figure 1.4: Categorization of own publications contributing to this thesis. Light grey boxes indicate first-author publications.

1.5 Summary of the Contributions of this Thesis

Based on the research questions and objectives, the contributions of this thesis can be summarized as follows:

C_1 Proposal of a plausible reference architecture (PlauRA) for Industrial Control Systems that allows the analysis of potential covert communicators, carrier and derivation of threat scenarios and countermeasures.

C_2 Description of potential covert sender, receiver and corresponding carrier for Industrial Control Systems, based on the plausible reference architecture (PlauRA) and the State-of-Art.

C_3 Proposal for a classifcation of Information Hiding methods in Cyber-Physical Systems derived from the State-of-Art and own investigations and publications.

C_4 Description and proposal for a classification of adversarial use of IH-methods for covert lateral and vertical communication in defense-in-depth architectures at the example of ICS.

C_5 Detailed investigation of three distinctive types of carriers (Automation Protocols, Process Data and Infrastructure Protocols) in the course of three selected exemplary case studies with a detailed description and analysis of plausible covert channels, performance, resulting threat scenarios and mitigation.

C_6 Discussion of lessons learned for Information in Cyber-Physical Systems, including plausibility, challenging features of such systems, techniques for cover object selection, bandwidth modulation as well as the persistence and location of cover objects.

C_7 Derivation of a classification of (stego-) key distribution schemes from the case studies and State-of-the-Art.

C_8 Proposal for an extended mitigation and countermeasure model and discussion on potential design principles for covert channel-robust architectures as well as general discussion on detection and mitigation based on the results of the investigated covert channels from the case studies.

In order to facilitate transparency, reproducibility, and further exploration of the research findings, datasets and corresponding code samples that are used in the case studies are published independently as open-access repository and is available under [Lam23][5].

In the following, the structure of this thesis is outlined.

[5] https://doi.org/10.24352/ub.ovgu-2023-112

1.6 Thesis Outline

In the next Chapter 2, related work and fundamentals required for this thesis are described. Chapter 3 discusses principles of Information Hiding in CPS and provides the theoretical frame for the case studies. Chapter 4 describes the case studies. In Chapter 5 key insights derived from the case studies are discussed and the results of this thesis summarized. Chapter 6 gives an outlook on potential future work and remaining open questions.

2. Thesis Fundamentals and Related Work

This chapter provides the required fundamentals and related work relevant for this thesis. These fundamentals are split into two main topics: The first Section 2.1 describes fundamentals of Information Hiding and Section 2.2 is dedicated to topics related to Cyber Physical- and Industrial Control Systems.

2.1 Selected required Fundamentals of Information Hiding

This section gives relevant background information on core principles of Information Hiding, Stegoware, Network Covert Channels, a differentiation to side channels and gives a overview on published covert channels in the domain of Cyber Physical Systems.

2.1.1 Information Hiding

Domains of Information Hiding had been categorized at several occasions in the past, e.g. in [PAK99], [CA16] and [MWZ+16]. Common ground, however, seems to be the categorization of Petitcolas et. al. [PAK99], in which Information Hiding is categorized as the umbrella term for Covert Channels, Steganography and Watermarking. This categorization is illustrated in Figure 1.1 with extensions from [CA16] and [MWZ+16].

2.1.2 Steganography

Steganography constructs hidden communication channels by relying on the fundamental assumption that so called *cover objects* or *carrier* (in the literature also *overt communication* and *cover channels*) are available that can be leveraged to secretly communicate *hidden data* (also called *hidden information*, *hidden message* and *secret message*).

C.E. Shannon describes this *cover channel* as "[...] *the medium used to transmit the signal from transmitter to receiver.*" [Sha48], i.e. as any means of communication connecting a sender with one (or more) receiver(s) using a pre-negotiated communication protocol.

In the past the terms Information Hiding, Covert Channels and Steganography have blurred and are often used interchangeably. In their book "*Information Hiding in Communications Networks*" Mazurczyk et al. [MWZ+16] describe the development of these terms and describe the "*main difference between the terms steganography and covert channels is merely the type of the carrier (cover) used; for steganography it is digital media like images, audio, and video files, whereas for covert channels these are network protocols.*" Therefore, the authors propose to "*use the term network steganopgrahy to describe the 'methods' used for creating covert channels in communication networks*" [MWZ+16].

2.1.3 Covert Channels

The term *Covert Channel* has been characterized by Butler W. Lampson in his 1973 paper "*A note on the confinement problem*" as any communication performed over a channel that is originally "*not intended for information transfer at all, such as the service programs effect on system load*" [Lam73].

Besides Lampson, also others have phrased covert channels in a similar fashion, e.g. Matt Bishop [Bis02] as "*a path of communication that was not designed to be used for communication*" or the NIST [Lat86] as "*any communication channel that can be exploited by a process to transfer information in a manner that violates the systems security policy*". The latter definition is of special interest for this thesis because the part on *security policies* integrates the use or search for information hiding techniques into the scope of *security management* and addresses the difference of wanted information hiding (e.g. in the form of watermarking) vs. unwanted information hiding (e.g. by steganography or covert channels).

In summary of the previous definitions, the general understanding for this thesis is, that *cover objects* are transferred over *overt (cover) channels*. In these *cover objects/carrier hidden information* is embedded, by that establishing a covert (communication) channel.

As described before, such covert channels are typically further distinguished in *Timing Channels*, i.e., the modulation of temporal aspects of the cover channel or objects and *Storage Channel*, i.e., the modification of contents of the cover channel or object.

Besides this general distinction, several sub-types of Covert Channels can be found in the literature. As described in Section 1.1.2, which sets the scope for this thesis, relevant sub-types for this thesis are *Network Covert Channels* [MWZ+16], *Behaviour-based Covert Channels* [ABP+19, JLY10] and *Out-of-Band Covert Channels* [CA15, CA16]. Arguably, covert channels created by the application of *Cyber-Physical System Steganography* could be referenced as *Cyber-Physical System (CPS) Covert Channels* - however, this term has not yet gained broader usage in the literature.

2.1.4 Subliminal Channels

In his seminal work in 1984, "*The prisoners' problem and the subliminal channel*" [Sim84], Gustavus J. Simmons laid the foundation for describing covert channel based communication scenarios with the formulation of the so-called *Prisoner's Problem*. In this hypothetical scenario Alice and Bob are imprisoned in different cells and want to plan their escape. They are allowed to communicate, however, each message is inspected by a *Warden*. If the warden would find any sign of illicit communication, Alice and Bob would be put in solitary confinement, eliminating any chance of escape. The solution to this problem, proposed by Simmons, is the use of *subliminal channels*, a steganographic method to establish *covert channels* utilizing a cryptographic protocol as carrier for steganograms [Sim84]. As the carrier for subliminal channels are cryptographic methods, these are also called *cryptosystem steganography* in the literature and form a sub-type of covert channels [MWZ+16]. Even though Simmons was focusing on subliminal channels, the *Prisoner's Problem* has become the foundation for hidden communications scenarios and sets the scope for any research conducted in this field. The *Prisoner's Problem* formulates following basic hidden communication scenario which will be used in this thesis.

2.1.5 Hidden Communication Model (*Prisoners' Problem*)

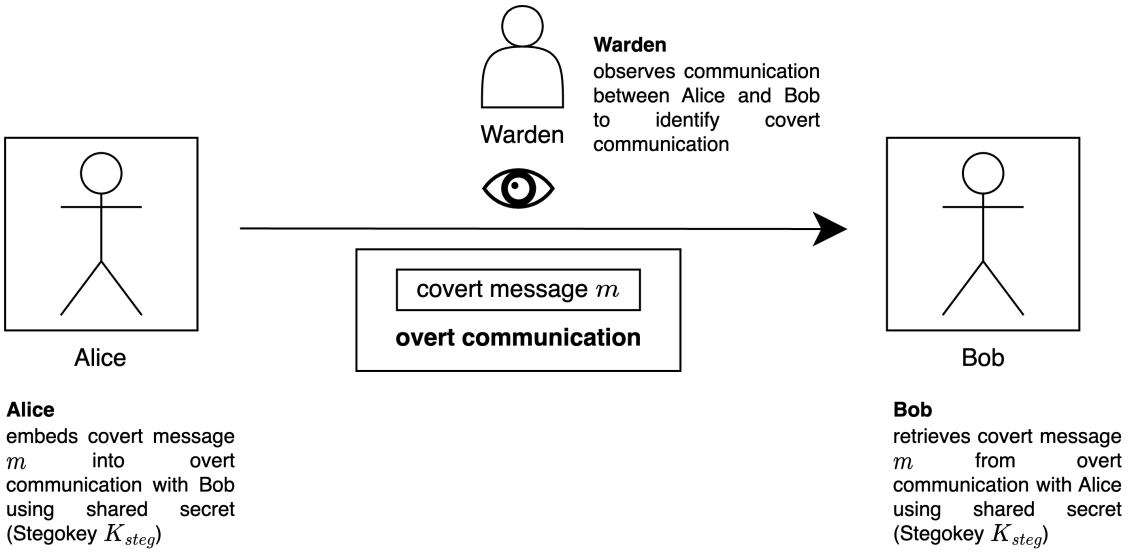


Figure 2.1: Hidden Communication Model with Alice, Bob and a passive Warden based on Simmons Prisoner's Problem [Sim84].

In 1984 Gustavus J. Simmons describes the *Prisoners' Problem*, a hypothetical communication scenario in which two persons Alice and Bob have been arrested an locked in separated cells [Sim84]. Their only way of communication is by sending messages known to be monitored by a warden. The aim of Alice and Bob is to coordinate an escape plan. The warden is willing to permit the exchange of communication if the content of the message is presumably innocuous. If he finds any sign of illicit communication, Alice and Bob are put in solitary confinement without the possibility to communicate. Therefore, the aim of Alice and Bob is to secretly communicate

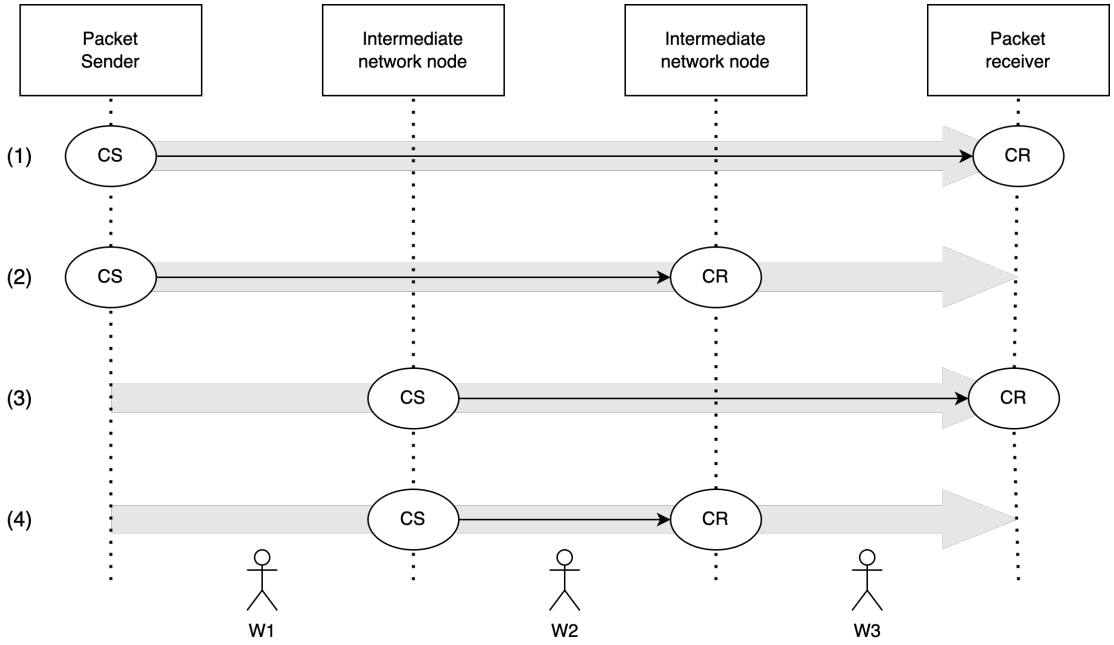


Figure 2.2: Hidden communication scenarios and potential localizations of the warden for network covert channels. CS: Covert Sender; CR: Covert Receiver; Warden Positions W1-W3. Modified figure based on [MWZ+16].

in the full view of the warden, i.e. Alice and Bob have to hide their secret communication (covert message) in the overt communication monitored by the warden. As illustrated in Figure 2.1, Alice and Bob make use of steganographic methods to embed a covert message m into the overt communication using a shared secret between Alice and Bob - the so-called stego-key K_{steg} which drives the embedding algorithm (where and how to hide). Using this key Bob can retrieve the hidden message from the overt communication. The assumption in the Prisoners's Problem is that Alice and Bob were able to exchange the stego-key (shared secret) before they got imprisoned. In real-world malware incidents, the stego-key is often static and hard-coded or the embedding algorithm uses fixed parameters (e.g., [Del15, Rha23]). These fixed parameters can be seen as a stego-key as the (covert) receiver requires these exact parameters to be able to retrieve the hidden message from the carrier/-cover object. In the context of the (Lockheed Martin) Cyber Kill Chain®[HCA11], the key exchange or key definition, respectively, takes place during weaponization (creation of the malware with hard-coded key or parameters) or on delivery (setting parameters or stego-key). In most scenarios the stego-key is a symmetric key, i.e., both sender and receiver use the identical key.

In the literature, three types of wardens are differentiated: *passive Wardens* only monitor the communication without interfering with it, *active Wardens* in contrast are modifying the communication to remove any hidden message [MC15]. A sub-type of active Wardens is the *malicious Warden*, who hijacks the covert communication and acts as a man-in-the-middle to alter or spoof messages [Cra98].

The placement of network covert channel wardens is discussed in [MWZ+16]. Figure 2.2 shows the importance of the positioning of these wardens for successful detection. The distinction between *active* and *passive* Information Hiding is based

on the position of the secret sender and receiver. If the sender is the origin of the packet and the receiver is the destination (as in (1) of Figure 2.2), it is referred to as *active* Information Hiding (cf. [DHH05]). If the sender and receiver are intermediate nodes (as in (4) of Figure 2.2), it is considered to be a passive information hiding scenario. Related to these positions of the warden and the distinction of where information is embedded and retrieved, additionally, the terms *semi-active* and *semi-passive* (*hybrid*) scenarios (covering scenarios (2) and (4) of Figure 2.2) were coined in our paper [LD20]. These are also further discussed in the context of ICS in Section 3.5.1.1.

2.1.6 Side Channels and Out-of-Band Covert Channels

Another distinction to make is between the terms "Side Channels" and "Out-of-Band Covert Channels". Again,these two terms are often used synonymously in the literature. However, in 2015 Carrara and Adams coined the term *Out-of-Band-Covert channels* in [CA15] and [CA16], which are categorized as a subtype of covert channels and make a distinction to *Side Channels*: *"An out-of-band covert channel is a low probability of intercept (LPI) communication channel established between isolated processes (i.e. processes not able to communicate through traditional links) by modulating and demodulating a shared medium using devices that are traditionally not used for communication"* [CA16]. Side Channels in contrast leak information *unintentionally* [MWZ+16]. In their comprehensive study on Out-of-Band Covert Channels Carrara et al. propose a taxonomy with following categories: (1) Acoustic, (2) Light, (3) Vibration, (4) Magnetic, (5) Temperature, (6) Radio-frequency [CA15, CA16]. As the main distinction to other covert channels is that Out-of-Band Covert Channels do not require any overt communication, the common use case of Out-of-Band Covert Channels are to circumvent *Air Gaps*, the most strict form of network segregation. Prominent examples are *Bitwhisper* (thermal) [GMME15], *LED-it-go* [GZE17] (light), *Glowworm* [NPG+21] and *Powerhammer* [GZBE20].

2.1.7 Steganographic Bandwidth, Robustness, Undetectability (*the "magic triangle"*) and Steganographic Cost

In Steganography, generally three types of metrics are used that form the so-called *magic triangle*, first introduced by Jessica Fridrich in 1999 [Fri99] and illustrated in Figure 2.3. These are *steganographic bandwidth*, *robustness* and *undetectability*. Instead of bandwidth, often *capacity* is used to describe the amount of hidden information per carrier (object). Undetectability is sometimes also referred to as *stealthiness* or *conspicuousness* (e.g., [LD20]) of a steganographic method. Robustness defines how much a steganogram (embedded, hidden message) can withstand alteration without loss of (secret) information. Probable alterations are for example compression, protocol-conversions, noise or active wardens. The magic triangle describes the relationship between these features:

1. With higher bandwidth/capacity detection is more likely.
2. To reduce detection probability the secret sender has to either sacrifice steganographic bandwidth or robustness.

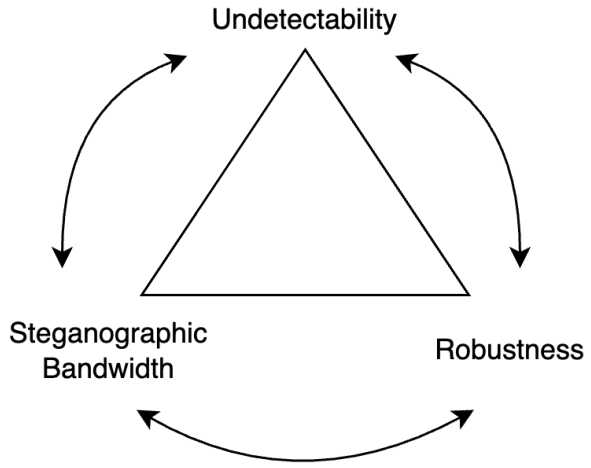


Figure 2.3: Relationship between Steganographic Bandwidth, Robustness and Undetectability as part of the *magic triangle* as introduced by Fridrich [Fri99]. Figure based on [MWZ+16].

3. To increase robustness, either steganographic bandwidth is sacrificed (e.g., for redundancy) or detection probability.

The underlying reason for this relationship, is a concept called *steganographic cost* [MWAVS16]. The core idea is, that by embedding secret information into a carrier object, the carrier object gets degraded. The more information is embedded, the more the carrier is degraded, and more likely is detection by a warden.

2.1.8 Network Information Hiding Taxonomy

In 2015, Wendzel et al. introduced the pattern-based taxonomy to describe patterns of Information Hiding in Network Steganography [WZFH15]. Over the years, with several improvements and overhauls [MWZ+16, MWC18], it became the de facto standard for describing network covert channels in recent research.

It is based on the analysis of more than one hundred techniques and summarizes them into 18 general patterns. In the extended taxonomy, published in 2018 [MWC18], covert channels are distinguished in eight timing patterns and ten storage patterns.

In 2022, this pattern-based taxonomy was incorporated into a domain-agnostic generic taxonomy [WCM+22]. The development of this generic Information Hiding taxonomy is currently still ongoing and done by a consortium of multiple reknown researchers in the field of Information Hiding[1].

2.1.9 Selected relevant publications connected to Cyber-Physical System Steganography

In the following, a brief overview is given on selected, for this thesis relevant publications with connections to Steganography in Industrial Control Systems or Cyber-Physical Systems in general.

[1] https://patterns.ztt.hs-worms.de/

2.1.9.1 Network Covert Channels in OT protocols

- *Comprehensive analysis of MQTT 5.0 susceptibility to network covert channels* [MVH+21]: A systematic analysis of network coverts in the MQTT protocol. MQTT is a common protocol found mainly in IoT but also ICS networks. In their work, Mileva et al. discuss 23 identified covert channels for MQTT 5.0.

- *Covert Channels in the MQTT-Based Internet of Things* [VMWM19]: Description of seven direct and six indirect covert channels in MQTT.

- *Covert Channels and their Prevention in Building Automation Protocols – A Prototype Exemplified Using BACnet* [WKR12]: Network Covert Channels in BACnet, a protocol commonly found in Building Automation Systems (BAS).

- *New Covert Channels in Internet of Things* [MVS18]: Covert channels in the Constrained Application Protocol (CoAP), typically found in (IoT) devices with low computation power or bandwidths.

- *Reset- and Reconnection-based Covert Channels in CoAP* [HZW21]

- Network Covert Channels in Modbus/TCP

  - *Covert Channels-based Stealth Attacks in Industry 4.0* [ABP+19]: In their work, Alacaraz et al. described network covert channels in Modbus/TCP and also describe Behaviour-based covert channels in the context of CPS.
  - *Modbus Covert channel* [LJ14]: Proposal of a covert channel using register numbers to encode ASCII symbols in request packets.
  - *A Modbus command and control channel* [LFK16]: Proposal of a covert channel using the Least-Signficant-Bits (LSB) of holding registers (payload).
  - *A timing-based covert channel for SCADA networks* [LK17]: Proposal of a Timing channel in Modbus/TCP.

2.1.9.2 Physical Covert Channels

- *Covert channel communication through physical interdependencies in cyber-physical infrastructures* [GSMZ14]: Covert communication scenarios using physical properties, in this case power line loads in smart grids.

- *The Chatty-Sensor: A Provably-covert Channel in Cyber Physical Systems* [HK19a]: Modification of sensor values to modulate the timing when a certain threshold for the control loop is reached.

- *The Leaky Actuator: A Provably-covert Channel in Cyber Physical Systems* [HK19a]: Using a high-quality actuator and adding arbitrary/artificial delays in the response time of the actuator to encode a mesage.

2.1.9.3 Covert Channels using Sensor Data

- *Covert Channel Establishment Through the Dynamic Adaptation of the Sequential Probability Ratio Test to Sensor Data in IoT* [Ho19]: Establishment of Covert Channels in IoT Sensor Data using the Sequential Probability Ratio Test (SPRT).

2.1.9.4 Covert Channels in the Control Logic/Loop

- *Wyner wiretap-like encoding scheme for cyber-physical systems* [LY20]: Wyner wiretap-like encoding scheme for covert communication in control signals within a control loop.

- *Covert Channels in Cyber-Physical Systems* [ALY21]: Establishment of Covert Channels in a control loop by altering the control logic of a controller (e.g., PLC).

- *Covert channels instochastic cyber-physical systems* [LY21]: Covert Channels, observable in process data, embedded by alterations of the control logic.

2.1.9.5 Out-of-Band CPS Channels

- *Process-Aware Covert Channels Using Physical Instrumentation in Cyber-Physical Systems*[KKK+18]: Using the analogue emissions of physical instrumentation (e.g., actuators, sensors, and mechanical structures) of a (CPS) to send or leak information, demonstrated at the example of an acoustic channel of a motorized valve.

- *WaterLeakage: A Stealthy Malware for Data Exfiltration on Industrial Control Systems Using Visual Channels* PLC LEDs Exfil, [RDMMR19]: Data Exfiltration from PLCs leveraging LEDs.

- *"Robot Steganography"?: Opportunities and Challenges* [CJV21]: Covert Channels established by robots using physical observables e.g., actuator positions, velocity, position, lights, sounds etc.

2.1.9.6 Covert Channels in Smart Buildings/Building Automation Systems

- *Covert and Side Channels in Buildings and the Prototype of a Building-aware Active Warden* [Wen12a]: High-level covert and side channels in BAS (Building Automation Systems) on the application layer and discussion of active wardens in such systems.

- *Don't You Touch My Nuts: Information Hiding in Cyber Physical Systems* [WMH17a, WMH17b]: Secret data storage in Smart Buildings.

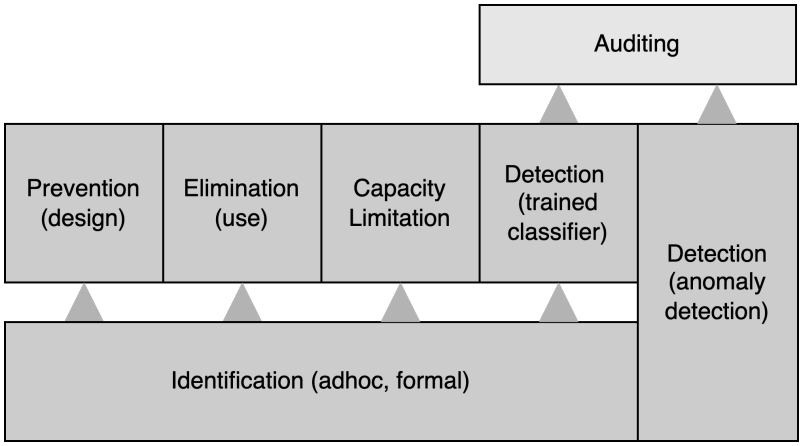


Figure 2.4: Countermeasures against covert channels as classified by Mazurczyk et al. [MWZ+16].

2.1.10 Countermeasures against Information Hiding

In their book "*Information Hiding in Communication Networks*" [MWZ+16] Mazurczyk et al. provide a basic classification of countermeasures against covert channels: *Prevention*, *Elimination*, *Capacity Limitation* and *Detection*. Methods of detection are further differentiated in *anomaly*-based detection and detection using *trained classifier*.

As illustrated in Figure 2.4 these methods can either be used exclusively or used in combination. In their model, *Prevention*, *Elimination*, *Capacity Limitation* and *Detection using trained classifier* require prior *identification* of the covert channel that is to be countered. Once a covert channel for a specific protocol is identified, one might take actions to *prevent* the establishment of the covert channel, e.g., by using a different protocol which is not affected by the covert channel or known to be more robust against information hiding.

If prevention is not possible or feasible, it might be possible to *eliminate* the covert channel, e.g., by using *traffic normalization*. If a covert channel cannot be prevented nor eliminated, the next option is to take actions in order to *limit* the *capacity* of the channel, eventually by that rendering it useless, e.g., by the introduction of noise.

As pointed out in a recent survey paper, covert channel detection methods have been dominated by Machine-Learning based approaches in the last years [EG22]. Besides these, another way that got attention in the last years is to make common (Network-) Intrusion Systems aware of covert channels [KWJ21, Cav21, Zan17, Gun17, Wen12b].

The two approaches discussed in this thesis have their own advantages and disadvantages, which will be discussed in the case studies. Each case study will look at countermeasures, but the main focus will be on the general principles of prevention and detection, as a more detailed examination of detection would be too extensive and is out of scope for this thesis.

2.2 Selected required Fundamentals on the Security of Cyber Physical- & Industrial Control Systems

This section provides relevant background information on the definition and components of Cyber Physical and Industrial Control Systems, common architecture models and adversarial behavior models.

2.2.1 Definition of Cyber Physical Systems

The term Cyber Physical Systems is a broad term and as pointed out in [KM15] has been defined on many occasions (e.g., [GRSS12, CDB+12, SGLW08, HG12, LMS18]) and overlaps with other terms e.g., *Internet of Things (IoT)* [DS21]. According to [DS21] the term has been coined by H. Gill (National Science Foundation, NSF) in 2008 [Gil08] and has since evolved to include yet not to be limited to systems of several domains, e.g., communication (sensor networks), energy (e.g., production and distribution), infrastructure (e.g., water distribution), manufacturing, military (e.g., UAVs), physical security (access control), robotics, smart buildings and transportation (automotive, avionics, railroads). However, a common ground seems to be the description of "*systems that offer integrations of* ***computation, networking, and physical processes***" [KM15] and the definition by the U.S. National Science Foundation [Nat21] which describes Cyber Physical Systems as "*engineered systems that are built from, and depend upon, the seamless* ***integration of computation and physical components***". Additionally, the National Institute of Technology (NIST) has defined six basic characteristics of CPS in the NIST SP 1900-202 [GBWG19] representing patterns found in several definitions of CPS, which also happen to be consistent with six characteristics identified by the Platforms4CPS consortium [Pla18]. In 2021, [DS21] DeFranco and Serpanos compared 12 definitions from key CPS stakeholders against these six CPS characteristics. While each definition maps to several of these characteristics at least one characteristic is missing in each definition. However, this comparison illustrates and emphasizes that these six CPS characteristics are a solid foundation to describe and define CPS. Therefore, as it is the common ground, these six characteristics of the NIST SP 1900-202 [GBWG19] as summarized in [DS21] are used to define CPS in the context of this thesis:

Hybrid systems A CPS consists of **physical** and **logical** elements e.g., a feedback loop between **sensors** and **actuators**.

Hybrid methods "**Software** to join the integrated physical and logical systems, comprising the **networking**, **information processing**, **sensing**, and **actuation** that allow the **physical device** to operate in a **changing environment**." (wording of [DS21])

Control "Using computational systems to **control physical processes** and engineered systems, such as to **monitor, coordinate, and control physical operations using computing and communication**." (wording of [DS21])

Component classes CPS consists of sensors, actuators, control systems, information technology (IT systems) including networking/communications.

Time Event-driven computation in a physical-world timeline.

Trustworthiness CPS require safety, reliability and security.

As described before, common examples for Cyber Physical Systems are Smart Grids, Automotive Systems, Aviation Systems, Building Automation Systems, Smart Homes and Industrial Control Systems (ICS) [KM15], which are the focus of the thesis and are defined in the following.

2.2.2 Definition of Industrial Control Systems

Industrial Control Systems are defined by the National Institute of Standards and Technology (NIST) in the Special Publication 800-82 (NIST SP 800-82) as "*A communication network of actors, sensors and processing units geared towards controlling a physical process*" [SPL+15]. This definition describes on the one hand the purpose of such systems and on the other hand its components: The purpose of Industrial Control Systems is the control of a physical process, e.g. for manufacturing goods or generating power. Its core components are sensors which provide information on the physical surroundings, actuators which influence the physical environment and processing units which control the physical environment by using the information gathered by sensors and influencing the environment using actuators.

However, as 'Cyber Physical System' is a general term so is the term of Industrial Control Systems and often includes subsystems, such as *Supervisory Control and Data Acquisition (SCADA)*, *Programmable Logic Controllers (PLCs)* and *Remote Terminal Units (RTUs)*. According to the NIST SP800-82 [SPL+15] "*SCADA systems are designed to collect field information, transfer it to a central computer facility, and display the information to the operator graphically or textually, thereby allowing the operator to monitor or control an entire system from a central location in near real time*". In other words the purpose of SCADA systems is the collection, aggregation and visualization of relevant process information for the operation of the controlled physical process(es). Remote Terminal Units (RTUs) and Programmable Logic Controllers are aforementioned processing units for controlling the local process using actuators and sensors. RTUs are mainly used for remote locations, e.g. in the case of distributed field sites in smart grid environments whereas PLCs are commonly used for controlling local processes, e.g. in a power plant. Other relevant components of ICS are Human-Machine-Interfaces (HMI), Engineering Workstations (EWS) and Data Historians. How these components are interconnected and interworking is modeled in the widely accepted *Purdue Enterprise Reference Architecture (PERA)* [Wil92] and shown in Figure 2.5.

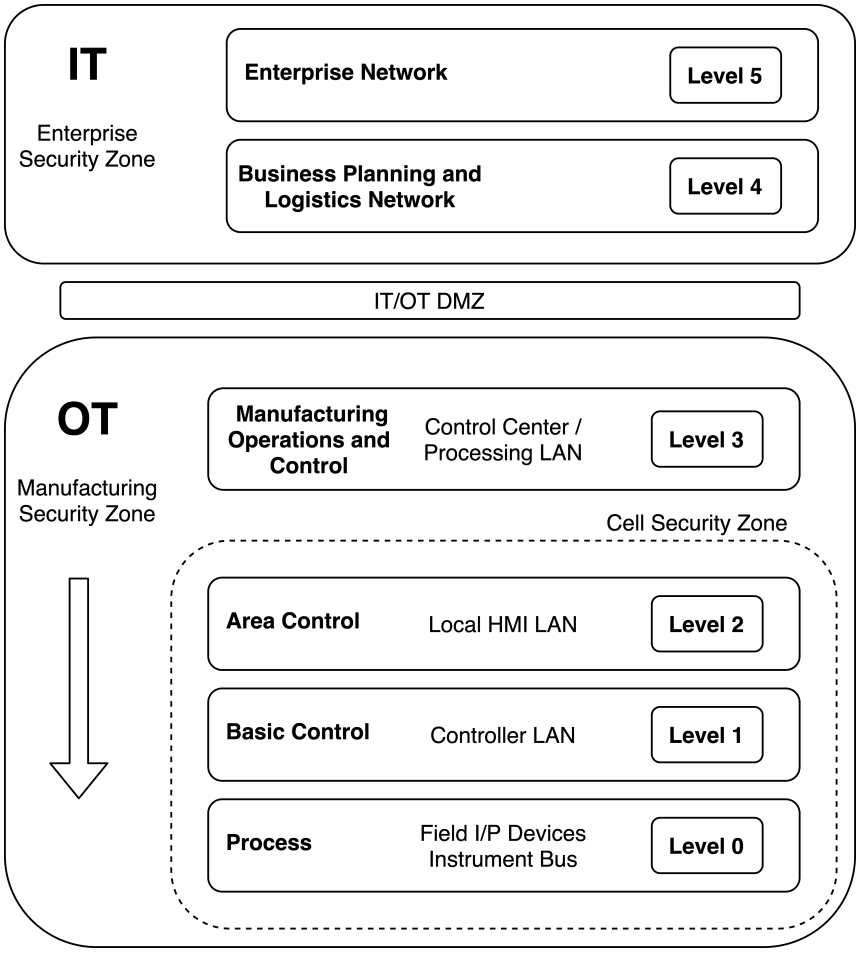


Figure 2.5: IT/OT Differentiation [HS16] based on the Purdue Enterprise Reference Architecture (PERA) Levels [Wil92] plus demilitarized Zone between IT and OT as suggested by [SPL+15].

2.2.3 Purdue Enterprise Reference Architecture (PERA)

The Purdue Enterprise Reference Architecture includes six functional levels, which can be summarized as follows [Alt20, Cis11]:

- **Level 0 - Process** Sensors and Actuators
- **Level 1 - Basic Control** Controllers, e.g. PLCs or RTUs interfacing with the Level 0 devices
- **Level 2 - Area Supervisory Control** Supervision and Operation, e.g. Local Human-Machine-Interfaces, Alerting Systems
- **Level 3 - Site Level** Plant-wide Operation and Control, common components are: Plant Historian, Engineering Workstations, Control Room HMIs, and network services, like Active Directory (AD), Dynamic Host Configuration Protocol (DHCP), Dynamic Naming Services (DNS), Time Protocol (NTP)
- **Level 4 - Site Business Planning and Logistics** Basic business administration tasks based on standard IT services, e.g. scheduling systems manufacturing

execution systems (MES), and local IT services, for example E-mail and printing.

- **Level 5 - Enterprise** Corporate level applications (for example, ERP, CRM, document management) and services (Internet access, VPN entry)

However, the Purdue Enterprise Reference Architecture was not designed with security in mind and only provides a differentiation of functional levels. Albeit being a high level model it still provides a first starting point to identify potential communication flows relevant for the analysis of potential covert channels. For deeper analysis of common architectures in the following the most prevalent recommendations and guidelines regarding ICS security are described.

2.2.4 ICS Security Guidelines and Recommendations

In the following the security guidelines and recommendations of the NIST SP 800-82, the SANS Secure Architecture for Industrial Control Systems, the ICS-CERT Recommended Secure Network Architecture and the IAEA Grade Approach as published in the NSS-17T are described.

2.2.4.1 NIST SP 800-82

One of the most comprehensive and referenced guidelines for Industrial Control System security is the NIST Special Publication 800-82 published in 2015 [SPL+15]. The NIST SP 800-82 provides detailed guidelines on risk management, how to employ an ICS Security Program and provides recommendations for a defense-in-depth architecture.

Similar to the IT/OT DMZ illustrated in Figure 2.5 this architecture mainly focuses on the *network segregation* of the corporate network (IT) and the control network (OT) using a demilitarized zone (DMZ) (yet neglecting threats that reside within the OT network segment). The concept of this defense-in-depth strategy is to forbid any direct communication between control network and corporate network. Instead only (firewall) filtered communication from both networks is allowed towards the DMZ, aiming to prevent adversaries accessing critical components from the corporate network. This marks the importance of the DMZ and its components as those are potential pivot points for adversarial intrusions and in combination with further guidelines on firewall policies the NST SP 800-82 gives useful insight on how covert channels might be used to pivot from the IT to OT or vice versa as it also covers details on recommended firewall rules. For example an adversary might not be able to pivot across the IT-OT-firewall using common protocols as they are filtered but is able to enter the OT network using covert channels within the legitimate traffic from IT to DMZ and then pivot from DMZ to OT.

2.2.4.2 SANS Secure Architecture for ICS

In the same year (2015) the SANS Institute published a modified reference architecture [Obr15] based on the NIST SP 800-82 and the Purdue Enterprise Reference Architecture [Wil92] with the first introduction of different zones by the addition of two monitoring zones for the corporate and control network, each comprising a log collector, an Intrusion-Detection-System (IDS), and a Security Information and Event Management System (SIEM).

2.2.4.3 ICS-CERT Recommended Secure Network Architecture

In 2016, the Department of Homeland Security (DHS)'s National Cybersecurity and Communications Integration Center (NCCIC) and Industrial Control Systems Cyber Emergency Response Team (ICS-CERT) published further recommendations aiming at improving the Defense-in-Depth strategies [HS16]. Besides general recommendations a further refined architecture is described similar to architectures of the NIST SP 800-82 [SPL+15] and the SANS Secure ICS Architecture [Obr15] but providing more details on used systems and components as well as different firewall segregated zones in the OT environment on PERA levels 1 and 2.

2.2.4.4 IAEA Graded Approach (NSS-17T)

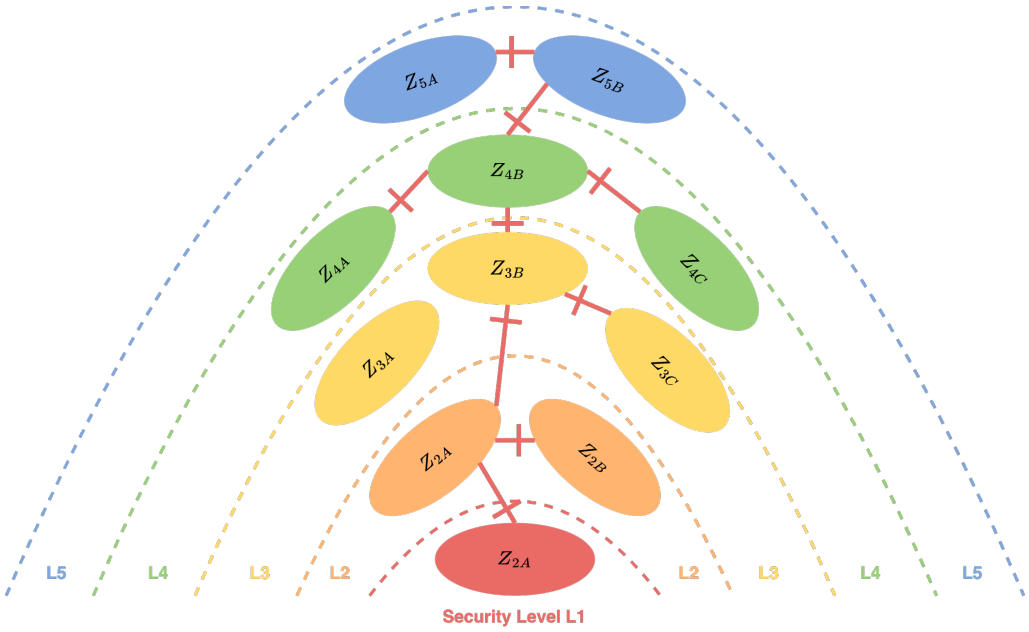


Figure 2.6: Simplified illustration of the IAEA Defensive Computer Security Architecture (DCSA) based on the IAEA NSS-17T (Rev.1) [IAE21] (Graded Approach to Security).

Further improvements in the concept of Defense-in-Depth can be seen in the "Graded Approach to Security" in the Nuclear Cybersecurity Domains as published in the IAEA NSS-17T (Rev.1) [IAE21]. The core idea behind this approach is to define digital assets and assign them into security levels and separate the levels into security zones. Each security level defines a set of security requirements that need to be addressed. The communication between security levels and security zones is strictly restricted and monitored. With higher security levels the requirements regarding security measures rise as well. A reference model illustrating the security level and -zone structure is shown in Figure 2.6. This model incorporates five security levels with well-defined and monitored level transitions (indicated by red lines) and exemplary security zones per level.

2.2.5 Adversarial Behavior Modeling

While the previous section described the architecture of Industrial Control Systems, this section aims at giving an insight on common models for adversarial behavior in Industrial Control Systems.

2.2.5.1 SANS ICS Kill Chain

The SANS ICS Kill Chain [AL15] is an extension of the widely accepted Cyber Security Kill Chain by Lockheed Martin [HCA11]. The major difference to the Cyber Security Kill Chain is a 2-stage design. As described in Section 2.2.3 the Purdue Enterprise Reference Architecture (PERA) differentiates between IT and OT segments. This circumstance is paid respect with this 2-stage design. The aim of the first stage is to gain foothold in the IT compartment of the target system environment, move laterally and gain information and capabilities to pivot into the OT compartment, which is described by the second stage attack. Therefore, the first stage can be seen as a preparatory supporting attack enabling the actual targeted attack on critical assets of the OT. This marks the complexity and longevity of such attacks and emphasizes the need to gather intelligence on components, communications and assets in order to achieve the aimed impact on the systems.

2.2.5.2 The MITRE ATT&CK® Framework

As pointed out in Section 1.2, the MITRE ATT&CK® Framework is a curated knowledge base and description framework for adversarial behavior and their correlating tactics, techniques and procedures (TTPs) [SAM+18], the current version is available at https://attack.mitre.org/.

At the time of writing, MITRE ATT&CK® is composed of three attack matrices covering distinct areas, namely the ATT&CK® Matrix *for Enterprise*, the ATT&CK® Matrix for *Industrial Control Systems* and the ATT&CK® Matrix for *Mobile*. These matrices display adversarial tactics in columns, with techniques and procedures in rows beneath the corresponding columns. The tactics might be interpreted as some form of kill chain. However, unlike the Cyber Kill Chain and the ICS Cyber Kill Chain, an adversary may not use all of the tactics, nor in the order listed in MITRE ATT&CK®.

The ATT&CK® framework started with a description of TTPs for common enterprise IT and later for mobile. In 2021, MITRE released the first version of an attack matrix specific for ICS, called MITRE ATT&CK® for Industrial Control Systems [ABS21] (shortened ATT&CK®ICS), based on the widely accepted and used MITRE ATT&CK® Framework for Enterprise [SAM+18]. To the time of writing, ATT&CK®ICS is under active development, the current version is available at https://attack.mitre.org/matrices/ics/.

The adversarial tactics covered in ATT&CK® *for Enterprise* are listed in Table 2.1.

The purpose of MITRE ATT&CK®ICS is to mirror the actions of adversaries in Industrial Control Systems and provide comprehensive information on tactics, techniques, procedures, malware, common assets, and known adversary groups and Advanced-Persistent-Threats (APT) for this particular domain. The framework

Table 2.1: Overview on listed Tactics of MITRE ATT&CK® for *Enterprise*. Shortened description based on https://attack.mitre.org/tactics/enterprise (2024-01-26). Bold indicates the inclusion of Information-based techniques.

ID	Name	Description
TA0043	Reconnaissance	Gather information to plan future operations.
TA0042	Resource Development	Establish resources to support operations.
TA0001	Initial Access	Get into the network.
TA0002	Execution	Run malicious code.
TA0003	Persistence	Maintain foothold.
TA0004	Privilege Escalation	Gain higher-level permissions.
TA0005	**Defense Evasion**	**Avoid beeing detected.**
TA0006	Credential Access	Steal account names and passwords.
TA0007	Discovery	Figure out the environment.
TA0008	Lateral Movement	Move through the environment.
TA0009	Collection	Gather data of interest.
TA0011	**Command and Control**	**Communicate with compromised systems to control them.**
TA0010	**Exfiltration**	**Steal data.**
TA0040	Impact	Manipulate, interrupt or destroy your systems and data.

is based on publicly available reports, security bulletins, and published research. ATT&CK®ICS includes 12 tactics, 10 of which are the same as the Enterprise matrix (cf. Table 2.1 and Table 2.2). However, the order of tactics varies between the matrices, and the techniques and procedures of the individual tactics are distinct and domain-specific. The tactics are described in more detail in Table 2.2.

In October 2023, ATT&CK®ICS was updated to include a first set of typical assets commonly found in ICS. This list, as well as additional enhancements to the ICS matrix, is still a work in progress and has not yet been fully incorporated into the framework to the time of writing. The main distinction between Enterprise and ICS in terms of tactics is the inclusion of "Inhibit Response Function" and "Impair Process Control" for ATT&CK®ICS. The Enterprise matrix includes Reconnaissance, Resource Development, Credential Access and Exfiltration, but these are not present in the ICS matrix.

Techniques of Information Hiding can be found at several locations in ATT&CK® *for Enterprise*. Figure 2.7 gives an overview in which tactics and techniques Information Hiding-based methods can play a role. Generally, IH-based methods can be mapped to three tactics of ATT&CK® *for Enterprise*:

Defense Evasion `[TA0005]` comprises multiple techniques in which Information Hiding plays a role. These techniques can be generally linked to filesystem and media steganography, e.g. to hide components of malware on an infected host.

Table 2.2: Overview on listed Tactics of MITRE ATT&CK® for *Industrial Control Systems*. Shortened description based on https://attack.mitre.org/tactics/ics/ (2024-01-26). Bold indicates the potential inclusion of Information-based techniques.

ID	Name	Description
TA0108	Initial Access	Get into the ICS environment.
TA0104	Execution	Run code or manipulate system functions, parameters, and data in an unauthorized way.
TA0110	Persistence	Maintain foothold in the ICS environment.
TA0111	Privilege Escalation	Gain higher-level permissions.
TA0103	**Evasion**	**Avoid security defenses.**
TA0102	Discovery	Locating information to assess and identify targets in the environment.
TA0109	Lateral Movement	Move through the ICS environment.
TA0100	Collection	Gather data of interest and domain knowledge on the ICS environment.
TA0101	**Command and Control**	**Communicate with and control compromised systems, controllers, and platforms with access to the ICS environment.**
TA0107	Inhibit Response Function	Prevent safety, protection, quality assurance, and operator intervention functions from responding to a failure, hazard, or unsafe state.
TA0106	Impair Process Control	Manipulate, disable or damage physical control processes.
TA0105	Impact	Manipulate, interrupt, or destroy ICS systems, data, and their surrounding environment.

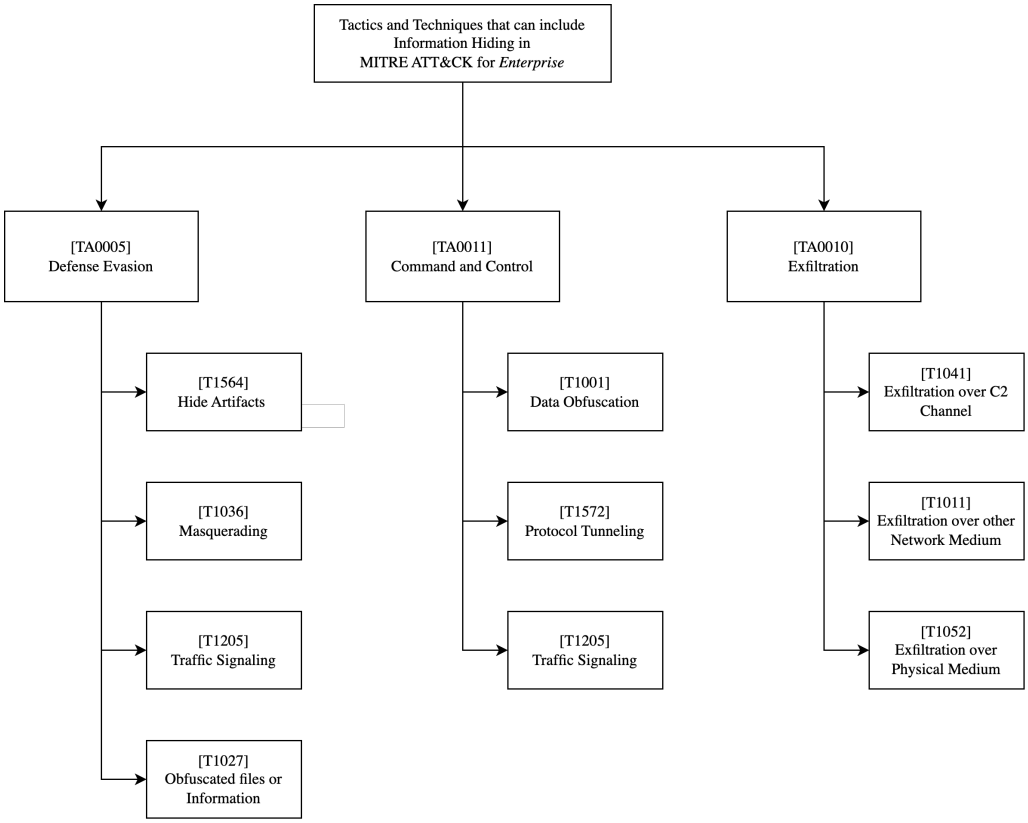


Figure 2.7: Tactics and Techniques of MITRE ATT&CK® *for Enterprise* that might make use of Information Hiding methods.

Command and Control `[TA0011]` Under the umbrella term data obfuscation `[T1001]` as well as Protocol Tunneling `[T1572]` and Traffic Signaling `[T1205]`, common methods of Information Hiding and Covert channels can be linked to those techniques.

Exfiltration `[TA0010]` makes mainly use of the previously described Command and Control channels. However, Exfiltration also includes other network media `[T1011]` and physical media `[T1052]` which resembles without explicitly mentioning Out-of-Band Covert Channels.

As illustrated in Figure 2.8, in case of MITRE ATT&CK® for *Industrial Control Systems*, Information Hiding-based techniques are only reflected in `[TA0103]` *Evasion*. Again, similar to the enterprise version, these techniques can be mainly linked to methods of filesystem and media steganography.

At the time of writing[2], Covert Channels in the communication of ICS or CPS-based Covert Channels are not mentioned or covered by MITRE ATT&CK® for *Industrial Control Systems*.

Such techniques should be expected in `[TA0101]` *Command and Control*. Arguably, two techniques of *Command-and-Control* could be extended to incorporate covert

[2] as of 2024/02/20

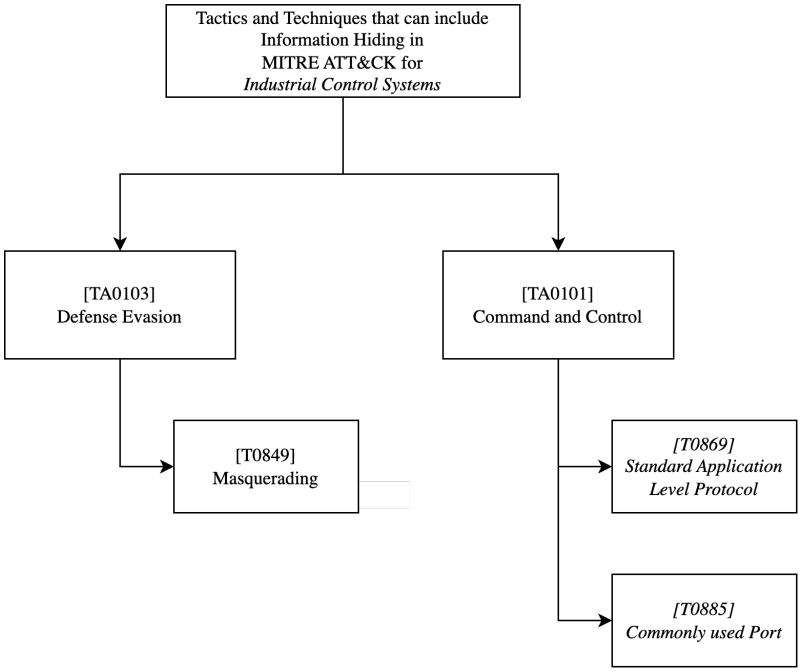


Figure 2.8: Tactics and Techniques of MITRE ATT&CK® *for Industrial Control Systems* that might make use of Information Hiding methods.

channels: `[T0869]` *Standard Application Level Protocol* and `[T0885]` *Commonly used Port* (cf. Table 2.2 and Figure 2.8).

Section 5.1.12 discusses such an extension in more detail based on the results of the research conducted for this thesis.

2.3 Summary of required Fundamentals

The pattern-based Network Information Hiding Taxonomy by Wendzel et al. [WZFH15] is used in its' 2018 extended form [MWC18] in two ways in this thesis. On the one hand, the taxonomy is used when describing and comparing network covert channels in the context of Industrial Control Systems i.e., using the taxonomy to classify the findings of thesis. On the other hand, the taxonomy is used in Chapter 4 as a method to reveal potential covert channels in the analysis of network protocols commonly found in Industrial Control Systems e.g., in the analysis of Modbus/TCP in Section 4.1 (also published in [LD20]).

The ICS Security Guidelines and Recommendations of Section 2.2.4, namely the NIST SP800-82 [SPL+15], the SANS Secure Architecture for Industrial Control Systems [Obr15] and the ICS-CERT Recommended Secure Network Architecture [HS16] are used in Section 3.1 to derive a plausible reference architecture representing the security-aware state of art of ICS architectures.

The SANS ICS Kill Chain [AL15] and MITRE ATT&CK®ICS [ABS21] are used in Chapter 3 in the description of threat scenarios and to highlight the use-cases of covert channel in the context of these attack frameworks, i.e. to show where covert channels might be useful in adversarial behavior.

The countermeasure model by Mazurczyk et al. [MWZ+16], described in Section 2.1.10, is used in Section 5.2 to classify potential mitigation methods derived from the results of the covert channel analysis in Chapter 4.

The definitions and differentiation of Covert Channels, Side Channels and Out-of-Band Covert Channels is used to derive a classification of covert channels in the context of Industrial Control Systems as well as to put them into perspective in their use in adversarial behaviour as well as in the derivation of potential countermeasures. This is due to the fact that these different types of information hiding methods require according sets of mitigation methods.

The definitions of Cyber Physical Systems in Section 2.2.1 and the Purdue Enterprise Reference Architecture (PERA) (Section 2.2.3) are used to identify potential participants in hidden communication and to identify and classify potential carrier for hidden information.

The summary of selected publications connected to Cyber-Physical System Steganography is used together with the performed case studies in Chapter 4 to derive a classification of Information-Hiding methods in Industrial Control Systems.

3. Information Hiding in Industrial Control Systems

This chapter puts the fundamentals of Chapter 2 into perspective and provides a generalized view of Information Hiding in the context of Industrial Control Systems. Initially, a plausible reference architecture is derived from the State-of-Art in Section 2.2 to get an understanding of the overall system architecture and to derive potential covert sender/receiver and plausible carriers. Next in Section 3.3, Information Hiding methods in CPS are categorized based on the selected publications of Section 2.1.9. With this classification in mind, Section 3.5 follows with a brief description of assumptions and concepts of Information Hiding applied to the specifics of ICS, which is required for the understanding of the case studies that follow in Chapter 4. Section 3.6 concludes this chapter with a classification of adversarial use cases of covert channels in Industrial Control Systems.

3.1 Plausible Reference Architecture derived from the State-of-Art

To analyze the potential of covert channels in Industrial Control Systems it is important to have a clear picture of how these systems are constructed, which components are likely to be found, and how they are interconnected.

For this thesis, a generalized plausible reference architecture is derived from the State-of-Art as described in Section 2.2. In the following, this generalized architecture is named *PlauRA-1*. For deeper analysis, two variants are used in this thesis: *PlauRA-2*, illustrated in Figure 3.2, represents a more detailed view on the Operational Technology part with two Cell/Area Zones. In Section 3.6 covert adversarial movement is illustrated and categorized along with a security-improved version of PlauRA-1, called *PlauRA-3*.

In the following *PlauRA-1* and *PlauRA-2* are described, *PlauRA-3* later in Section 3.6.

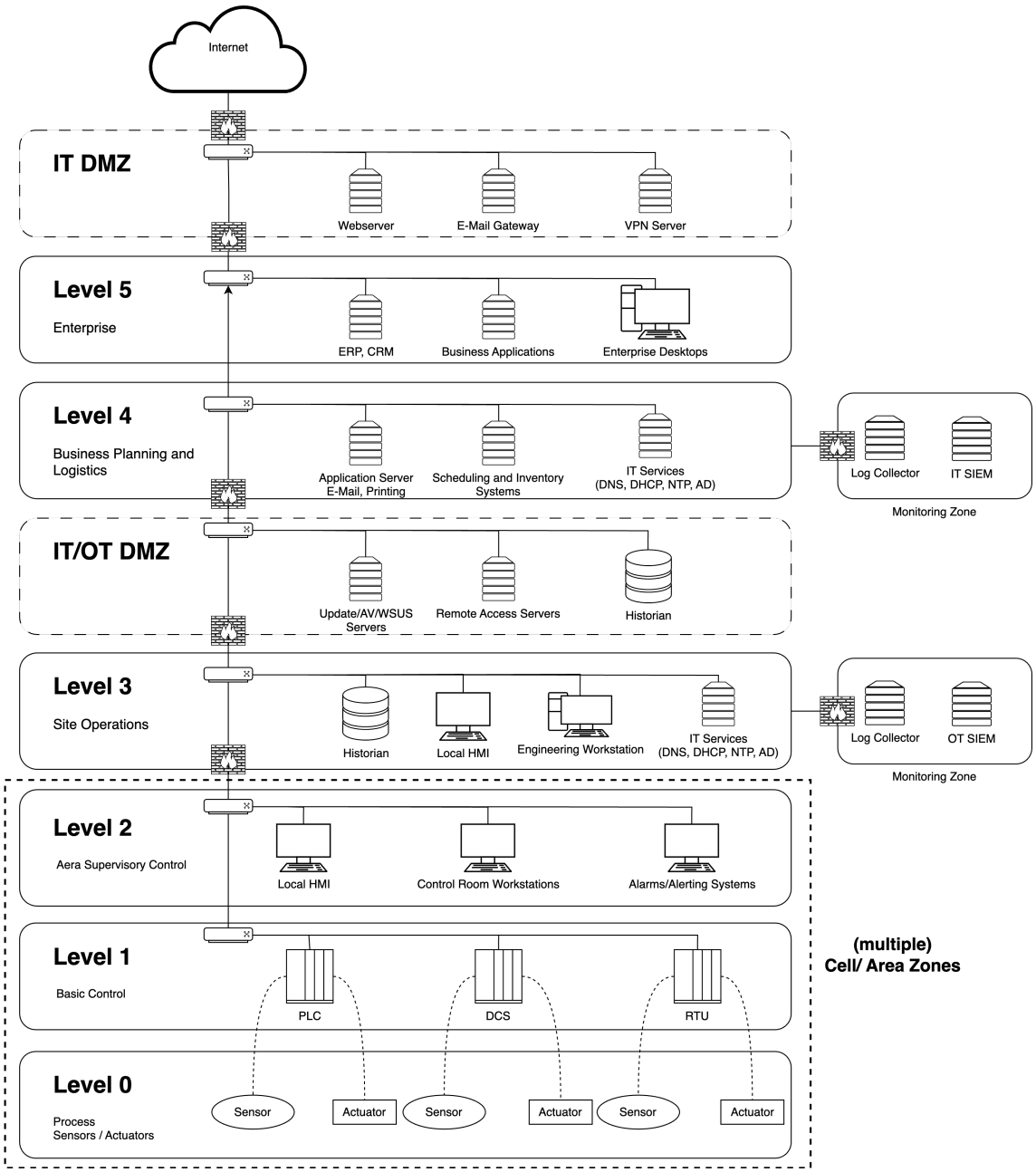


Figure 3.1: **PlauRA-1:** Plausible Reference ICS Architecture derived from NIST SP 800-82 [SPL+15], SANS Secure Architecture for Industrial Control Systems [Obr15] and ICS-CERT Recommended Secure Network Architecture [HS16].

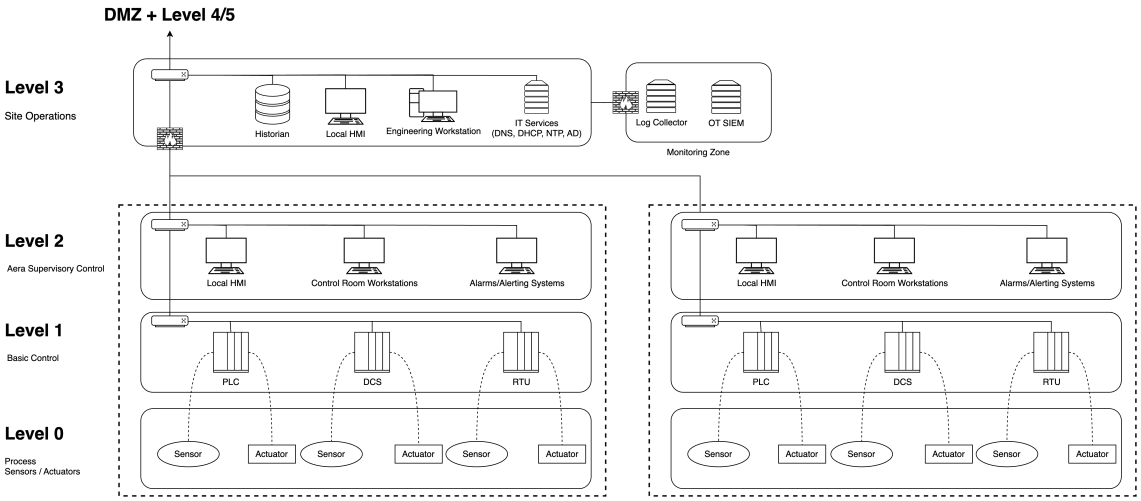


Figure 3.2: **PlauRA-2**: Detailed view on the Operational Technology part with two Cell/Area Zones as part of the plausible reference architecture based on NIST SP 800-82 [SPL+15], SANS Secure Architecture for Industrial Control Systems [Obr15] and ICS-CERT Recommended Secure Network Architecture [HS16].

The plausible reference architecture represents an exemplary facility that can plausibly cover many application and threat scenarios. As ICS can be very diverse and purpose-built, it is important to note, that the structure and components can differ significantly between systems. Still, this generalized approach provides a structured foundation apt for discussion of covert channels-based threat scenarios. As it is the common ground and de-facto standard, when describing ICS environments, here the Purdue Enterprise Reference Architecture (PERA) [Wil92], described in Section 2.2.3, builds the foundation for the plausible reference architecture as well.

Based on the Purdue Enterprise Reference Architecture six functional levels are defined, from Level 0 which comprises the process control using sensors and actuators to Level 5 which comprises all enterprise-level applications. From summarizing [Wil92, Alt20, Cis11] following levels and corresponding exemplary components are derived to describe the plausible reference architecture:

- **Level 0 - Process** Sensors and Actuators
- **Level 1 - Basic Control** Controllers, e.g. PLCs or RTUs interfacing with the Level 0 devices
- **Level 2 - Area Supervisory Control** Supervision and Operation, e.g. Local Human-Machine-Interfaces, Alerting Systems
- **Level 3 - Site Level** Plant-wide Operation and Control, common components are: Plant Historian, Engineering Workstations, Control Room HMIs, and network services, like Active Directory (AD), Dynamic Host Configuration Protocol (DHCP), Dynamic Naming Services (DNS), Time Protocol (NTP)
- **Level 4 - Site Business Planning and Logistics** Basic business administration tasks based on standard IT services e.g., scheduling systems, manufacturing

execution systems (MES), and local IT services, for example, email and printing.

- **Level 5 - Enterprise** Corporate-level applications (for example, ERP, CRM, document management) and services (Internet access, VPN entry)

This methodology gives the reference architecture a (vertical) differentiation of functional levels. However, the Purdue Enterprise Reference Architecture was designed only to describe the *functional* difference of these levels and not to design a system to be secure. Therefore, in practice, these components should not be connected directly; instead, some security features should be expected to be in place.

Therefore, common ICS security guidelines and recommendations, namely the defense-in-depth architecture of NIST SP 800-82 [SPL+15] and the SANS Secure Architecture [Obr15] are used to derive a plausible security-focused reference architecture for the analysis of the potential for covert channel-based threats. Both models provide a communication-based view and introduce several security-related additions. The major contribution of NIST SP 800-82 is the introduction of a demilitarized zone (DMZ) between the corporate network (IT) and the control network (OT). The SANS Security Architecture takes this concept further and provides a concept on how to integrate log collector, Intrusion-Detection-Systems (IDS), and Security Information and Event Management systems (SIEM) for IT and OT compartments.

The resulting reference architecture, *PlauRA-1* is illustrated in Figure 3.1. PlauRA-1 is the foundation for further derivations.

From the ICS-CERT Recommended Secure Network Architecture, the separation of Level 1 and Level 2 systems in different area/cell zones is adapted for the reference architecture as well, which further increases security and at the same time allows for better modeling horizontal adversarial movement. These multiple area/cell zones are illustrated as *PlauRA-2* in Figure 3.2 with two zones on Level 1 and Level 2.

3.2 Derivation of Potential Covert Sender/Receiver & Carrier

The Plausible Reference Architecture (PlauRA) from the previous section already provides a first insight on which components and carriers can be expected to be found in Industrial Control Systems. This section goes into more detail about these components and how they might be relevant for Information Hiding methods.

To do so, at first, the definitions of Cyber-Physical Systems and ICS are revisited, to systematically identify and discuss components, potential carriers and sender/receiver for hidden information:

In the NIST SP 800-82 [SPL+15], it is described that a typical ICS *"[...] contains numerous* ***control loops***, ***human interfaces***, *and* ***remote diagnostics and maintenance tools*** *built using an* ***array of network protocols on layered network architectures***. *A* ***control loop*** *utilizes* ***sensors, actuators, and controllers*** *(e.g., PLCs) to manipulate some* ***controlled process***".

From this definition, the most important components can be identified. In the following, their role regarding Information Hiding is described as well:

Controlled Process Physical process, controlled by a *control loop*. The physical process can represent a potential carrier, e.g. when certain properties are modulated to encode a hidden message (for example temperature).

Control Loop Logical system for controlling the *process* using *sensors*, *actuators* and *controllers*.

Controllers Control of the *physical process* using a *control loop* with measurements obtained from *sensors* and manipulation of the process using connected *actuators*. Changes in the control logic can be a way to act as a covert sender as well as in its communications.

Sensors measure (physical) properties of the *controlled process*. Potential receiver, when information is encoded in measurable (physical) properties. Potential sender, when communicating sensor data.

Actuators manipulate (properties of) the *controlled process.* Potential sender of hidden information, e.g. by influencing physical properties.

Human-Interfaces or Human-Machine-Interfaces (HMIs) allow user interaction with *controllers* or *the control loop*, respectively. Can be covert receiver (e.g. by certain events or data) or sender (e.g., by sending commands to controllers).

Network Protocols are used in *layered network architectures* for *communication* between components. Can be carrier for hidden information, either in protocol-specific ways or within the payload or in timing.

Network Components Infrastructure devices and services that are required for the system to work properly. This includes devices like switches, firewalls et cetera but also IT-services, for example DNS(-servers), NTP(-servers) etc. Devices,

services and servers can be covert sender and receiver. Communication between networked components can be a carrier for hidden information.

In summary, it can be stated that controllers, sensors, actuators, and HMIs can act as covert senders and/or receivers, leveraging features/properties of the control loop and the controlled physical process as carriers for hidden information. Any form of communication between components and any data on the physical process (process data) are also subject to potential covert channels. Another important aspect are components and communication in the infrastructure, which is required for the control loop to be operable. This can be functional infrastructure, e.g., network devices and infrastructure services and servers like domain resolution, time synchronization but also security infrastructure, like IDS/IPS systems, SIEM, and devices like firewalls and data diodes.

In the following, this list of potential covert senders, receivers, and carriers is used in conjunction with the Plausible Reference Architecture (PlauRA) and selected publications on Information in CPS to derive a generalized categorization of Information Hiding Methods in Cyber-Physical Systems.

3.3 Proposal for a Categorization of Information Hiding Methods in Cyber-Physical Systems

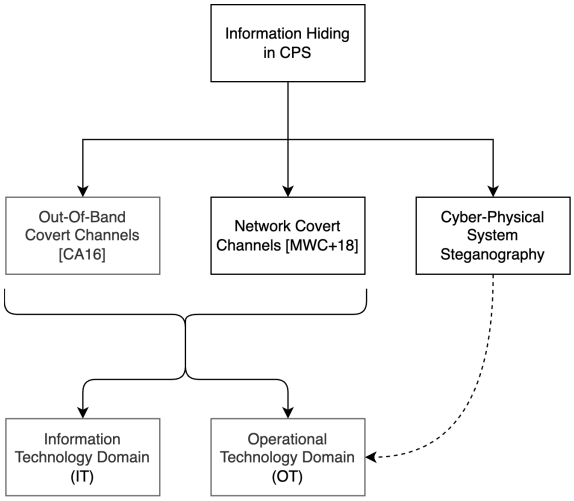


Figure 3.3: Categorisation of Information Hiding Methods in Cyber-Physical Systems. (full view)

As pointed out in Section 2.1.9, a classification of Information-Hiding based techniques specific to Cyber-Physical Systems seems missing. The network pattern taxonomy [WZFH15] provides patterns only for Network Steganography, and the revised pattern taxonomy [WCM+21] represents a domain-agnostic approach. One exception can be seen in [ABP+19], in which Alcaraz et al. distinguish between Timing-, Storage- and Behavior-based Covert Channels in CPS. However, these timing- and storage covert channels only cover (OT-) network covert channels, yet some of the covert channels described in Section 2.1.9 do not fit this classification (e.g., [HK19a, HK19b]). Therefore, the aim here is to derive a classification that fits the current state of the art. Such a classification is critical for a proper understanding of the potential threat imposed by adversaries leveraging Information Hiding-based techniques. Furthermore, such classification of methods aids in the design of targeted countermeasures. In the following, a first classification is proposed based on the previously discussed architectures, potential carriers, and selected publications from the State-of-Art.

Figure 3.3 illustrates three major domains to be found in CPS. Figure 3.6 shows all parts of the classification in one single graph to give an overview.

The first class that can be derived from the state of the art, is **Network Steganography** (or **Network Covert Channels**) applied to protocols found in Cyber-Physical Systems. In such systems, both, common IT protocols as well as protocols specific for Operational Technology (OT) can be found. Therefore, as shown in Figure 3.3, such network covert channels can be further distinguished whether applied to a general purpose (IT) protocol or OT-specific protocol.

To classify such network covert channels, the widely accepted taxonomy of Wendzel et al. [WZFH15] is used in its 2018 extended form [MWC18] (see Section 2.1.8).

The extended pattern-based taxonomy of Mazurczyk et al. (2018) [MWC18], originally introduced by Wendzel et al. (2015) [WZFH15], is an extensive classification of methods to establish network covert channels. The pattern-based taxonomy is a unified approach for modeling and describing covert channels as part of information hiding in network communications. It is based on the analysis of over one hundred techniques and summarizes them into 18 general patterns. In the extended taxonomy, there are eight timing patterns and 10 storage patterns. With the pattern-based taxonomy, covert channels in IT and OT protocols can be described in a unified language, by mapping the corresponding method to one of the 18 patterns.

The next class of Information Hiding methods in Cyber-Physical Systems are **Out-Of-Band Covert Channels** (Section 2.1.6). They form a completely different class since they do not rely on existing communication, but rather open new (difficult to intercept) communication channels. This makes them very different from the other two classes. In the literature, the term *side channel* (unintentional information leakage) is often used synonymously. However, Out-of-Band (OOB) covert channels are *intentionally* leaking information to a receiver. Common examples are the use of LED status indicators (e.g. on switches [Gur18b], hard drives [GZE17] or PLCs) to optically encode hidden messages using morse code. The main purpose of such covert channels is to establish a hidden communication channel between devices that are actually not allowed to communicate. This is often the case in scenarios where air gaps (physical segregation) are used. In [GZE20], Guri et al. even demonstrated data exfiltration from faraday-caged, air-gapped computers via magnet fields. In [CA15] and [CA16] Carrara and Adams provide a comprehensive classification for such techniques. Similar to the pattern-based taxonomy for network covert channels, Carrara and Adams derive the classification from a comprehensive survey on the state of the art. From their survey Carrara and Adams derive six sub-types of Out-of-Band Covert Channels: (1) Acoustic, (2) Light, (3) Vibration, (4) Magnetic, (5) Temperature, (6) Radio-frequency.

The third and last class are **Cyber-Physical Covert Channels**, i.e., those methods that use features that are specific (and only) to Cyber-Physical Systems and do not fit one of the other two classes (Network- and OOB covert channels).

In the following, a classification for these CPS-specific covert channels, derived from the State-of-Art, is described in more detail.

3.4 Derived Classification of Cyber-Physical System specific Covert Channels

Based on the considerations of the previous sections and selected publications on Information Hiding methods in CPS, the following classification is proposed. These classes may not be non-overlapping and some covert channels might use a combination of multiple techniques to establish a hidden communication channel.

Figure 3.4 illustrates three major categories of Cyber-Physical System-specific IH-methods:

Property-based P_i Methods inspired by [HK19a, HK19b]

Behaviour-based B_i Methods derived from [ABP+19]

Cache-based Ca_i Methods inspired by [WMH17a]

In the following, each of these categories is explained in more detail.

3.4.1 Property-based Methods P_i

Property-based covert channels are further differentiated in P_1 ***Physical Property Modulation*** and P_2 ***Feedback Modulation***.

In the case of P_1 *Physical Property Modulation* one or multiple physical properties that are measured by the CPS are physically modified, for example, by increasing the temperature of a measured feature (e.g., a fluid). In contrast to that, P_2 *Feedback Modulation* covert channels do not actually change the physical property, instead, they only modify the feedback from sensors readings or actuator states (without actually performing a physical change).

3.4.1.1 P_1 Physical Property Modulation

Physical Property Modulation schemes modulate a physical property that is monitored by a controller to encode hidden information. One example for *Physical Property Modulation* would be the following, simple scenario (based on Case Study CS_2 and [LNK+21]): Controller C_1 measures the water level of a water tank. Controller C_2 can open or close a valve (actuator) to control the water flow into the water tank. By opening and closing the valve accordingly Controller C_2 can send secret information towards Controller C_1 by increasing the water level in the water tank in specific ways.

Another example is the approach proposed by Herzberg and Kfir [HK19b] in which an actuator modulates the time it takes to change its state (in that case opening/-closing a valve). In the provided scenario, a controller has the target of keeping the pH level of a water tank in a certain range. This is a typical scenario in water treatment processes. In this example, the controller has the set point of 7 pH, with a lower threshold of 6 pH and an upper threshold of 8 pH. Once the pH level reaches the lower threshold, the controller will engage the actuator to open a valve letting a pH-increasing fluid flow into the water tank, till the pH level reaches the upper

threshold, and the controller gives the command to the actuator to close the valve. The speed of the actuator when closing and opening the valve will determine how quickly or slowly the pH level changes. This transition can be monitored by a PLC that uses a sensor to measure the pH level. By observing the transition, the PLC can determine if the actuator is operating quickly or slowly, thus retrieving the covert message.

3.4.1.2 P_2 Feedback Modulation

One example for P_{2_1} *Sensor Measurement Feedback Modulation* as part of *Feedback Modulation* is the modulation of sensor measurements by a PLC as described later in the Case Study CS_2 and published in [LNK+21]. In that case, sensor readings are tampered with while the physical property remains untouched. Similarly, an actuator can give feedback to the controller to have performed a certain action without actually doing it, e.g., closing a valve (P_{2_2} *Actuator State Feedback Modulation*).

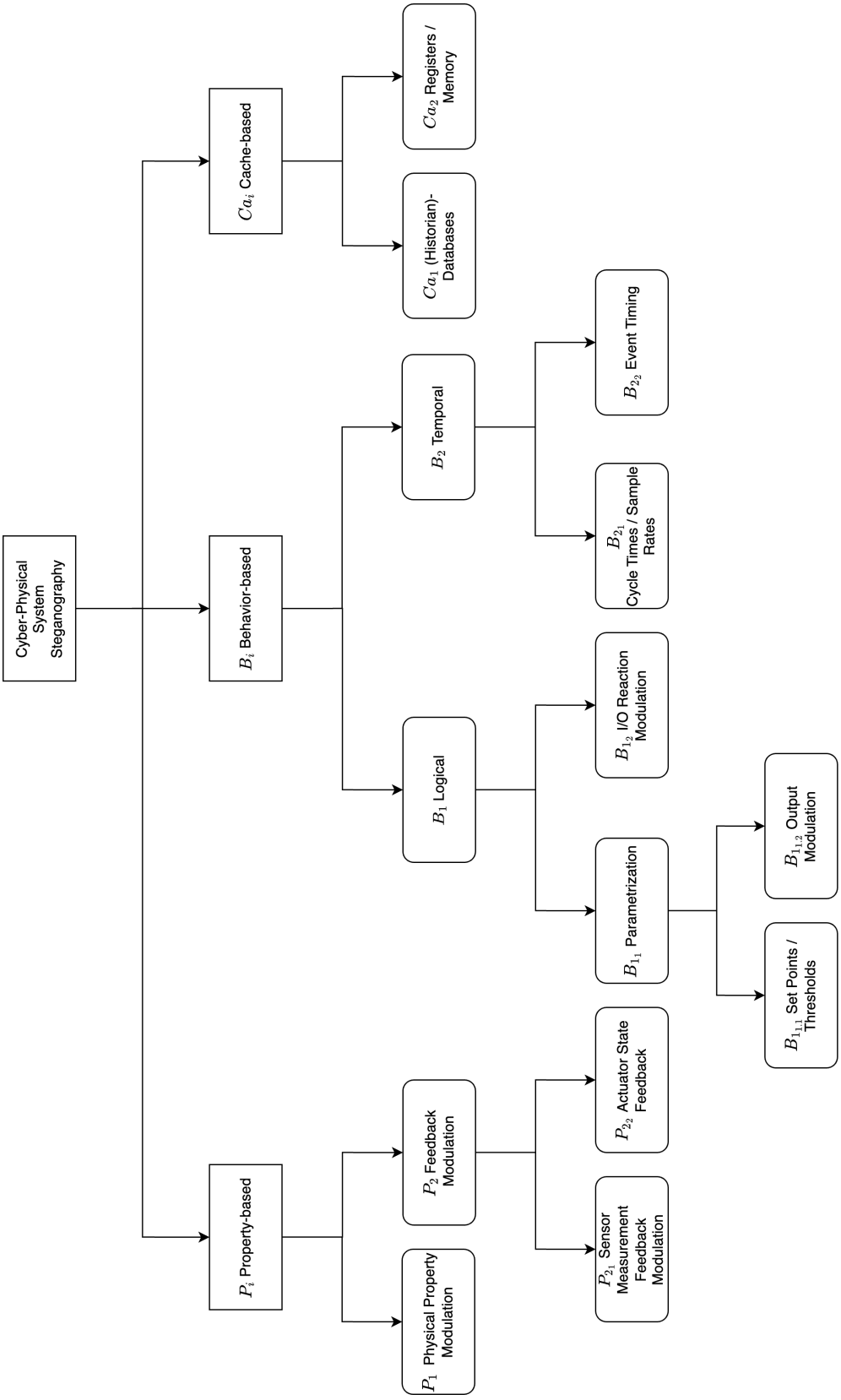


Figure 3.4: Derived classification of methods in the domain of Cyber-Physical System Steganography. (detailed view)

3.4.2 Behavior-based Covert Channels B_i

Alcaraz et al. describe *Behaviour-based Covert Channels* in [ABP+19] as covert channels that *"operate by intentionally changing the behavior of an application"*. While the authors do not further distinguish such covert channels, here in this classification, the original definition of Alcaraz et al. is further diversified into *logical* changes to the behavior and *temporal* changes to behavior.

3.4.2.1 B_1 Logical Behavior Modulation

Logical behavior modulation make changes to the logic of the control loop to encode hidden information. Here, *Logical* behavior modulation is further distinguished between B_{1_1} *Parametrization* and B_{1_2} *I/O Reaction Modulation (Reaction in certain conditions)*.

B_{1_1} Parametrization Covert Channels

Parametrization Covert Channels make slight changes to the parametrization of the control loop, e.g., changing set points, thresholds or modulating outputs of acutators. Here, they are further differentiated between covert channels leveraging set points or thresholds to modulate the behavior and output modulation. The idea of modulating set points or thresholds is to change when the controller takes action, e.g., to heat a fluid that is cooling to far down. *Output Modulation* in contrast, changes the actions of actuators, e.g. by modulating the RPM (speed) of a motor (actuator).

B_{1_2} I/O Reaction Modulation

I/O Reaction Modulation (Reaction in certain conditions) covert channels change the usual reaction of a controller in certain conditions. For example, a controller responsible for keeping the temperature of a water tank within a specific range might usually regulate the temperature by turning off a corresponding (heating) actuator and wait to let the water cool down. To encode a hidden message, the logic of the controller could be altered, so that the controller might leave the actuator (heater) turned on, even though the temperature is rising, but to cool the water at the same time by letting fresh, cold water flow into the water tank by opening a valve (actuator). This represents a different reaction to specific conditions and might be used to encode a message (e.g., encoding a binary zero when using the common reaction and encoding a binary one when using the alternative reaction when reaching the condition).

3.4.2.2 B_2 Temporal Behavior Modulation

The other branch of *behavior-based* covert channels modulates *temporal* aspects of CPS behavior. Two identified aspects are *cycle times/sample rates* (B_{2_1}) and *event timing* (B_{2_2}).

B_{2_1} Cycle Times/Sample Rate Modulation

Code execution in OT controllers like PLCs and RTUs is *cyclic*, i.e., the code is structured to be run in cycles. The resulting time to run one such cycle (basically the code execution time) is the so-called cycle time and resembles an important metric that is closely monitored. Cycle times are generally static with only minor fluctuations. This circumstance can be leveraged to encode hidden information within deviations from the the usual cycle times, e.g., using specifically crafted function blocks in the programming of the PLC to delay the cycle times. These deviations in cycle times can be monitored from outside to retrieve the hidden message. Another example is the modulation of sample rates of sensors, i.e., to change the time span between measurements. Figure 3.5 exemplary illustrates the impact of such modified sample-rates in a simulation. This example uses a sawtooth wave function with a linear increase from -2.0 to 1.6 from where it drops to -2.0 again. In the upper graph from Figure 3.5 a sample rate of 1000ms is used, in the lower graph the sample-rate is increased to 1333ms. As depicted in the figure, this leads to a shift of every second (local) minimum from -2.0 to only -1.6. This is due to fact that the sample rate is too slow to represent the full range of values provided by the simulation. Such differences in local minima can be used to encode hidden information which then can be retrieved at different locations, e.g., an Human-Machine-Interface or Historians (for options of retrieval cf. Case Study CS_2 in Chapter 4 and [LNK+21]).

B_{2_2} Event Timing

Event timing methods leverage the circumstance that some events in CPS systems occur at specific points of time. For example, in a smart building the system might close all windows in the evening, every day at the same point of time (cf. [Wen12a]). By shifting this event slightly (e.g., one minute before/after) hidden information can be encoded.

Another example is the approach by Herzberg and Kfir [HK19a] in which the authors modify sensor values to modulate the timing when a certain threshold is reached.

3.4.3 Cache-based Methods Ca_i

Cache-based[1] methods, in the sense of secret storage, are inspired by a covert channel published by Wendzel et al. in [WMH17a] and [WMH17b], respectively. The authors propose to use unused registers of temperature sensors to hide information to be retrieved at a later point, i.e., to use these memory spaces as a secret cache in the sense of a dead drop. Wendzel et al. use an analogy from nature by using hoarding and caching tactics found in animal behavior to describe how attackers could potentially use Information Hiding to establish secret *steganographic data storage* in smart buildings. Their focus lies on actuators and unused registers of automation systems in smart buildings, which are used as secret storage to hide information. This idea of distributing a hidden message in multiple covers is known as *scattered hoarding*. In their work, they focus on *steganographic data storage* in Cyber Physical Systems by using a smart building automation system (BAS) as a *secret storage*.

[1] Cache in the sense of a secret storage or dead drop, not system cache.

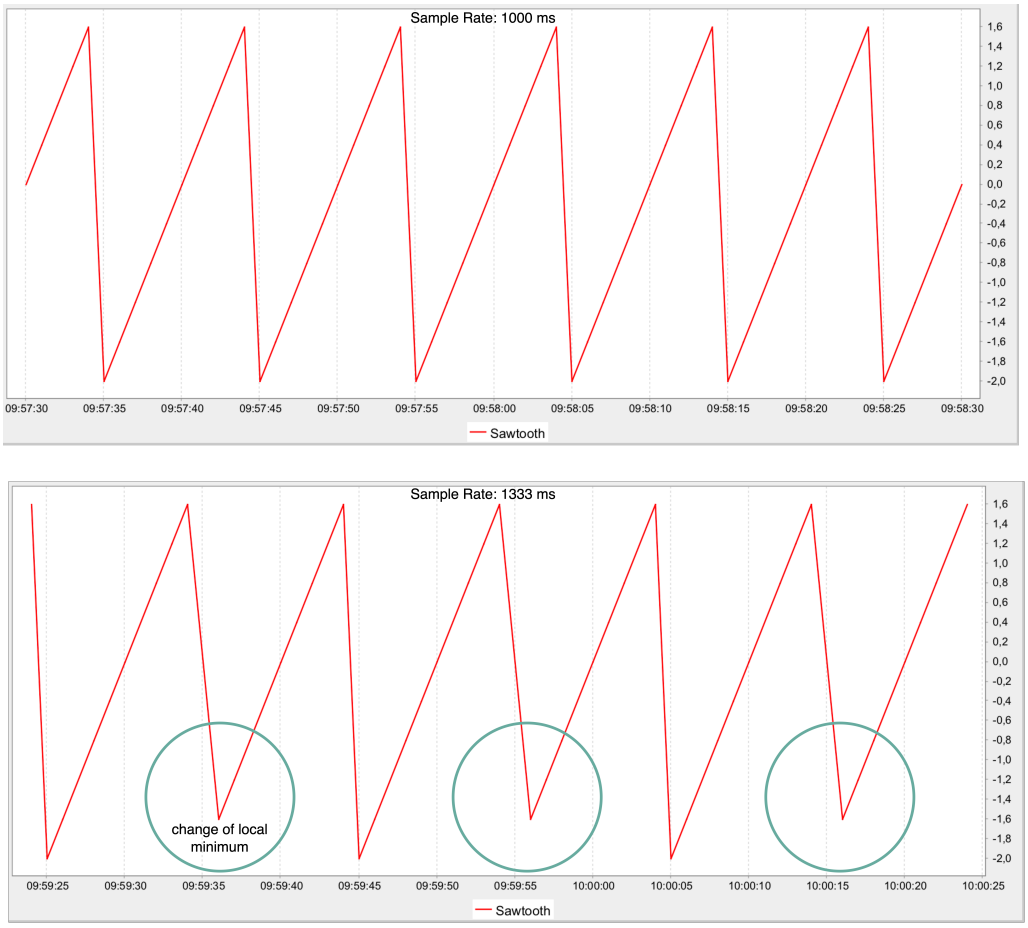


Figure 3.5: Visual example of the impact on sensors values in case of sample rate modulation (B_{2_1} Cycle Times/Sample Rate Modulation)

This is framed by a scenario in which two spies want to secretly exchange information in an asynchronous way by using a *dead drop*, which in this case is the smart building. Since ICS typically deploy long-term storage of historical process data, in Case Study CS_2 in Chapter 4 (based on our publication [LNK+21]), this fact is leveraged in encoding hidden messages by manipulating sensor or actuator values and retrieving those by accessing long-term storage. Based on these two approaches, this classification further distinguishes *Cache-based* methods in Ca_1 *Register/Memory* based techniques (cf. [WMH17a, WMH17b]) and Ca_2 methods using (Historian-) Databases as secret storage (cf. [LNK+21]).

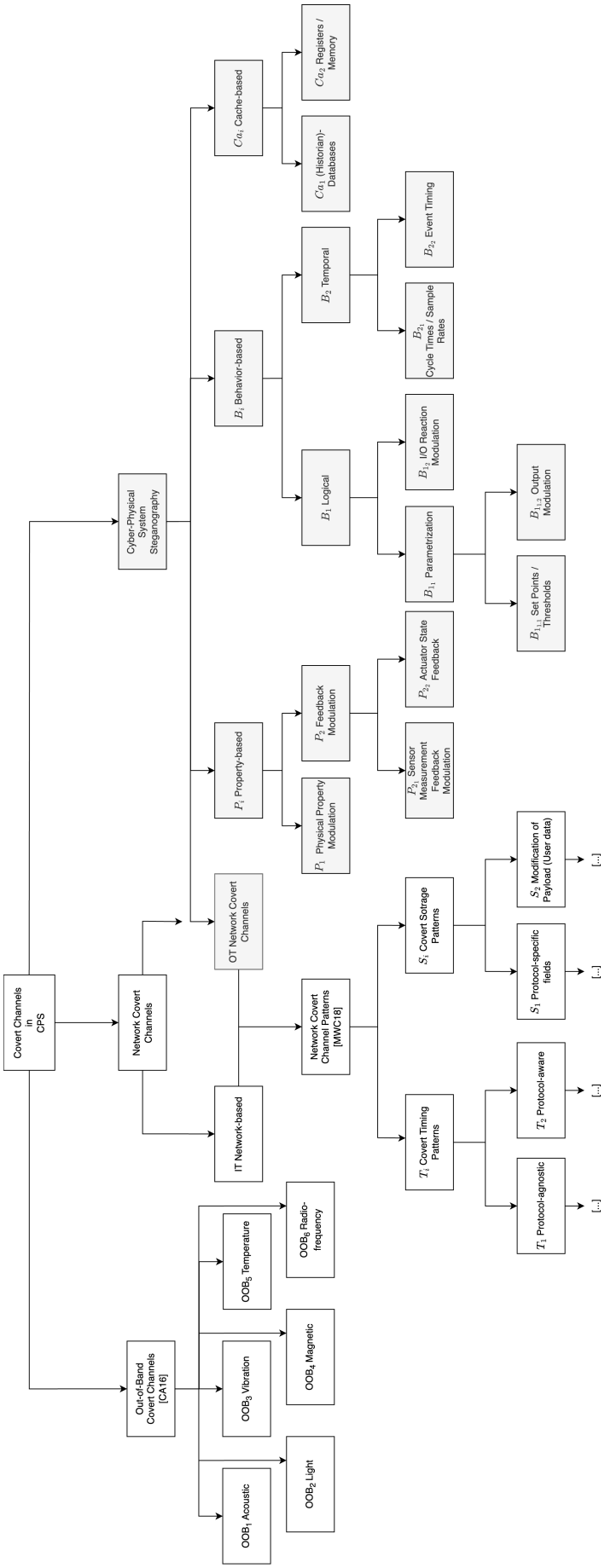


Figure 3.6: Full view of the categorization of Information Hiding Methods in Cyber-Physical Systems.

3.4.4 Limitations of the Derived Classification

This classification is based on selected publications and by that represents a classification for a selected current state of art in terms of covert channels in CPS environments. However, as the state of art evolves, this classification has to evolve as well. Novel covert channels might require major revisions to the proposed classification. Still, the aim is to give the the state of art more structure and beeing able to compare covert channels and put them into perspective. Variations of the proposed classification are plausible as well. For example, the subclass of behavior-based methods (originally introduced to CPS by Alcaraz et al. [ABP+19]), could arguably also be seen as a combination or hybrid form of the classical timing and storage distinction. When describing behavior, it always has a notion of a temporal component, since any action/movement requires time; therefore, behavior cannot be described at a specific point in time rather than over a specific time span. For example, the methods described in Section 3.4.2.1 (Logical Behavior Modulation) also (unintentionally) lead to changes when, which, event occurs (modification to temporal behavior). However, the decisive factor for the classification is **where** the hidden information encoded to be retrieved - if it is encoded in a change of a point in time or a time span it is considered a *temporal method*, if is a semantic change it is considered a *logical method*.

This problem of potential confusion of methods to embed secret information and where it is actually encoded and to be retrieved, was one major motivation for the ongoing work of establishing a generic, domain-agnostic taxonomy for information hiding methods that differentiates between embedding patterns and retrieval patterns. A preprint of this work is available [WCM+22].

In general, the classes are not non-overlapping and some covert channels can be described by a combination of techniques from the classification. This can be seen as a limitation but it also provides some advantages. One example is the scenario where secret information is embedded into network packets containing sensor values. For example, the information is encoded in the LSB of the sensor values. On the one hand, the embedding process is a network covert storage channel in terms of the Network Covert Channels Patterns [MWC18]. On the other hand, if the covert receiver retrieves the hidden data from the sensor values stored in the (historian-) database, it can also be considered to be a Cache-based method in the proposed classification. However, by fitting into two categories, there are also two potential ways of detecting this covert channel. A warden could detect this covert channel on the network level i.e., in transmission or later on in the (historian-) database i.e., in storage.

In summary, this classifcation represents a first step towards a systematic categorization of Information Hiding methods specific to CPS.

3.5 Methods of Steganography in ICS

This section provides the required foundation for the description and analysis of the following case studies in Chapter 4. In the following, the underlying hidden communications model, assumptions on embedding, retrieval and key distribution, participants in hidden communication and relevant metrics are described.

3.5.1 Hidden Communications Model

As described in Section 2.1.5, the foundation of covert channel research is G.J. Simmons *Prisoner's Problem*, which defines the hidden communication model and its participants. In its original form [Sim84], Alice and Bob want to secretly form an escape plan. They are allowed to communicate, however, the warden (often called *Wendy*) analyzes the messages for any sign of forbidden communication.

For the processes of embedding, retrieval and key distribution, some assumptions have to be made in order to have a uniform model for the evaluation of covert channels, the corresponding threat scenarios and countermeasures. The basic communication model is illustrated in Figure 3.7 and is based on the hidden communication model used to describe storage channels in Network Information Hiding [MWZ+16]. It is used for the entire analysis if not other indicated.

The model assumes that between Alice (Covert Sender) and Bob (Covert Receiver) some form of overt communication takes place and that Alice and Bob **share a secret key**, noted as K_{shared}. This overt communication is used as a **carrier** for hidden messages and is considered to be the **cover channel**. Within this cover channel, **cover objects** are transferred from Alice to Bob. In Network Information Hiding, these are network packets or flows (series of related packets). As seen in the classification of IH-methods in CPS, in case of CPS Steganography types of cover objects can be manifold. Common examples are measurable features of physical properties e.g., the temperature or the readings of a sensor. This communication model is generalized in such a way, that it fits all carriers previously mentioned in the classifcation.

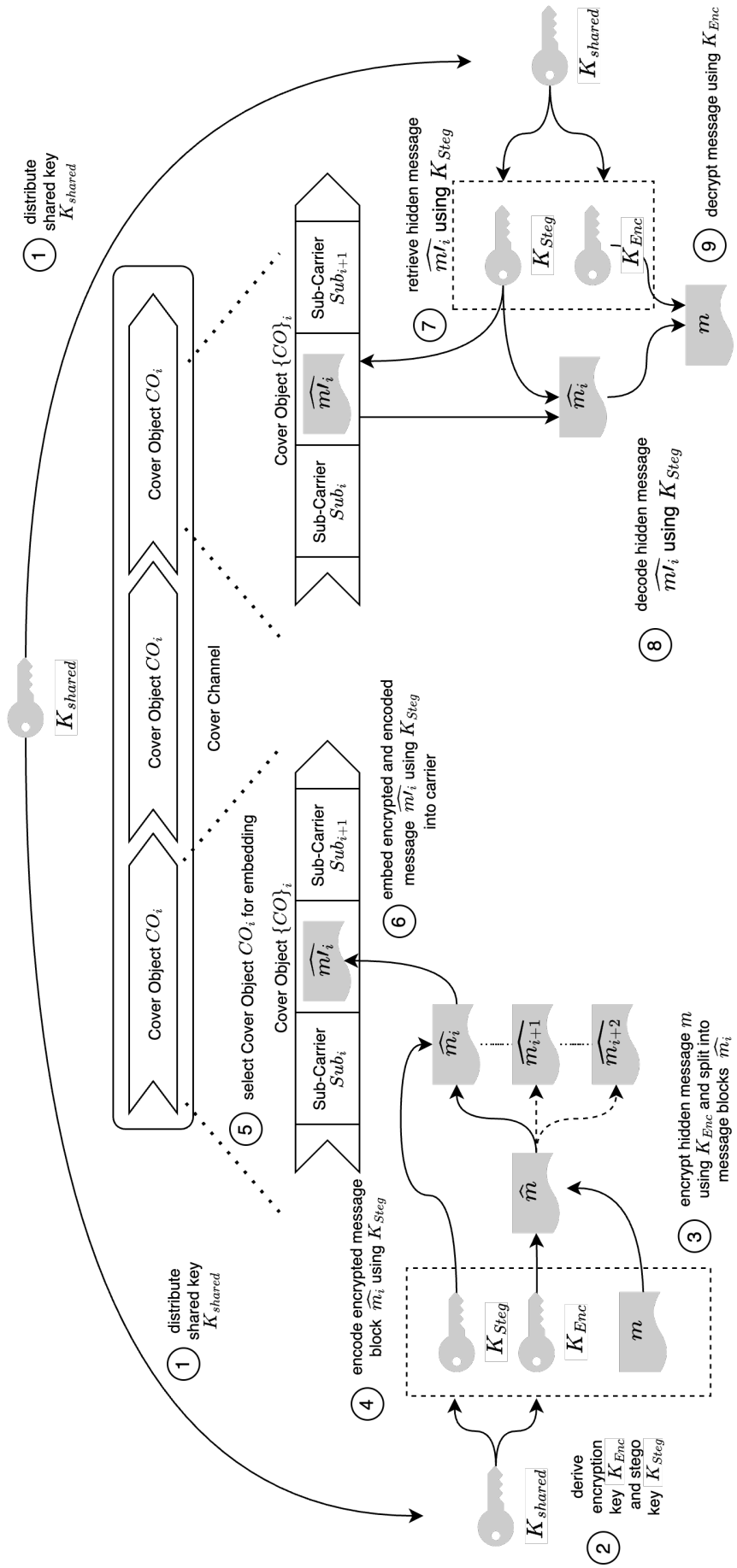


Figure 3.7: Exemplary Embedding and Retrieval process using a shared secret to derive key material.

In the following each step from this communication model, illustrated in Figure 3.7, is explained in more detail:

1. **Shared Key Distribution** The first step in this model is the distribution of a shared secret key K_{shared}. Depending on the actual threat scenario this key is either shared a priori (in the sense of the prisoners' problem *before* Alice and Bob are imprisoned; in case of malware, the key might be hardcoded or parameterized before delivery) or the key is exchanged via a specific protocol (e.g., cf. [LD22, LNH+22]). More methods for key establishment, distribution and derivation are discussed in Section 5.1.10.

2. **Key Derivation** From the shared secret K_{shared} two sub-keys are derived (e.g., using a cryptographic key derivation function). One key is used to encrypt the actual (hidden message), **encryption key** K_{Enc} and another key is derived that drives the embedding algorithm, the **stego key** K_{Steg}. For both keys **Kerkhoffs' Principle** is applied, i.e., the warden has complete knowledge of all algorithms and methods used for establishing the secret communication so that the security of the scheme therefore relies exclusively on the used *key* (see [KP00], derived originally from [Ker83] where this principle was defined (amongst other criteria) for military applications of secure communication).

3. **Encryption & Splitting** In the third step, the secret message m is encrypted using K_{Enc} and split into n message blocks $\widehat{m_i}$ to be embedded across multiple cover objects.

4. **Cover Selection** Depending on the scenario, based on K_{Steg} a cover selection process can be in place: this might include the selection of the cover channel (under the condition that multiple are available), the selection of cover objects CO_i are used for embedding, and also the selection of sub-carriers Sub_i within the cover objects to be used to embed the hidden message.

5. **Encoding** Carrier-specific encoding of message block $\widehat{m_i}$ using K_{Steg}.

6. **Embedding** Embedding of the encrypted and encoded message split $\widehat{m_i}$ using K_{Steg} into the cover object CO_i. Embedded message block is denoted as $\widehat{m_i'}$.

7. **Retrieval** Retrieve hidden message block $\widehat{m_i'}$ from the cover object CO_i using K_{Steg}.

8. **Decoding & Re-Assembly** Re-Assemble secret message $\widehat{m}$ from retrieved message blocks $\widehat{m_i}$ and decode message using K_{Steg}.

9. **Decryption** Decrypt plaintext secret message from $\widehat{m}$ using K_{Enc}.

This model provides the foundation for the evaluation and comparison of the following case studies in Chapter 4.

3.5.1.1 Information Hiding Communication Scenarios

As described in Section 2.1.5, Information Hiding communication scenarios are generally divided into two categories, *active* and *passive* hiding scenarios. Based on this differentation, in [LD20], we proposed to extend this model with additional scenarios, named *semi-active* and *semi-passive* hiding. In these communication scenarios, four participants with specific roles in communication are defined:

Alice represents a Covert Sender CS.

Bob represents a Covert Receiver CR.

Oscar represents an Overt Sender OS.

Orwell represents an Overt Receiver OR.

In the following, all four communication scenarios are described in more detail.

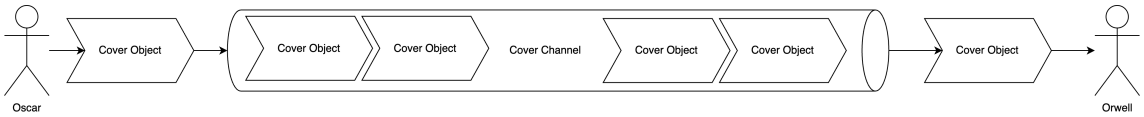


Figure 3.8: Overt Communication Model between Oscar OS and Orwell OR without hidden messages in the sense of Simmons' Prisoners' Problem [Sim84].

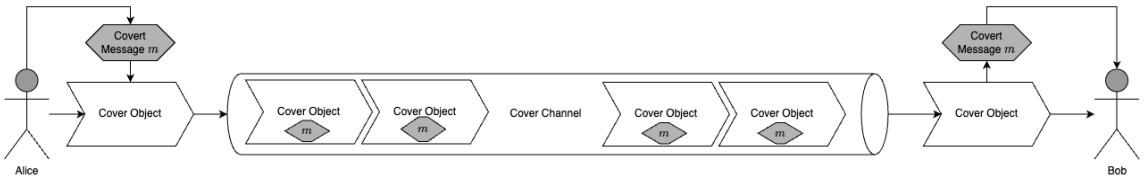


Figure 3.9: Active Information Hiding scenario in the in sense of Simmons' Prisoners' Problem [Sim84] with Alice and Bob as participants in the overt as well as covert communication. As Alice and Bob are both originator of overt as well as the hidden communication, Alice is the overt and covert sender $(OS + CS)$ and Bob overt and covert receiver $(OR + CR)$ at the same time.

The active hiding scenario is equivalent to the communication model in Simmons' Prisoners' Problem, where Alice and Bob are overtly communicating with each other, while at the same time hiding messages within this communication. In this model, as illustrated in Figure 3.9, Alice acts as an overt sender (OS) and covert sender(CS) at the same time. The same holds true for Bob, who is acting as overt receiver (OR) and covert receiver (CR) simultaneously, i.e., Alice *embeds* hidden messages into her communication with Bob who *retrieves* these hidden messages from the communication with Alice. Based on Section 3.2, this scenario is likely to be found in the (process- /automation-) communication between OT components, for example HMIs, PLCs, Engineering Workstations et cetera.

In contrast to that, in a **passive** hiding scenario, illustrated in Figure 3.10, Alice (covert sender, CS) and Bob (covert receiver, CR) are using legitimate, overt traffic

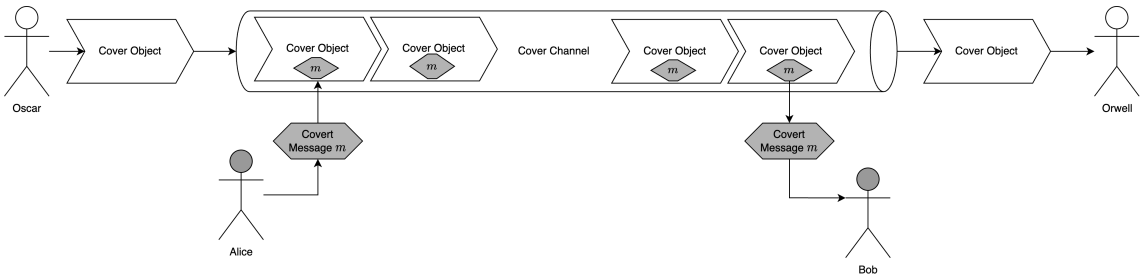


Figure 3.10: Passive Information Hiding scenario as published in [LD20] in which Alice (covert sender, CS) and Bob (covert receiver, CR are using overt traffic from innocent persons (here Oscar as overt sender OS and Orwell as overt receiver OR) as cover.

of other (innocent) persons to hide their messages. In this model, the overt sender (OS) is named **Os**car, the overt receiver (OR) **Or**well. In this Man-in-the-Middle (MitM)-like scenario, Alice (CS) *embeds* hidden messages into the overt communication from Oscar (OS) to Orwell (OR). Bob (CR) intercepts this communication to retrieve the hidden message from the cover object. Based on Section 3.2, this scenario is likely to be found with network components (e.g., switches, firewalls, data diodes etc.) as covert sender/receiver which have access to communication of other devices, in CPS for example OT components.

If it is possible for Bob to retrieve the covert message from the cover object and restore it to its original form (before the embedding) so it is equivalent to the original cover object from the overt sender and forwards this restored object to Orwell, this covert channel can considered to be **reversible** in the sense of *reversible covert channels* proposed by Mazurczyk et al. in 2019 (see [MSWC19]).

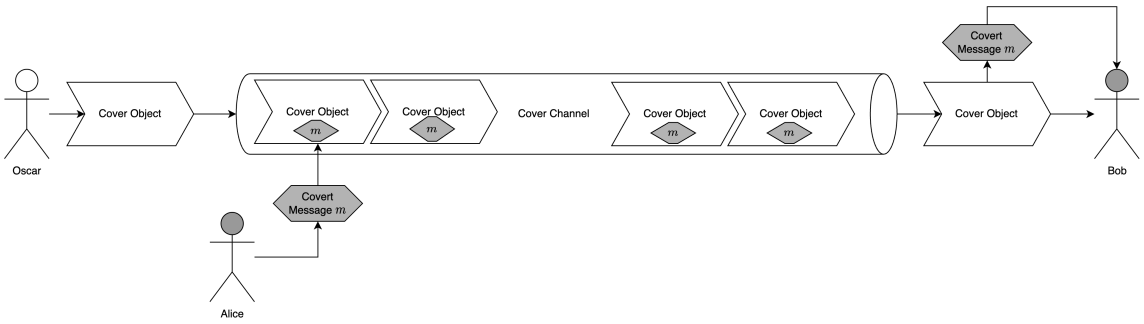


Figure 3.11: Semi-Active Information Hiding scenario based on [LD20] in which Alice (overt and covert sender, $OS + CS$) and Bob (covert receiver, CR) are using overt traffic between Alice and Orwell (overt sender, OR) as cover.

In the *semi-active* scenario, as proposed in [LD20], Alice is both overt and covert sender ($OS+CS$), while Bob as covert receiver (CR) is intercepting the cover object with the embedded hidden message before it gets transmitted to Orwell, the overt receiver (OR). In CPS, this scenario might occur in the (process- /automation-) communication between OT components, with an OT component as covert sender and a network devices as covert receiver.

Vice versa, in the *semi-passive* scenario, as proposed in [LD20], Bob is both overt and covert receiver ($OR+CR$), while Alice acts as covert sender (CS), intercepting the overt communication between Oscar, the overt sender CS and Bob. In CPS, this scenario might occur in the (process- /automation-) communication between OT components, with a network device as covert sender and an OT component as covert receiver.

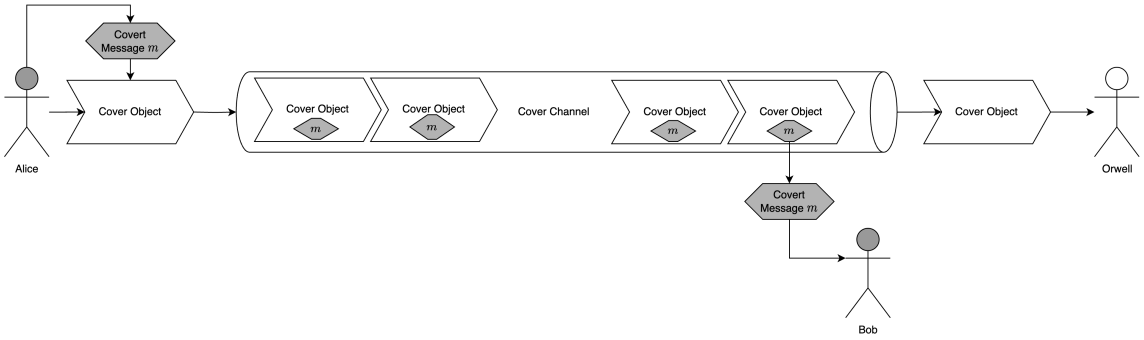


Figure 3.12: Semi-Passive Information Hiding scenario based on [LD20] in which Alice (covert sender, CS) and Bob are using overt traffic between Oscar (overt sender, OS) and Bob (overt and covert receiver, $OR+CR$) as cover.

3.5.2 Selected Performance Metrics for the Description of Covert Channels

Based on the State-of-Art in Chapter 2, in the following the chosen performance metrics for this work are briefly defined. Additionally, three other aspects for the description of covert channels (plausibility, protocol- and warden compliance) are described.

3.5.2.1 Bandwidth, Capacity, Robustness, Undetectability

Based on the (steganographic) *magic triangle*, described in Section 2.1.7, following features will be used to describe and compare covert channels:

Steganographic Bandwidth describes the amount of hidden information transferred by time unit.

Steganographic Capacity describes the amount of hidden information per carrier (object).

Undetectability/Stealthiness describes the type or amount of work or chances for realiable successfull detection by a warden.

Robustness describes the relationship between successful and failed hidden transmissions. Failed transmissions may occur by accident, e.g. due to noise, or by purpose, e.g., due to an active warden.

3.5.2.2 Plausibility, Protocol- and Warden Compliance

The contents of this section have been peer reviewed and published in part within the scope of the following publications as joint work with the co-author Jana Dittmann: [LD20].

In [LD20], we introduced the terms *Protocol-Compliance* and *Warden-Compliance* to describe the plausibility of covert channels, i.e., a covert is considered to be plausible if two conditions are met:

Protocol Compliance A covert channel is **protocol-compliant**, when the modification of the cover objects does not break the (communication) protocol in a way that the recipient would not receive, accept or process the packet, i.e. the overt communication is not affected in noticeable way for the warden.

Warden Compliance Warden-Compliance is a graded approach to describe the detectability based on probability. In [LD20], we proposed three levels of warden-compliance:

- **L1** The message is hidden in a way that a potential warden has no knowledge of the existence of a hidden message (inconspicuous).
- **L2** The warden has a suspicion that there is a hidden message but can not access it (e.g., an anomaly detection systems triggers, yet can not specify which packet of flow is off, or can specify a packet, but not which part of it contains the hidden message, which is the case when the warden has no knowledge on the stego key K_{steg})
- **L3** the warden can identify and access but not reconstruct the hidden message, i.e. when the warden does not have the encryption key K_{enc}.

Based on the insights of the following case studies in Chapter 5 these concepts are extended and improved on.

3.6 Adversarial Lateral/Vertical Movement by Application of Information Hiding in ICS

As described in the Fundamentals in Section 2.2.3, Industrial networks commonly can be described using a vertical aligned reference architecture, called PERA [Wil92]. In addition to a vertical segegration, horizontal (network)-segegration is common as well. This is also reflected in the Plausible Reference Architecture of Section 3.1. Each vertical *level* can be further split into (security-) *zones* or *cells*. These can be implemented as *logical cells* e.g., as shown in Figure 3.2, where components are grouped by their function in to different cells. Another form of segmentation is to use (firewall-) segregated *(security-) zones* as part of a defense-in-depth architecture, e.g., the IAEA Graded Approach (NSS-17T) [IAE21], aimed at preventing or mitigating adversarial lateral movement. Levels can be *functional levels* as described in PERA [Wil92] or defined as *Security-Level* for example in case of the IAEA NSS-17T recommendations for a secure architecture [IAE21]. Security Zones may be distributed over multiple PERA-levels. An exemplary security-zoning architecture (named *PLauRA-3*), based on the Plausible Reference Architecture *PlauRa-1*, is illustrated in Figure 3.13. At this example, adversarial application scenarios for covert channels are categorized as follows.

Following the common distinction between *North-South* and *East-West* traffic flows in IT networks as well as the concept of *Lateral Movement* in adversarial behavior modeling [ABS21], hidden communication in Industrial Control Systems can be dinstinguished in *vertical* and *lateral* communication:

Lateral Covert Communication Covert Channels are used to move/communicate in East-West or West-East direction within a zone/cell or between zone/cells:

- **Lateral Intra-Zone** Hidden communication between agents/components in the same zone or cell.
- **Lateral Inter-Zone** Hidden communication between agents/components of different zones or cells.

Vertical Covert Communication Covert Channels are used to move/communicate in North-South or South-Nord direction across levels.

- **Vertical Inter-Level** Hidden communication between agents/components in neighboring levels.
- **Vertical Cross-Level** Hidden communication between agents/components skipping levels in between.

A special case of *Vertical communication* is the pivot between IT and OT networks and vice versa, illustrated in Figure 3.13 as *vertical cross-level* communication, passing the IT/OT DMZ. This is a case of special interest, as in many cases IT and OT networks should be properly segregated, mitigating the threat that adversaries compromise the IT network and pivot to OT[2].

[2]Or vice versa in case of an initial compromise in the OT environment.

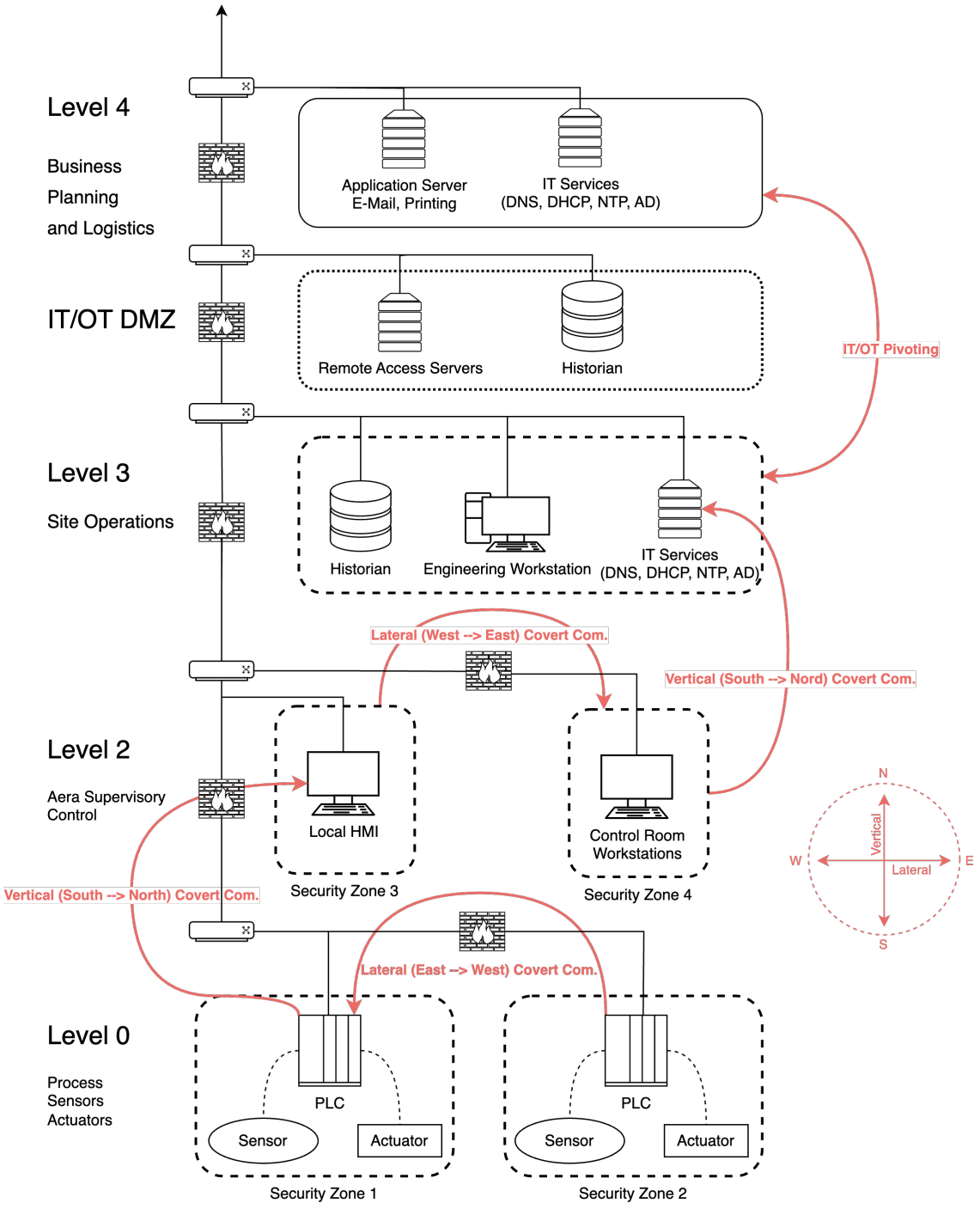


Figure 3.13: Illustration of adversarial lateral and vertical hidden communication at the example of the Plausible Reference ICS Architecture with additional firewall-segregated Security Zones (Compartmentalization) **PlauRA-3** to highlight the importance of covert channels for stealthy lateral and vertical movement.

In addition to this special case, the *purpose* of the communication can also be distinguished. In general, the purpose is in line with its *direction*:

Infiltration Unidirectional communication in North-South direction aimed at bringing information from the adversary into a target network, generally towards a compromised device as receiver.

Exfiltration Unidirectional communication in South-North direction aimed at bringing information out of a target network, typically from a compromised device device towards the adversary.

Command-and-Control (C2) A bidirectional communication channel (combination of infiltration and exfiltration). Generally used for full remote control of compromized devices.

For these types of communication, the use of information hiding techniques, such as covert channels, are of special interest to make a detection of the attack less likely.

4. Selected Case Studies

4.1 *Case Study CS_1*: Modbus/TCP

Parts of this case study have been peer-reviewed and published in part within the scope of the following publication as joint work with the co-author Jana Dittmann: [LD20].

Since Modbus/TCP is one of the most widely used non-proprietary ICS protocols, Modbus, besides OPC (UA) can be expected to be used in adversarial attacks within Operational Technology. This assumption is also backed by recent incidents, involving Modbus/TCP in Cyber attacks, e.g., in the case of Pipedream/Incontroller [Dra22, NKK+22].

This case study examines the potential for covert channels in Modbus/TCP by applying the Network Information Hiding Taxonomy to its specifications. A subset of these channels are then implemented and tested and evaluated using real-world datasets. Additionally, a testbed was created to generate realistic, reproducible datasets that can be used in combination with the publicly available datasets to evaluate the performance of the implemented covert channels. For the implemented covert channels, detection and mitigation methods are designed and discussed. Finally, the potential use of Modbus/TCP covert channels by adversaries is discussed in the context of the Plausible Reference Architecture of Section 3.1 and the description of adversarial lateral and vertical movement of Section 3.6.

4.1.1 Technical Background of Modbus/TCP

As stated in the [Mod12] specification, Modbus is an application layer (OSI layer 7) messaging protocol based on a client/server model that enables communication between devices connected to different types of buses or networks. Modbus was introduced in 1979, quickly became the defacto industry standard for serial communication in ICS networks, and was later ported to TCP/IP [Mod06]. Modbus uses a request/response model and specifies function codes for accessing specific services of a device. The PDU (Protocol Data Unit) is independent of the layers below and contains a function code describing what action is to be performed, as well as the actual user data. As shown in Figure 4.1, around the PDU lies the ADU (Applica-

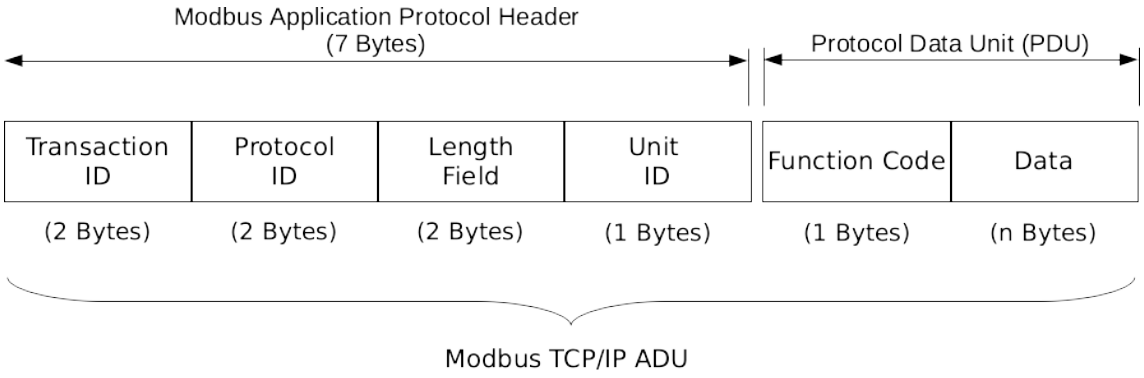


Figure 4.1: Modbus/TCP Frame with Application Data Unit (ADU) und Protocol Data Unit (PDU) as published in [Mod06]. Figure based on [LD20].

tion Data Unit), which in the case of Modbus/TCP consists of the MBAP (Modbus

Application Protocol) header and the PDU (function code plus data). The MPAB header contains four fields: a Transaction Identifier (Request/Response), a Protocol Identifier (static for Modbus), length and a Unit Identifier (for addressing remote slaves). According to the specification [Mod06], the maximum allowable size of a Modbus/TCP Frame are 260 bytes (ADU = 253 bytes PDU + 7 bytes MBAP = 260 bytes). The Modbus specification defines three categories of function codes: (1) Public Function Codes, (2) User-defined Function Codes, and (3) Reserved Function Codes. In this work, we focus on eight public function codes, which are described in Table 4.1, as those are the most prevalent in common, real-world deployments [Mod06, LD20].

Table 4.1: Overview on Modbus Function Codes.

Data Access	Physical/Internal	FC	Description
Bit access	Internal Bits / Physical Coils	01	Read Coils
		05	Write Single Coil
		15	Write Multiple Coils
	Physical Discrete Inputs	02	Read Discrete Inputs
16 bits access	Physical Input Registers	04	Read Input Register
	Internal Registers or Physical Output Registers	03	Read Holding Registers
		06	Write Single Register
		16	Write Multiple Registers

4.1.2 Participants in Hidden Communication

Since Modbus distinguishes between servers and clients, or more precisely between slaves (servers) and masters (clients), the capabilities between them are also very different. In the Modbus protocol, only clients (masters) are able to initiate communication. PLCs usually work mainly as slaves, while HMIs and engineering workstations generally work as masters to query I/O or set parameters of the PLCs. PLCs can also act as masters when communicating with other PLCs or devices. Therefore, the analysis distinguishes whether only the slave, the master, or both are capable of embedding or extracting hidden information in a given hidden channel. For the sake of simplicity and comparability to other protocols using client-server models, in the following Modbus masters are named *clients* synonymously and Modbus slaves considered to be *servers*.

Modbus/TCP is typically found in communications between (1) PLCs, (2) engineering workstations, (3) human-machine interfaces (HMIs), and (4) historians. Therefore, these devices can be seen as the primary participants in hidden communication. Since network elements such as (5) switches, (6) hubs, (7) firewalls, and (8) protocol converters have access to Modbus/TCP network traffic without being the originator, they have to considered as potential transmitters and receivers of covert information as well. In terms of the reference architecture from Section 3.1, Modbus/TCP covert channels are located in Levels 1-3.

4.1.3 Public Modbus/TCP Datasets

In order to properly evaluate covert channels in terms of their parametrization and performance it is important to validate results against multiple datasets which serve as cover for hidden channels to avoid biases induced by the selected dataset.

Although Modbus/TCP itself might be considered fairly easily structured when compared to other protocols from the like of OPC UA the analysis of publicly available datasets (pcap-files) of recorded Modbus/TCP traffic shows significant differences in the use of available functions and features of Modbus/TCP. In the following, these differences are highlighted and described as to why these motivate the generation of an own custom dataset (in the following DS_1) to be used in the following evaluation. For this purpose, five datasets from different sources were selected for further investigation:

DS_2 **Cyberville** Network capture from the 2020 SANS ICS Virtual Conference Capture-the-Flag (CTF) [Rob20]. Contains Modbus/TCP as well as Siemens S7Comm.

DS_3 **DEFCON23** Packet captures from the ICS village at the DEF CON 23 conference, 2020 [DEF20]. Contains mainly Profinet and some samples of Modbus/TCP.

DS_4 **CRITIS18-1v2** Clean network capture (`eth2dump-clean-1h_1.pcap` from the 1v2-capture) by Frazão et al. [Cru18], presented at CRITIS 2018 [FAC+19]. Contains one hour of clean Modbus/TCP traffic (i.e., capture does not contain any conducted attacks).

DS_5 **Lemay-run8** Modbus network captures by Antoine Lemay [LFK16]. `run8.pcap` used for investigation; contains one hour of non-modified Modbus/TCP traffic [Lem16].

DS_6 **Chinese CTF** contains Modbus/TCP and Siemens S7Comm traffic from a Chinese Capture-the-Flag [New18].

Table 4.2 compares these datasets in terms of their duration, how many frames are included in the PCAP file, how many of these frames are Modbus/TCP packets and the amount of Modbus packets per minute. From this table, it becomes clear how diverse these datasets are. The duration ranges from 3 minutes (DS_6 *chinese-ctf*) to over 5 hours (DS_3 *DEFCON23)*. The same holds true for the amount of packets. While DS_4 *CRITIS18-1v2* dataset only contains around 72,000 frames, the DS_3 *DEFCON23* datasets contains over 1.3 million frame, around 20 times more. Despite the amount of 1.3 million frames in the DS_3 *DEFCON23* dataset, only 711 frames are Modbus/TCP packets. Another outlier is the amount of packets per minute in DS_6 (*chinese-ctf*). In only 3 minutes there are over 200,000 packets, equaling around 29,000 Modbus frames per minute. This might be the result of artificially injected traffic and seems rather unrealistic for a real-world environment.

In addition to the duration, total number of frames and the relative amount of Modbus/TCP packets, the use of different features, i.e., *function codes* are important

Table 4.2: Overview on overt Modbus/TCP datasets.

		Meta		Modbus		
		Duration	Frames	Abs.	Rel.	Pkts/min.
DS_2	*Cyberville*	143	252,310	32,623	12.93%	228
DS_3	*DEFCON23*	338	1,368,167	711	0.05%	2
DS_4	*CRITIS18-1v2*	60	72,150	24,245	33.60%	410
DS_5	*Lemay-run8*	60	72,186	13,022	18.04%	220
DS_6	*chinese-ctf*	3	203,192	87,281	42.95%	29,093
DS_1	*Generated Dataset*	60	137,175	80,765	58.88%	1,346

for the analysis of covert channels as some might require certain fields or functions for embedding or retrieval.

Table 4.3 compares the absolute and relative amounts of function codes used in the datasets. The comparison shows significant differences between the datasets.

For example, Function Codes $FC05$ (Write Single Coils) and $FC02$ (Read Discrete Input) are only used in datasets DS_2 and DS_5. Function code $FC16$ (Write Multiple Registers) surprisingly is only found in DS_6.

From the public datasets, DS_2 is the most diverse, comprising all relevant function codes with the aforementioned exception of $FC16$.

DS_3 only contains read operations with 24% distributed on $FC1$ (Read Coils) and 76% on $FC3$ (Read Holding Registers).

DS_4 contains only Read Holding Registers ($FC03$) packets and by that represents the least diverse dataset.

DS_5 is mainly contains read-requests with the exception of 1.5% write single coils packets ($FC05$) The read requests are fairly distributed with roughly one third each on $FC1$ (Read Coils), $FC02$ (Read Discrete Inputs) and $FC03$ (Read Holding Registers).

DS_6 is distributed quite unevenly: There are two $FC01$ (Write Single Coil) and two $FC15$ (Write Multiple Coils) packets and 120 $FC16$ (Write Multiple Registers) packets. In total these make only 0.14% of the whole dataset. The remaining 99.86% of Modbus packets are Read Holding Registers ($FC03$).

This comparison highlights the major differences and biases in the datasets. As most of these datasets lack proper documentation, the environments in which the data was captured is generally unknown and therefore unreproducible. Despite these major differences in the datasets, the diversity in the datasets provides a way to evaluate covert channels in varying environments - a circumstance an adversary has to face in real-world (threat) scenarios as well.

Nevertheless, from a scientific perspective, it is important to have a documented and reproducible test environment to achieve a consistent, diverse and representative dataset.

Therefore, the following section describes the reproducible test environment that is used to generate a realistic dataset that is used as a baseline in combination and comparison with the public datasets for the evaluation of covert channels.

Table 4.3: Comparison of public datasets with the custom generated dataset as baseline in regards to absolute and relative amounts of Modbus frames sorted by their function code: $FC1$ *ReadCoils*, $FC15$ *WriteMultipleCoils*, $FC05$ *WriteCoil*, $FC02$ *ReadDiscreteInputs*, $FC03$ *ReadHoldingRegisters*, $FC16$ *WriteHoldingRegisters*.

Dataset		$FC1$		$FC15$		$FC05$		$FC02$		$FC03$		$FC16$	
		Abs.	Rel.	Abs.	Rel.	Abs.	Rel.	Abs.	Rel.	Abs.	Rel.	Abs.	Rel.
Public Datasets	DS_2	7,580	23.24%	2,716	8.33%	2,716	8.33%	3,933	12.06%	10,245	31.40%	0	0.00%
	DS_3	171	24.05%	0	0.00%	0	0.00%	0	0.00%	540	75.95%	0	0.00%
	DS_4	0	0.00%	0	0.00%	0	0.00%	0	0.00%	23,003	94.88%	0	0.00%
	DS_5	4,272	32.81%	0	0.00%	198	1.52%	4,274	32.82%	4,278	32.85%	0	0.00%
	DS_6	2	0.00%	2	0.00%	0	0.00%	0	0.0%	87,157	99.86%	120	0.14%
Generated	DS_1	36,091	44.69%	1,444	1.79%	0	0.00%	0	0.00%	14,436	17.87%	28,794	35.65%

4.1.4 Custom Dataset Generation (Modbus/TCP Testbed)

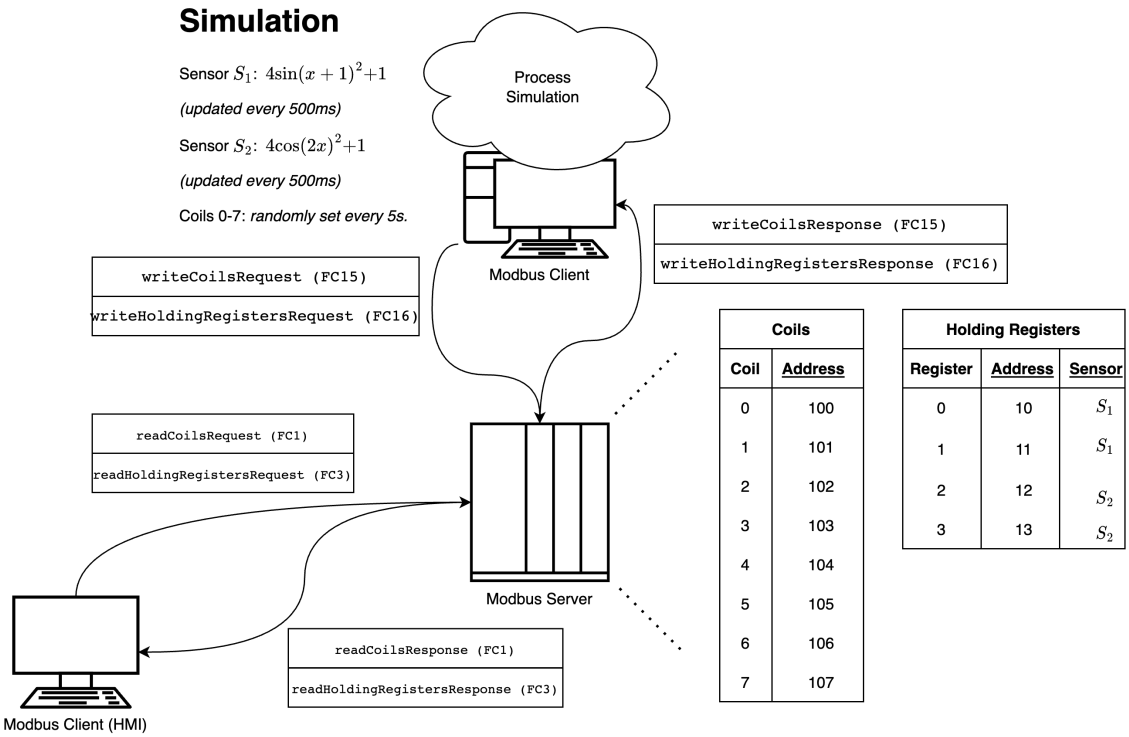


Figure 4.2: Overview of the Modbus/TCP testbed used for representative dataset generation using a process simulation, two Modbus clients and one Modbus server.

For the evaluation of Modbus/TCP an open-source based testbed is designed to generate a realistic, representative and reproducible dataset for covert channel analysis. The basis for this testbed is `node-red`[1], using the `node-red-contrib-modbus`[2] package to fully simulate Modbus/TCP server and clients.

Figure 4.2 illustrates the full simulation. Central point is the Modbus Server. On it, eight coils are used and 4 holding registers. A Modbus client is driving the process simulation. In order to simulate realistic process data, two sensors are emulated. The simulation function for sensor S_1 is $4sin(x+1)^2+1$, where x is a counter, that is increased by 0.25 every $500ms$. Sensor S_2 is simulated using $4cos(2x)^2+1$ with the same counter and update interval. The Modbus process simulation client (Modbus master) encodes and splits the sensor values (32bit float) as two 16-bit unsigned integer for transmission as one holding register can only hold 16 bit. The values of sensor S_1 are stored in the Holding Registers $10+11$, sensor values S_2 in Holding Registers $12+13$. Each sensor value is transmitted using Function Code $FC16$, `writeHoldingRegistersRequest` packets. The process simulation also emulates eight coils which are (randomly) set every 5 seconds (true/false). The Modbus process simulation client writes all eight coils at once using Function Code $FC15$ (`writeCoilsRequest`) at once to the registers $100-107$.

Using this testbed, a representative dataset is generated over an duration of one hour. The resulting dataset (see bottom row of Table 4.2) has a total of $137,175$

[1] https://nodered.org
[2] https://flows.nodered.org/node/node-red-contrib-modbus

packets which when filtered for Modbus/TCP traffic only, consists of 80, 765 Modbus frames (relative amount of 58.88%). In the dataset on average 1, 346 Modbus packets are sent per minute. The distribution of function codes is shown in the bottom row of Table 4.3. 44.69% of all Modbus frames are `ReadCoils` (Function Code 1) request and response packets. 17.87% are `ReadHoldingRegisters` (Function Code $FC03$) request and response packets. 1.79% are `WriteCoils` (Function Code $FC15$) request and response packets. 35, 65% are `WriteHoldingRegisters` (Function Code $FC16$) request and response packets.

Due to how the process is simulated, this test setup does not use WriteSingleRegister ($FC06$) packets. This is due to the fact that for one sensor value (32bit) two holding registers are required (16bit) and using two separate packets would be unrealistic. The same is true for $FC05$ writeSingleCoil packets. In the simulation all eight coils are computed simultaneously. Therefore, again the usage of multiple $FC05$ writeSingleCoil would be overhead that is avoided by using only $FC16$ (WriteMultipleCoils). This leads to the side effect of having large amounts of $FC16$ packets compared to public datasets: The custom dataset has around 28,000 $FC16$ while all others have none or only 120 in case of DS_6.

4.1.5 Derivation of potential covert channels using Network Information Hiding Patterns

The following two passages summarize, ***extend and enhance*** *the results from [LD20] with insights from further analysis.* In [LD20] the storage patterns from the the 2018 version of the extended taxonomy for Network Information Hiding Patterns [MWC18] are applied to Modbus/TCP to systematically identify potential, plausible covert channels. From these identified covert channels a subset is selected for implementation and further evaluation in realistic testbests and real-world datasets. For the sake of simplicity and clarity, simplified identifiers, that are different from the original taxonomy, are used in the following to name the ten storage (S1-S10) and eight timing channels (T1-T8) from the taxonomy.

4.1.5.1 Covert Timing Patterns applied to Modbus/TCP

As shown in Figure 4.3, the Network Information Hiding taxonomy differentiates between *Protocol-Agnostic* and *Protocol-Aware* methods.

Protocol-agnostic patterns are applied without any specific knowledge on the used protocol. These include the modulation of inter-packet times (T1) (time delta between consecutive network packets), message timing (T2) and (T3) Rate / Throughput Modulation.

T1 Inter-packet Times alters the timing intervals between network packets of a flow (inter-arrival times) to encode a message or hidden information. As the communication in ICS networks is heavily determined by cycle times of PLCs, the inter-packet times should vary only in a small window, i.e. the noise is very low. This aids in creating robust covert channels, on the other hand, the risk of detection is comparably higher than in common IT networks.

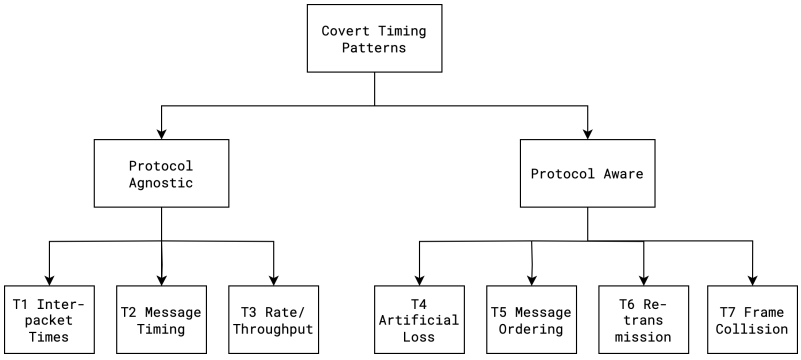


Figure 4.3: Simplified illustration of timing patterns from the 2018 extended Network Information Hiding Pattern Taxonomy [MWC18] with the omission of *PT14. Temperature*. In this work, the *PT14* pattern is classified as *Out-of-Band Covert Channel* and therefore not included in the (Network-) Timing patterns.

The common way for applying this pattern is to delay certain packets of a flow in order to encode a message. The limitation is given by the timeout of the communication partner, which then classifies the packet as not-received, which can result in (unwanted) re-transmissions. Too many retransmissions may raise suspicion and can lead to detection by a warden. Depending on the timeout settings, even more symbols could be encoded by delaying packets in different timing windows. The modulation of inter-packet times can be performed by Modbus Slaves as well as Masters with the limitation that only Masters can initiate a communication. Therefore, a transmitting Slave has to ”wait” for an incoming communication. The same holds true for any network devices that route or forward traffic. The **capacity** of the channel depends on the throughput of regular packets, the encoding, and the specified timeout settings. In a simple approach, where each packet is delayed or not, the **capacity** would be 1 bit/packet. This pattern is quite common, as it is protocol-agnostic and can be applied to basically any protocol found in targeted networks.

The **T2 Message Timing** pattern encodes data in the timing of message sequences within a flow, an example derived from the taxonomy description in [MWC18] would be to send a specific command x times. For Mobdus/TCP this pattern applies only to Modbus Master devices as they initiate a communication flow. A man-in-the-middle network device would be capable as well by simply duplicating packages of a Master device. A slave device that would change the timing of messages within a flow would violate the specification (protocol-compliance) and therefore raise suspicion (warden-compliance). A reasonable scenario would be an HMI sending an I/O read or write request two or three times instead of only once to encode a message. As the traffic would derive significantly from the usual traffic flow, a detection using anomaly detection based algorithms seem highly probable.

In the **T3 Rate/Throughput** pattern, the sender alters the data rate of a packet flow that is directed to the covert receiver. Usually the data rate of flows between Modbus devices are stable over time, for example an HMI querying and setting I/Os multiple times per second. An attacker might encode a hidden message by modulating this rate by applying a multiplicand k, in the form of $encode(bit) =$

$(x * k)packets/second$, where for example $k = 2$ could encode a binary one and $k = 1/2$ a binary zero respectively. For Modbus/TCP this is only applicable to devices that act as a Modbus Master, or take part in the communication as a network device. The **capacity** of this channels depends on the scale of the data rate which is modulated. An adversary could use packets per second as shown in the example, or use longer terms, for example packets per hour or day, which would result in lower bit rates but lower detection probability as well. Whereas in the example only one bit per flow is encoded, a higher bandwidth can technically be achieved by using more multiplicands to encode a message. As the data rate usually keeps in a well-defined window, anomaly detection based schemes should be able to detect this type of covert channel.

As the name suggests, **T4 Artificial Loss** makes use of dropping or corrupting frames to encode a message. While still applicable to Modbus devices, this pattern is especially useful for passive information hiding done by network elements. As Modbus is transferred over TCP, frame drops or packet losses induce re-transmissions. This limits the amount of achievable bandwidth without raising too much attention. However, the warden/defender might not directly relate these re-transmission to covert channels but rather to networking issues.

The **T5 Message Ordering** pattern encodes information by altering the order of packets/message in a flow. This pattern does not work for every protocol, as a change of order might break functionality. However, in case of Modbus/TCP often there are interchangeable flows of packets - for example an HMI which queries first the status of multiple coils and then the registers. This order can be changed without affecting the protocol. As no data is being altered or timings shifted, this is an inconspicuous pattern when implemented in Modbus/TCP. The pattern can be implemented as an active information hiding technique by Modbus Masters as well as Slaves, by altering the order of reply packets. This alteration in the response is only possible as Modbus/TCP uses Transaction Identifiers which separate each request from another. Therefore, the client can correlate the answers coming from the server, even in the incorrect order. This pattern can also be implemented by network elements for a passive approach or as an hybrid. The **capacity** depends on the amount of packets in a flow. By using different orders of the packet flow, multiple bits can be represented. In the simplest scenario, when the message order is either correct or altered, one bit per flow is transmitted.

Pattern **T6 Retransmissions** uses artificial retransmissions by transmitting previously sent or received messages again or by taking actions leading to retransmissions e.g., by corrupting checksums. This pattern can be used by any element that takes part in the communication as it depends only on previously sent packages and therefore is especially suitably for man-in-the-middle scenarios (semi-active, semi-passive and passive hiding). Retransmissions tend to offer low bandwidths as they usually do not appear often in regular traffic flows and can therefore only be used over longer periods of time in order not to rise any suspicion. One method to encode a bit is to define a threshold for how many retransmissions appear per timeframe. Using such a threshold-based technice increases robustness and decreases detectability when compared to alternate encodings (for example, one retransmission represents zero and two consecutive retransmissions encode a one).

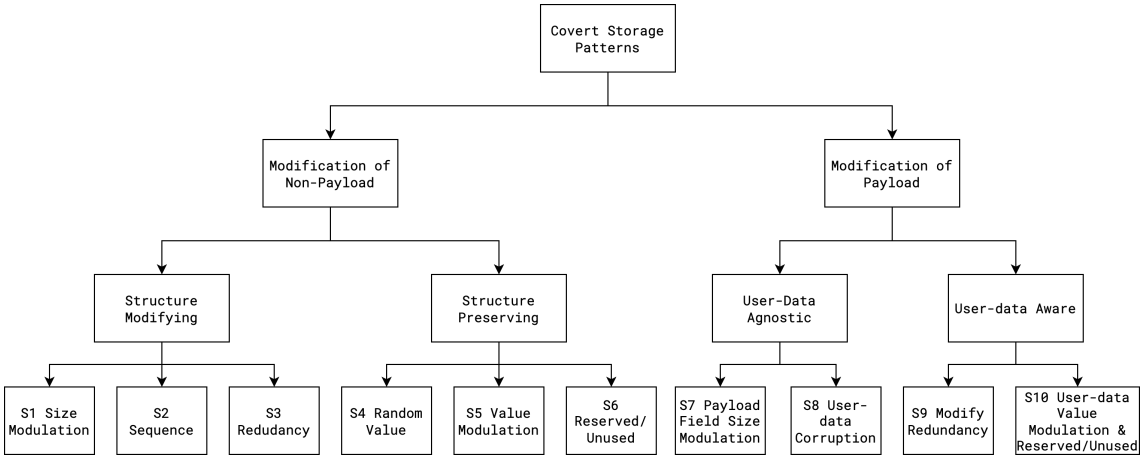


Figure 4.4: Simplified illustration of storage patterns from the 2018 extended Network Information Hiding Pattern Taxonomy [MWC18].

In Pattern **T7 Frame Collisions** the secret sender uses artificial frame collisions to encode a hidden information. In the case of full-duplex Ethernet switches frame collisions generally do not happen, therefore this patterns seems not to be applicable in the scope of this evaluation.

The idea of **T8 Temperature** is to influence the CPU temperature of a (third-party) device (for example a switch or firewall) by sending high amounts of packets in very short time to increase CPU load of that device leading to a rise of CPU temperature. The temperature has a direct effect on the clock skew which will differ depending on the current temperature of the CPU. By communicating with the (third-party) device the receiver can observe the clock skew and by that decode the hidden information. While this is a very interesting approach, in the scope of this work, this pattern is classified as **Out-of-Band Covert Channel** and due to its protocol-agnostic nature, does not directly apply to Modbus communication. However, the clock skew deviation might be observable in the TCP timestamps.

4.1.5.2 Covert Storage Patterns applied to Modbus/TCP

The first pattern, **S1 Size Modulation**, according to [MWC18] involves modifying the size fields of meta-data, i.e., protocol headers, to embed a hidden message. In case of Modbus/TCP this relates to the Modbus Application Protocol Header (MBAP, see Figure 4.1). This is an important distinction from the pattern **S7 Payload Field Size Modulation**, in which the payload size is modulated, i.e. the Packet/Protocol Data Unit (**PDU**) in Modbus/TCP. However, a change of the payload fields size, requires a change of the length field in the header to be protocol-compliant. Therefore, in most scenarios both these patterns are expected to appear hand-in-hand. For Modbus/TCP, plausible application of this pattern seems to be the length field of the MBAP, which describes the amount of bytes following this field (including the PDU). The length field values can vary between four bytes (Unit ID, Function Code and PDU with a minimum of 2 bytes) up to a maximum of 253 bytes. This pattern will be of further importance for a covert channel that is later used in the implementation and evaluation.

Pattern **S2 Sequence Modulation** alters the sequence of elements on the meta-data. Such modulation obviously does not comply with the specification and would break the communication. However, one way to implement this pattern would be to use a *reversible* approach, i.e. using a passive or hybrid approach, in which the sequence of the MBAP Header elements could be changed by a Modbus- or network device - but the modified packet would then be sent to or via a (rogue) network element which decodes the hidden message and reverses the packet to its to original state before it gets forwarded to the (original) receiver (Modbus device) (see Figure 3.10 for reference). As there are six fields in the Modbus ADU, there are 720 possible permutations, leading to a capacity of 9 usable bit per packet when using a binary encoding ($log_2(720)$) for the order of fields.

Pattern **S3 Add Redundancy** involves adding additional, unused information to metadata. Given only the limited options available in Modbus, a variation of the pattern could be used, in which an adversary would use different function codes in their messages while achieving the same results when using the original function code: Modbus/TCP has the option to write/force single registers/coils using function code FC5 for a forcing a status of a single coil and function code FC6 for writing to one holding register. In addition to that, with the function codes FC15 and FC16 multiple coils/registers can be forced/written with one packet. This circumstance can be leveraged to establish covert C2 channels that are hard to detect: Given a scenario in which a Modbus is client regularly writing single coils or registers using function codes FC5 and FC6: To encode a hidden bit, the idea is instead of using FC5/FC6 to write a coil or register to use the function codes for accessing multiple I/Os (i.e., FC 15/16), yet only specifying only one coil or register to be written. This method is only applicable for Modbus clients sending write requests, as response with a different function code would not be protocol-compliant and break the communication. A reversible variation is possible implementing a passive approach where two network elements would change the function code in the communication between those devices and reverse it back before forwarding the packet to the original recipient. Another variation is to split `WriteMultipleRegisters` (FC16) or `WriteMultipleCoils` (FC15) into multiple `WriteSingleRegister` (FC6) or `WriteSingleCoil` (FC5), respectively.

Pattern **S4 Random Values** is not applicable to Modbus/TCP as there are no random values used in the specification.

Pattern **S5 Value Modulation** modulates one of values in a field from the protocol header. In [MWC18], the authors divide this pattern into two sub-patterns: Case- and Least Significant Bit Modulation. As Modbus/TCP does not have any literals and uses only 16-bit integers, case variation is not applicable. Value Modulation of header fields in Modbus/TCP seem plausible applicable to the Transaction ID field. This field is used to correlate requests and responses. The specification defines the use of two bytes for a 16bit Integer as transaction identifier. In the following practical evaluation, a covert channel that is using this field is investigated in depth.

Pattern **S6 Reserved/Unused** uses reserved or unused fields of the metadata to embed a hidden message. In Modbus/TCP, this pattern can be applied to the Unit ID field in the MBAP header, which is only used when remote Slaves are addressed that are connected to the target devices. As this feature is rarely used,

a modulation of the UnitID can be considered to fall under the Reserved/Unused Pattern. Arguably it could also fit into the S5 Value Modulation pattern. As the Unit ID takes up one byte in the MPAB header, the maximum achievable bandwidth for hidden information is 8 bits per packet. This covert channel is applicable for Masters, Slaves, network elements, and in active, passive, and hybrid scenarios. This covert channel is protocol-compliant and requires implementation-dependent rules or anomaly detection for proper detection.

The **S7 Payload Field Size Modulation** pattern is derived from **S1 Size Modulation**. In contrast to S1, S7 modulates the size of the payload, yet not the header. When changing the payload size for embedding, the embedder likely would consider changing the length field in the header as well to avoid inconsistencies. Therefore, in many scenarios both patterns come hand-in-hand.

The maximum size of the payload can be extended to 253 bytes, which provides high capacity but is also conspicuous. However, by adding just one byte per packet, this method can still provide a capacity of 8 bits per packet without being too noticeable. This pattern is mainly applicable to Masters and network elements using active, passive, or hybrid methods (as described in Pattern S1).

In the **S8 User-data Corruption** pattern covert data is inserted (blindly) into the payload. For Modbus/TCP this would in most scenarios result in wrong values for reading and writing coils and registers. As this has a direct influence on the physical processes controlled by the Modbus devices this approach is very **obtrusive**. When used in a very low frequency though and for example for only one specific sensor reading (or similar) a warden might not detect this corruption as a cover for a hidden message, as the value might have been corrupted by the sensor itself. Therefore, this pattern can be implemented by all devices in active, passive and hybrid scenarios. The **capacity** depends on the payload length. A plausible corruption of one register would lead to a **capacity** of **16 bit/packet**.

The **S9 Modify Redundancy** pattern uses compression to free up space in the payload and embed a hidden message. However, as Modbus uses 16-bit unsigned integers, this pattern is not applicable.

The **S10 User-data Value Modulation and Reserved/Unused** pattern modifies the payload in a way that the interpretation of the data does not differ significantly from the original packet. This can be achieved by altering the least significant bits or hiding data in unused or reserved bits. For Modbus/TCP, this pattern can be implemented in multiple ways. For instance, modulation of the least significant bit of register values can encode a hidden message while leading to only minor variations in the values. This can result in a capacity of 1 bit per packet and applies to Modbus Slaves (servers) for active information hiding, as well as network elements as part of passive or hybrid embedding methods.

Since Modbus is byte-oriented, some bytes may contain ”unused” bits. Suchs bits should be set to zero. This circumstance can be exploited for covert channels. For example, when a Master (client) is querying a single coil or discrete input of a Slave (server), only one bit of the payload byte is used, leaving seven bits available for embedding a hidden message by an attacker. This results in a capacity of 7

bits per packet, and the embedding can take place on a Slave or network elements. This covert channel is also selected for further evaluation as it is inconspicuous and leverages flaws in the specification.

Table 4.4: Storage and Timing patterns (*simplified to T:Timing, S:Storage*) of [MWC18] applied to Modbus/TCP. The *Master*, *Slave* and *network* (element) column indicate whether the pattern is plausibly applicable (✓) to the specific type of device or not (✗). *Active*, *Passive* and *Hybrid* (○ indicates semi-active) indicate the applicability of the information hiding approach. *Capacity* is an estimation for maximum capacity in representative environments. The *conspicuousness* (tendency of warden-compliance) is estimated on a binary scale (high/low). Patterns used in the following evaluation are marked in bold. Table based on [LD20].

Patterns by [MWC18]		Findings for Modbus/TCP							
	Pattern	Master	Slave	Network	Active	Passive	Hybrid	Capacity	Conspic.
T1	Inter-Packet Times	✓	✓	✓	✓	✓	✓	1 bit/packet	low
T2	Message Timing	✓	✗	✗	✓	✗	○	n bit/flow	high
T3	Rate/Throughput	✓	✗	✗	✓	✗	○	n bit/flow	high
T4	Artificial Loss	✓	✓	✓	✓	✗	✓	1 bit/packet	high
T5	Message Ordering	✓	✓	✓	✓	✓	✓	1 bit/flow	low
T6	Retransmission	✓	✓	✓	✓	✓	✓	1 bit/time unit	high
T7	Frame Collisions	✗	✗	✗	✗	✗	✗	✗	✗
T8	Temperature	✓	✓	✓	✓	✓	✓		
S1	Size Modulation	✓	✓	✓	✗	✓	✓	8 bit/packet	low
S2	Sequence Modulation	✓	✗	✓	✓	✓	✓	9 bit/packet	high
S3	Add Redundancy	✓	✗	✓	✓	✓	✗	1 bit/packet	low
S4	Random Value	✗	✗	✗	✗	✗	✗	✗	✗
S5	**Value Modulation**	✗	✗	✗	✗	✗	✗	✗	✗
S6	**Reserved/Unused**	✓	✓	✓	✓	✓	✓	**8 bit/packet**	**low**
S7	**Payload Field Size Modulation**	✓	✗	✓	✓	✓	✓	**8 bit/packet**	**low**
S8	User-data Corruption	✓	✓	✓	✓	✓	✓	16 bit/packet	high
S9	Modify Redundancy	✗	✗	✗	✗	✗	✗	✗	✗
S10	**Value Modulation** & **Reserved/Unused**	✗	✓	✓	✓	✓	✓	**7 bit/packet**	**low**

4.1.6 Selected derived Covert Channels for Implementation and Evaluation

Using the previously discussed hiding patterns applied to Modbus/TCP, four covert storage channels are derived to be implemented and evaluated in more depth. These are selected based on the probability to be used by adversaries i.e., they provide the best trade-off between bandwidth, robustness, detectability and ease-of-use (see Table 4.4). For the implementation and evaluation an offline-approach is selected: Both embedding as well as retrieval are performed on (existing) pcap-files, not on live traffic. This allows for greater comparison on different datasets. However, as a side-effect this limits the evaluation to covert storage channels as the implementation and realistic evaluation of timing channels should be performed on live traffic. The covert channels derived from the selected patterns are implemented and evaluated in terms of the previous described evaluation criteria. In the following these covert channels are described in more detail:

CC_{MB1} - UnusedCoils implements *S10 Reserved/Unused* by leveraging unused bits in ReadCoilsResponse packets.

CC_{MB2} - UnitID implements *S5 Value Modulation / S6 Reserved Unused* by modulating the UnitID.

CC_{MB3} - TransID implements *S5 Value Modulation* by modulating the Transaction ID.

CC_{MB4} - Padding implements *S7 Payload Field Size Modulation* by appending additional padding to the payload of the packet.

4.1.6.1 Modbus Storage Channel 1 (CC_{MB1}) - Unused Bits in ReadCoils responses

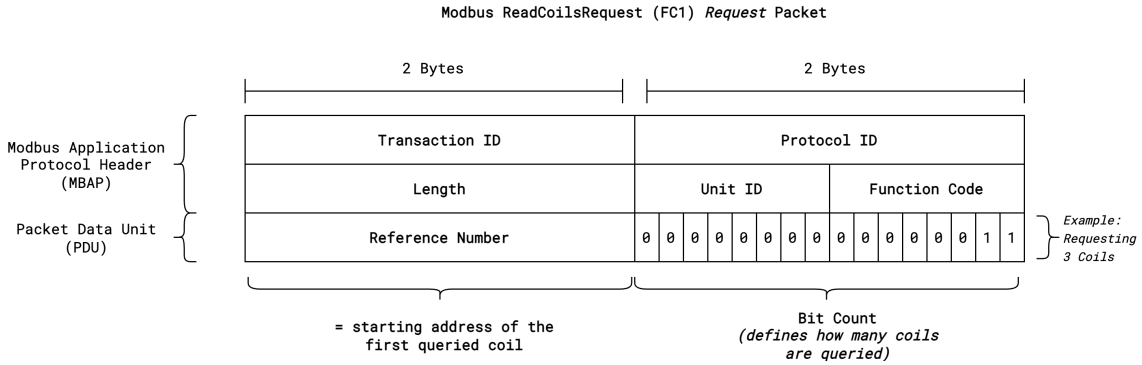


Figure 4.5: Illustration of Modbus/TCP a `ReadCoils` (Function Code FC1) request packet.

From an adversaries perspective one of the most interesting channels is to use unused bits in Modbus `ReadCoils` (Function Code FC1) response packets. This covert channel does not affect overt communication, provides sufficient bandwidth for most threat scenarios and requires the defenders to have specific knowledge and measures in place to successfully detect and mitigate this covert channel.

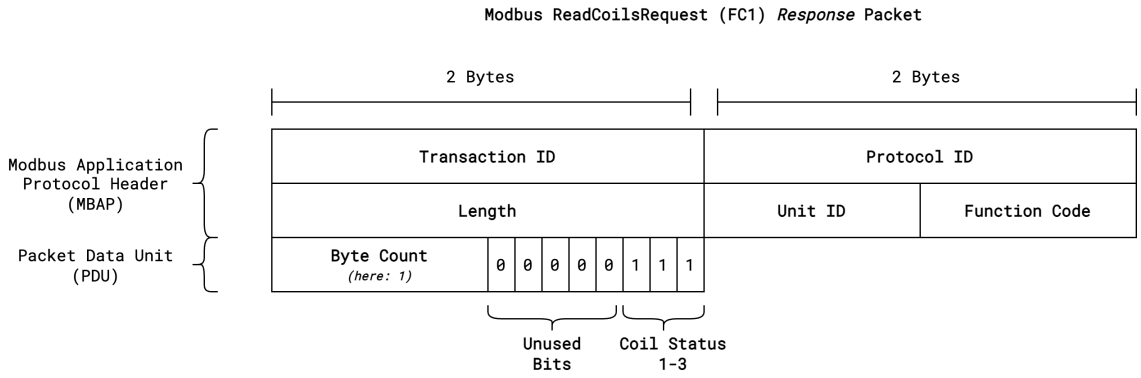


Figure 4.6: Illustration of unused bits in Modbus/TCP `ReadCoils` (Function Code FC1) response packets.

As illustrated in (see Figure 4.5), in `ReadCoils` request packets the client specifies the starting address (`Reference Number`) and the amount of coils to be requested in the `Bit Count` field. For example, if the (`Reference Number`) is 100 and the the `Bit Count` field 3, then the server (e.g., a PLC) will respond with the status of the coils stored in the internal registers/storage addresses 100, 101 and 102.

The response is likewise simple in structure. As shown in Figure 4.6, the payload or more specifically the PDU, only consists of two fields: `Byte Count`, which specifies the length of the PDU in bytes and the actual data, i.e., the status of coils (True/False) as bits. However, as Modbus/TCP is byte-oriented, the the coil status is always transmitted as bits with a little Endian encoding. As illustrated in Figure 4.6, with the previous example of three requested coils, the least significant bit encodes the status of the coil at address 100, the 7^{th} bit address 101 and the 6^{th} bit address 102. The first 4 bits are not used and therefore 0. The interesting finding in practical evaluations are, that these unused bits can be safely used to establish covert channels, as the tested Modbus client implementations only interpret the bits they have requested before. Even in Wireshark, the unused bits are only visible when inspecting a packet manually in binary mode. This channel has been tested with OpenPLC[3], ScadaBr[4], Node-Red-Contrib-Modbus[5] and Wireshark[6].

One specialty for this covert channel is that the available maximum capacity varies and is dependent on how many coils the client requests. The maximum of 7 bits per packet is available when the client requests only one coil. However, if the client requests 8 coils, there are not any unused bits in the response from the server, leaving no place for embedding at all. Therefore, the overall achievable bandwidth solely depends on the actual setup and behavior of the devices involved. This also implies that the embedder has to closely monitor the conversations between client and server to determine how many bits actually will not be used in the next response. Another limitation is that this covert channel is only applicable to the `ReadCoils` response packets, therefore, the direction is limited to passing hidden information from Modbus servers to clients. In the previous mentioned examples, coil amounts

[3] https://gitlab.com/openplcproject/openplc_v3
[4] https://sourceforge.net/projects/scadabr/
[5] https://flows.nodered.org/node/node-red-contrib-modbus
[6] https://www.wireshark.org/

between one and eight were used. Of course, a client might query more than 8 coils. Due to the byte-orientation the principle stays the same though, e.g., in case a client requests 84 coils, four bits in the last byte are unused. The amount of unused bits can be calculated with the following equation:

$$unused\_bits = (8 - requested\_coils) \ \% \ 8$$

The overall achievable bandwidth with this covert channel highly depends on the actual setup - a fact that is later validated in the evaluation comparing several datasets.

4.1.6.2 Modbus Storage Channel 2 (CC_{MB2}) - Unit ID Modulation

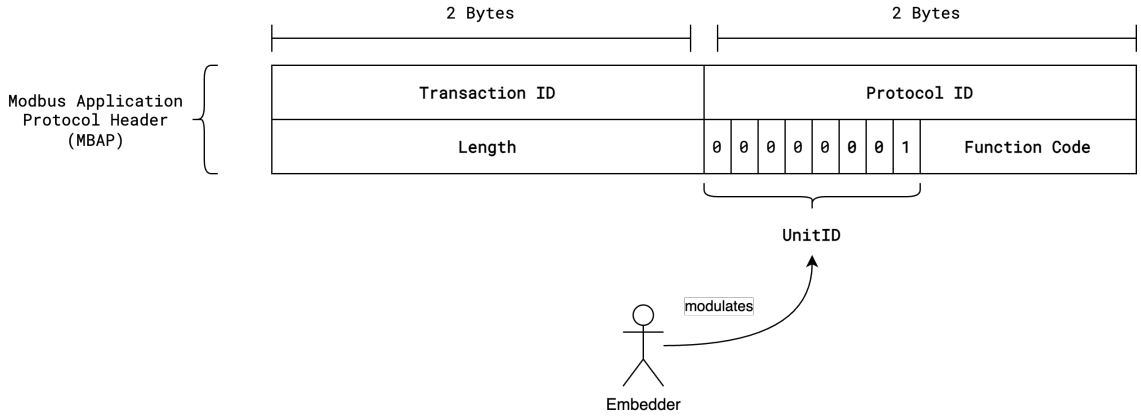


Figure 4.7: Illustration of UnitID Modulation in Modbus/TCP

As described before, the Unit ID field in the Modbus Application Protocol (MBAP) header, typically is only used when remote Slaves are addressed that are connected to the target devices. As this feature is rarely used but takes up one byte in the MPAB header, it is a plausible target to be leveraged as covert channel. The Unit ID defaults to `0x1`. This is observable for any of the previously described public datasets.

For the evaluation two encodings are implemented. The binary encoding uses `0xA` (decimal: 10) to encode a zero and `0xB` (decimal: 11) to encode a one and leaves the original UnitID the same when no part of the message is embedded. Therefore, when using this encoding the capacity is limited to one bit per packet. To overcome this issue, a second encoding is used where the eight bits of the hidden message are directly encoded in the UnitID. Using this encoding the capacity is raised to one byte per packet, eight times the capacity of the binary encoding. However, when looking at the resulting network captures the covert channel using this encoding is directly detectable as the UnitID changes with each packet. The binary encoding is far more unobtrusive as the selected UnitID might be used in the actual deployment.

4.1.6.3 Modbus Storage Channel 3 (CC_{MB3}) - Transaction ID (TID) Modulation

The Transaction ID is used to link Modbus/TCP responses from the server to actual requests from the client. For each request, the client will generate a transaction identifier that the server then uses in the response.

How the Transaction ID is generated depends on the implementation of the client. However, the specification defines the following [Mod21]:

"*The transaction identifier is used to associate the future response with the request. So, at a time, on a TCP connection, this identifier must be unique. There are several manners to use the transaction identifier: [..] For example, it can be used as a simple "TCP sequence number" with a counter which is incremented at each request. [..] It can also be judiciously used as a smart index or pointer to identify a transaction context in order to memorize the current remote server and the pending MODBUS request.*"

In the datasets the use of the transaction identifier varies. DS_2 is incrementing for for each request, with separate identifiers for each function code. DS_3 (DEFCON23) uses seemingly random transaction identifier, DS_4 (CRITIS18-1v2) uses mainly 0 and sometimes 1 as identifier. DS_5 (Lemay-run8) seems to be incrementing, however, not for every packet as there are many duplicates. In DS_1 the Modbus client uses only 8 bit for the transaction identifier (i.e. the range only goes from 0-255).

These differences in real-world deployments and datasets as well as the lack of strict specification open the door for the transaction id to be used as a carrier for hidden information and establishing covert channels.

For the implementation, an odd/even scheme is used to encode hidden information in the transaction identifier. To encode a *zero* the transaction identifier remains untouched if it is already even, else the identifier is reduced by one and vice versa for odd transaction IDs. In case of the linear increasing transaction IDs in DS_1 and DS_2, these modifications lead to partially duplicate IDs. DS_5 already contains many duplicates, therefore additional ones do not raise any suspicion without ground truth. In case of DS_3, which uses random transaction identifier, the modifications are fully inconspicuous. The achievable bandwidths using different parameters are evaluated in the following practical evaluation.

4.1.6.4 Modbus Storage Channel 4 (CC_{MB4}) - Padding

The padding covert channel is a simple yet very effective means of transferring hidden information in Modbus/TCP. The core idea is to add additional bytes to the payload of the padding **without** updating the length field in the Modbus/TCP header. This circumstance has one major observable advantage: In all conducted tests, Modbus clients and servers will ignore any padding after the actual payload and do not interpret the data appended. Moreover, as a side effect even Wireshark, Snort and Zeek do not interpret these data. This makes the covert channel much more inconspicuous, as one may expect. For example, to make the covert channel visible in Wireshark one has to inspect the network packets on byte or bit level (see right side of Figure 4.8). In the packet dissection pane the (covert) padding is not displayed on the Modbus layer. However, with the right knowledge, it is possible to see the data when looking at the Ethernet layer at the "Trailer" field (see the left side of Figure 4.8). In the implementation, the secret message is embedded directly and is retrieved from the padding.

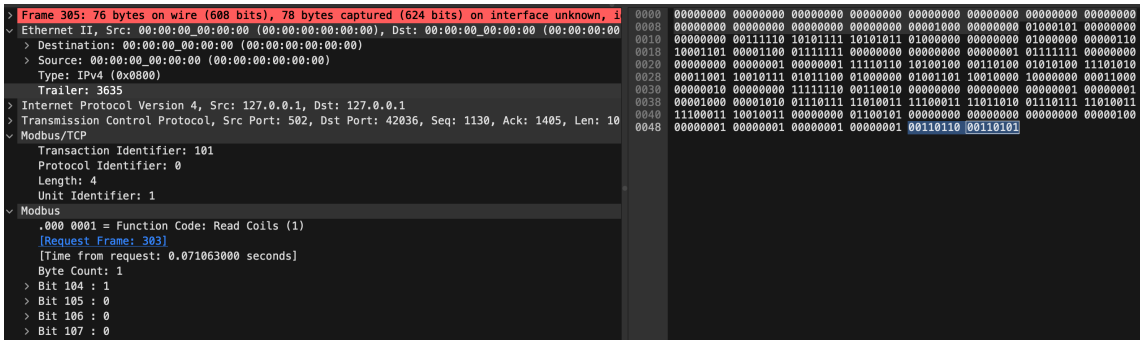


Figure 4.8: Screenshot of Wireshark, illustrating Modbus Storage Channel 4 (CC_{MB4}) - Padding.

4.1.7 Practical Covert Channel Evaluation

The selected covert channels are implemented in a custom framework that allows automated generation of data sets that contain covert channels with different sets of parameters. To be able to directly compare the performance of covert channels an *offline*-approach is taken, i.e. hidden information is injected in network capture files (pcaps) and not in live traffic. The alternative approach would be an online approach, which would manipulate network traffic on the fly. However, this would greatly impact comparability and reproducibility as network traffic might differ from run to run. Moreover, the offline approach allows for using public available pcaps of different environments, allowing for more confidence regarding generalization of results. The framework is designed in a modular way, so that multiple covert channels can be quickly implemented with as little overhead as possible while also enabling diverse parameterization. The tool takes a list of pcap or pcapng files as input and produces one pcap per covert channel/parameterization combination (representing the stego key). This allows for direct comparison between covert channels as well as comparison between different parametrizations for one specific covert channel.

Table 4.5 gives an overview on the four implemented covert channels ($CC_{MB1\text{-}4}$) and their parametrization. Each covert channel has three to eight configurations (set of parameters) that are performed on the six datasets DS_{1-6}. In total, this leads to 132 generated pcacps containing different parametrized covert channels.

The process of this procedure is illustrated in Figure 4.9. The stego-key in this case is the combination of parameters. For this evaluation, the covert channels are implemented using three parameters for *cover selection*, *bandwidth modulation* and *capacity variation*:

Packet Filter (F_i) selects only specific Modbus packets for embedding based on its **type**. Here, three filters are used: F_1 uses *all Modbus packets*, F_2 uses only *read coils* packets, F_3 uses all *read and write* packets for *coils and holding registers*. This is a method for *cover selection*.

Capacity (Cap) defines the amount of hidden bits embedded per cover object. In case of CC_{MB1} (Unused Bits) dynamically the maximum available unused bits

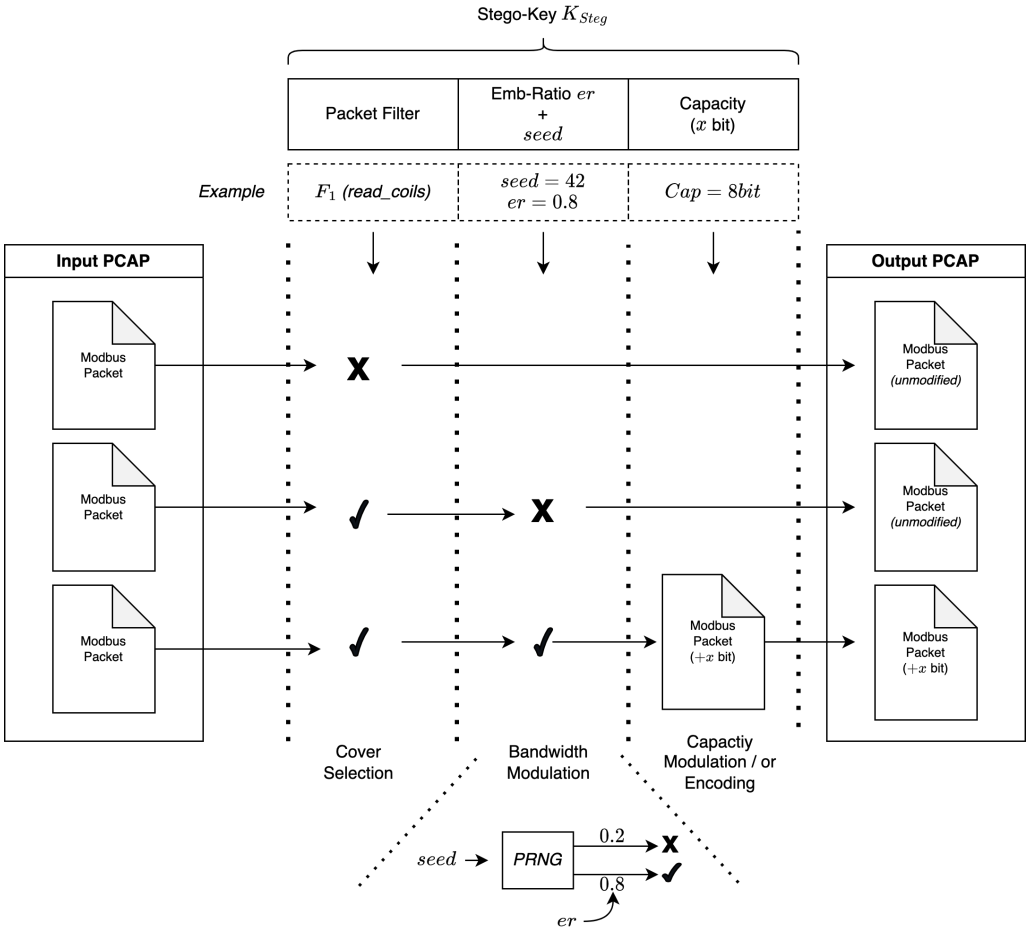


Figure 4.9: Illustration of the Offline Evaluation Framework.

are used for embedding, CC_{MB3} (TID) uses always one bit, and CC_{MB2} (UnitID) has two different encodings providing either 1 or 8 bit per packet. CC_{MB4} (Padding) can vary between 8, 16, 24 and 32 bit per packet.

Embedding Ratio (er) is implementing a probabilistic approach for *bandwidth modulation* by selecting Modbus packets for embedding using a pseudo-random selection method with a seed derived from the stego-key (see Figure 4.9).

The Offline Evaluation Framework has three major processes:

1. **Embedding** Secret messages are embedded using four covert channels with different parameters across multiple datasets.

2. **Retrieval** The embedded messages are retrieved using the stego key K_{Steg} and in some cases markers where information was injected. The retrieved message is checked for integrity.

Table 4.5: Overview on parametrization for the implemented Covert Channels $CC_{MB1\text{-}4}$. Each covert channel has four parameters: the (Packet-)*Filter* selects *type* of Modbus packets, *Capacity* directly sets the amount of hidden per cover and *Embedding ratio (er)* defines the probability a cover is used for embedding (bandwidth modulation). In total 132 different parameterized datasets were generated. Filter: F_1 uses *all Modbus packets*, F_2 uses only *read coils* packets, F_3 uses all *read and write* packets for *coils and holding registers*.

	Filter	Capacity	Emb.-Ratio (er)	Configs	DS	Output DS
CC_{MB1}	$F_2(read\_coils)$	*max. avail.*	$\{0.2, 0.5, 0.8, 1.0\}$	4	DS_{1-6}	24
CC_{MB2}	$F_1(all)$	$\{1, 8\}$	$\{0.2, 0.5, 0.8, 1.0\}$	8	DS_{1-6}	48
CC_{MB3}	$\{F_1, F_2, F_3\}$	1	1.0	3	DS_{1-6}	18
$CC_{MB4\text{-}1}$	$F_1(all)$	$\{8, 16, 24, 32\}$	0.8	4	DS_{1-6}	24
$CC_{MB4\text{-}2}$	$F_1(all)$	8	$\{0.2, 0.5, 1.0\}$	3	DS_{1-6}	18
total				22	6	132

3. Performance Measurement a log file is written containing

- the covert channel,
- used dataset,
- corresponding parameters (Cap, F_i, er),
- amount of modified cover objects,
- amount of embedded bits.

Calculated from the retrieval process are further metrics:

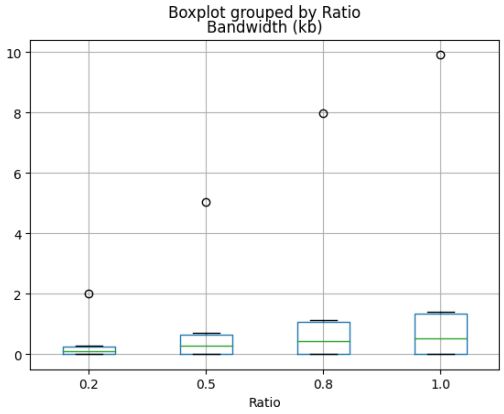
- robustness (relative amount of correct retrieved bits)
- bandwidth (per dataset)
- average capacity per cover object

For the practical evaluation the covert channel, dataset, parameters, robustness, bandwidth and average capacity are of special interest. These metrics are used for the following comparison and evaluation. The results of this evaluation are discussed separately for each covert channel.

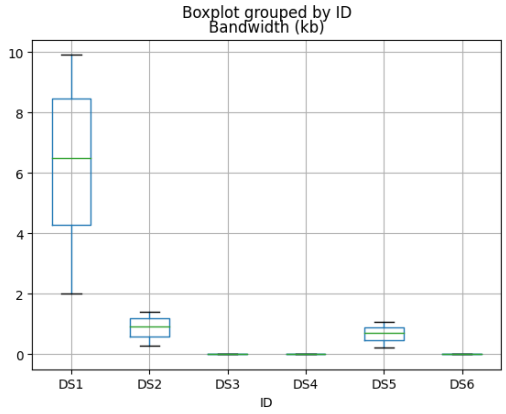
4.1.7.1 Evaluation of CC_{MB1} - Unused Bits

The implementation of this channel calculates the amount of unused bits for each ReadCoilsResponse packet based on the queried amount of coils and uses the maximum amount for embedding. To still be able to control the bandwidth, this covert channel makes use of the *embedding ratio (er)* parameter for cover (packet-) selection and bandwidth modulation. The embedding ratio er defines the relative amount of packets used for embedding. This is implemented using a weighted random-selection method. For example, with an embedding ratio of $er = 0.2$ every ReadCoilsResponse packet has a 20% chance of being selected for embedding (see also Figure 4.9 for

Figure 4.10: Boxplot comparison of the achievable bandwidth with Covert Channel CC_{MB1} - Unused bits in the different datasets and parameters for embedding ratio.



(a) Comparison of the effect of embedding ratio aggregated across all datasets on bandwidth (in kilobyte) for Covert Channel CC_{MB1} - Unused Bits.



(b) Boxplot comparison of the achievable bandwidth with Covert Channel CC_{MB1} - Unused bits in the different datasets.

Table 4.6: Comparison of average capacity and bandwidth in case of CC_{MB1} (Unused Bits) across all datasets. DS_3, DS_4 and DS_6 are omitted, since the covert channel was not applicable in these datasets.

	Capacity (avg.)				Bandwidth (bits)			
er	0.2	0.5	0.8	1.0	0.2	0.5	0.8	1.0
DS_1	4.45	4.40	4.42	4.40	16,061	40,246	63,784	79,398
DS_2	3.0	3.0	3.0	3.0	2,226	5,583	9,063	11,373
DS_5	4.0	4.0	4.0	4.0	1,676	4,284	6,824	8,544

reference). This directly affects the bandwidth, as shown in Figure 4.10a and Table 4.6.

Figure 4.10a illustrates the bandwidth spectrum across all data sets with the embedding ratio set to $0.2, 0.5, 0.8$ and 1.0 (using all ReadCoilResponse packets). As expected the bandwidth increases linear with the embedding ratio. However, for each step of the embedding ratio there is exactly one outlier. When looking at Figure 4.10b it becomes clear that the data set DS_1 is responsible for these outliers. As depicted in the figure, DS_1 achieves between 2 and 10kb whereas all other datasets range between 0 and around 1.8kb. This is due to the circumstance, that the custom dataset generation testbed (see Section 4.1.4) is specifically designed with this covert channel in mind. Therefore, the resulting dataset DS_1 contains much more ReadCoil packets than the other datasets (see Table 4.3). Moreover, as shown in Table 4.6 DS_1 does provide the highest capacity per cover on average as well. In contrast to DS_2 and DS_5, in DS_1 there are many more ReadCoilRequests with only one requested coil, leading to 7 unused bits, which ultimately leads to higher capacity on average. For DS_2 the average capacity is 3.0 and for DS_5 4.0. These values

are static because the amount of requested coils does not vary. DS_3, DS_4 and DS_6 are omitted in this comparison, since they do not include any ReadCoilRequests with unused bits (DS_4 and DS_6 do not contain ReadCoils at all, DS_3 has only 171 ReadCoil packets without any unused bits).

The robustness across all datasets is 1.0, i.e., every bit was transmitted and received correctly. In summary, the evaluation shows, that this covert channel is highly dependent on the dataset, yet can achieve high bandwidths (between 2kb/h and 10 kb/h) based on the amount of requested coils in the environment/dataset. Therefore, in attack scenarios the information hider will use prior reconnaissance to get an estimate on the potential bandwidth to calculate if this is sufficient for the attack (or required communication, respectively).

4.1.7.2 Evaluation of CC_{MB2} - UnitID

Table 4.7: Comparison of achievable bandwidth (kilobyte per dataset and kilobyte per hour) in case of CC_{MB2} (UnitID).

(a) Comparison of achievable bandwidth (kilobyte per dataset and kilobyte per hour) in case of CC_{MB2} (UnitID) with **1 bit** encoding (Capacity fixed at $Cap = 1bit$ and Filter $F_1(all)$.).

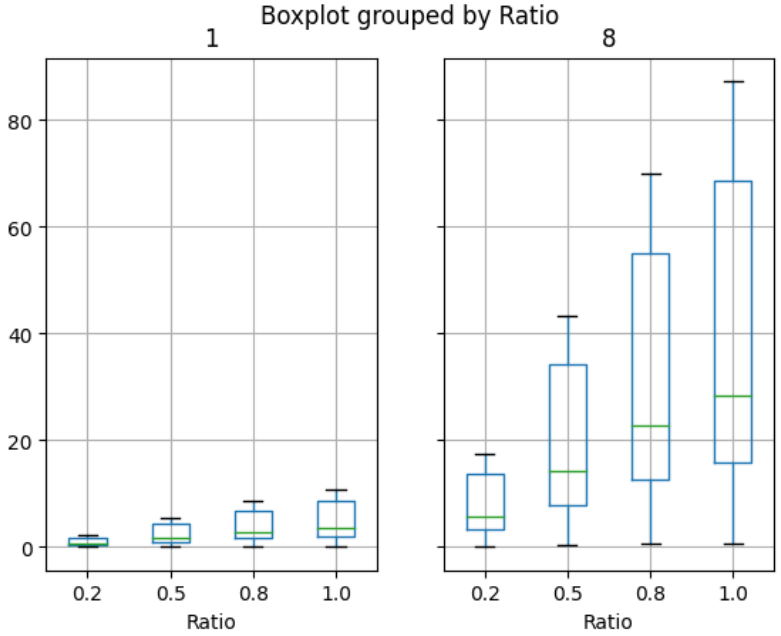
	Bandwidth (kb)				Bandwidth (kb/h)			
er	0.2	0.5	0.8	1.0	0.2	0.5	0.8	1.0
DS_1	2.01	5.08	8.08	10.10	2.01	5.08	8.08	10.10
DS_2	0.81	2.02	3.62	4.08	0.34	0.85	1.37	1.71
DS_3	0.02	0.04	0.07	0.09	0.003	0.008	0.013	0.016
DS_4	0.61	1.51	2.43	3.03	0.62	1.53	2.45	3.08
DS_5	0.33	0.82	1.31	1.63	0.33	0.82	1.31	1.63
DS_6	2.17	5.46	8.73	10.91	43.33	109.23	174.52	218.21

(b) Comparison of achievable bandwidth (kilobyte per dataset and kilobyte per hour) in case of CC_{MB2} (UnitID) with **8 bit** encoding (Capacity fixed at $Cap = 8bit$) and Filter $F_1(all)$).

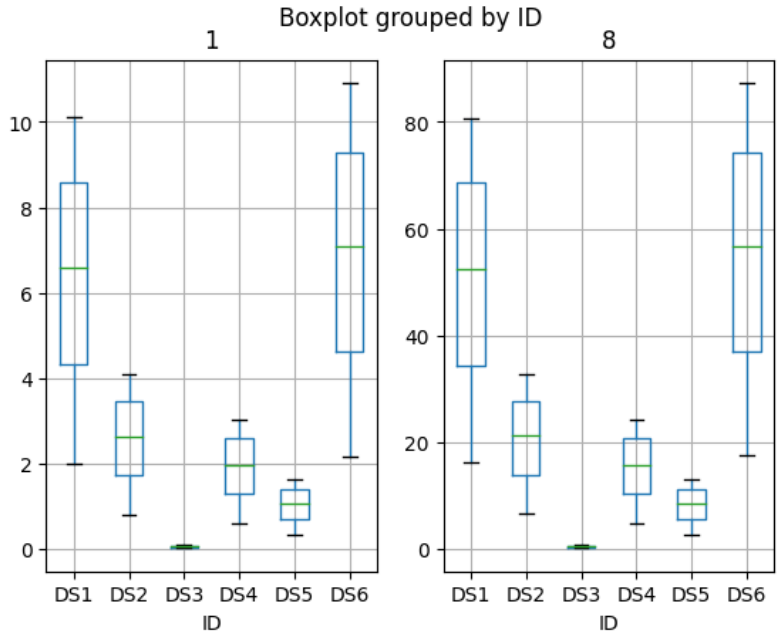
	Bandwidth (kb)				Bandwidth (kb/h)			
er	0.2	0.5	0.8	1.0	0.2	0.5	0.8	1.0
DS_1	16.17	40.37	64.70	80.77	16.17	40.37	64.70	80.77
DS_2	6.51	16.30	26.07	32.62	2.73	6.84	19.94	13.69
DS_3	0.13	0.36	0.57	0.71	0.02	0.06	0.10	0.13
DS_4	4.83	12.09	19.42	24.25	4.91	12.29	19.75	24.66
DS_5	2.66	6.56	10.45	13.02	2.70	6.67	10.63	13.24
DS_6	17.43	43.33	69.86	87.28	348.52	866.5	1,387.14	1,745.62

As shown in 4.11a, the bandwidth increases linearly with the embedding ratio *er*. The robustness of this covert channel has been 1.0 for each dataset, that is, every

Figure 4.11: Boxplot comparison of the achievable bandwidth with Covert Channel CC_{MB2} - UnitID.



(a) Comparison of the effect of embedding ratio aggregated across all datasets on bandwidth (in kilobyte) for Covert Channel CC_{MB2} - UnitID. The left chart is with capacity set to 1 bit per packet, the right chart uses the 8 bit capacity encoding.

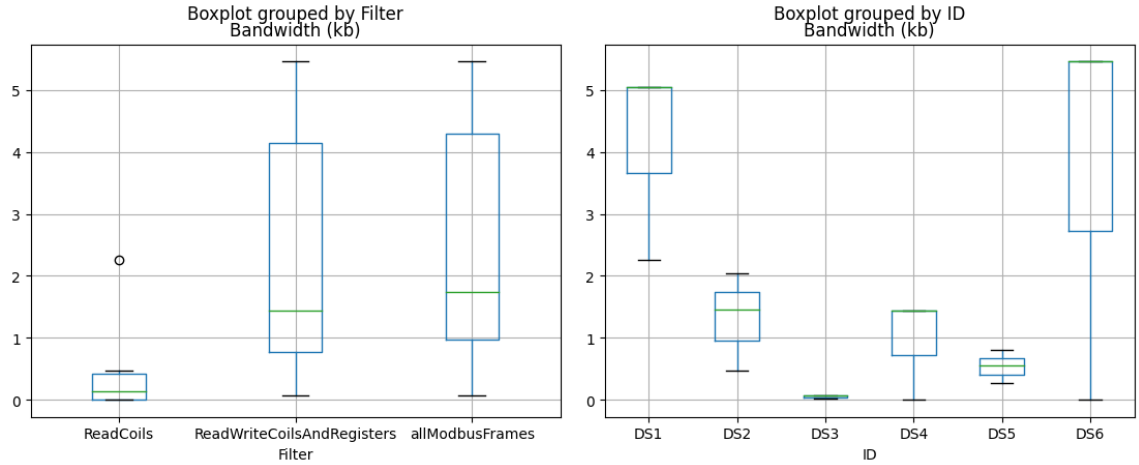


(b) Boxplot comparison of the achievable bandwidth with Covert Channel CC_{MB2} - UnitID using different parameters across the different datasets. The left chart is with capacity set to 1 bit per packet, the right chart uses the 8 bit capacity encoding.

covert bit could be successfully transmitted and retrieved without errors, loss, or uncertainty. Therefore, the embedder can directly optimize bandwidth against detection probability using the embedding ratio without loss of robustness. Also shown in this figure is the difference between the one bit encoding ($cap = 1$) in comparison to the eight bit encoding ($Cap = 8$). As expected, the resulting bandwidth increases in the same relation as the capacity. As shown in Table 4.7a (1bit encoding) and Table 4.7b (8bit encoding), the maximum bandwidth is achieved in dataset DS_6 with $10.91kb$ using $cap = 1bit$ and $87.28kb$ using $cap = 8bit$. The highest bandwidths could be achieved with DS_1 and DS_6. This is due to the absolute number of Modbus packets containing the UID which is used as carrier. Therefore, the amount of available Modbus packets is the single determining factor for maximum achievable bandwidth. The eight bit variant provides greater bandwidth, but it is also more noticeable and easier to detect.

4.1.7.3 Evaluation of CC_{MB3} - TransID

Figure 4.12: Comparison of the effect of embedding filter and use of different datasets on bandwidth (in kilobyte) for Covert Channel CC_{MB3} - TransID.



(a) Comparison of the effect of embedding filter aggregated across all datasets on bandwidth (in kilobyte) for Covert Channel CC_{MB3} - TransID.

(b) Boxplot comparison of the achievable bandwidth with Covert Channel CC_{MB3} - TransID using different parameters across the different datasets.

The bandwidth modulation for CC_{MB3} TransID is achieved by using only specific types of Modbus frames using the previously described Filters F_1 (all Modbus frames), F_2 (readCoils frames) and F_3 (only frames to read or write coils/registers). As shown in Figure 4.12a and Table 4.8, F_1 (all Modbus frames) and F_3 (only frames to read or write coils/registers) achieve nearly the same results: The minimum and maximum achievable bandwidth is identical and only the mean value is slightly higher in case of F_1. Whereas F_1 and F_3 can achieve up to $109kb/h$ in case of DS_6 (see the last row of Table 4.8 for reference) $F2$ achieves only up to $2.26kb/h$ in case of DS_1. This is due to the circumstance that DS_1 has been designed to incorporate as many readCoils packets as possible, whereas in public datasets DS_{2-6} these types of packets do not occur very often. When comparing the data sets in terms of achievable bandwidth using CC_{MB3}, DS_1 and DS_6 perform

Table 4.8: Comparison of achievable bandwidth (kilobyte per dataset and kilobyte per hour) in case of CC_{MB3} (TransID) Filters $F_1(all)$, $F_2(readCoils)$, $F_3(ReadWriteCoilsAndRegisters)$.

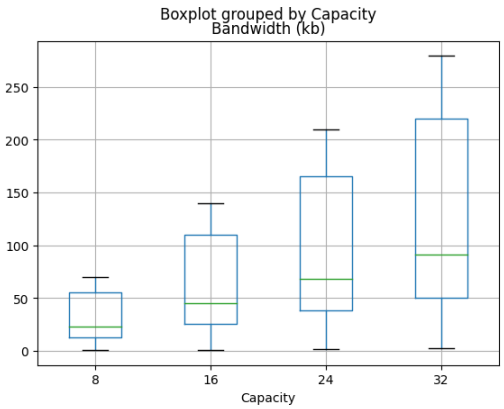
	Bandwidth (kb)			Bandwidth (kb/h)		
Filter	F_1	F_2	F_3	F_1	F_2	F_3
DS_1	5.05	2.26	5.05	5.05	2.26	5.05
DS_2	2.04	0.47	1.45	0.86	0.12	0.61
DS_3	0.07	0.02	0.07	0.13	0.004	0.13
DS_4	1.44	0.00	1.44	1.46	0.00	1.46
DS_5	0.81	0.27	0.55	0.83	0.03	0.56
DS_6	5.46	0.00	5.46	109.10	0.00	109.10

the best with the maximum bandwidth $5.05kb$ for DS_1 and $5.46kb$ in DS_6 (see the first three columns of Table 4.8). Subsequently follows DS_2 ($2.04kb$), followed by DS_4 ($1.44kb$). The least performing data set is DS_5, which achieves only $0.81kb$.

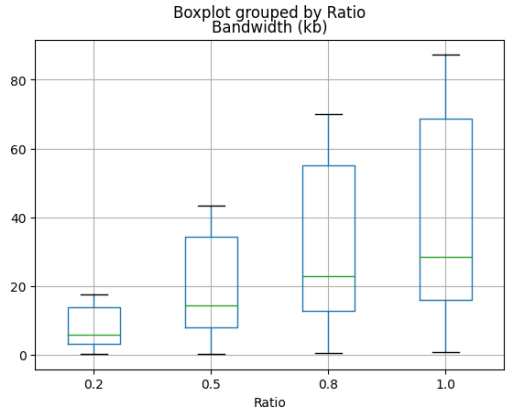
Again, the evaluation of these datasets highlights the impact of different configurations and environments on the actual performance of covert channels.

4.1.7.4 Evaluation of CC_{MB4} - Padding

Figure 4.13: Comparison of the effect of embedding capacity *cap* against embedding ratio *er* aggregated across all datasets on bandwidth (in kilobyte) for Covert Channel $CC_{MB4\text{-}1}$ - Padding.



(a) Comparison of the effect of embedding capacity aggregated across all datasets on bandwidth (in kilobyte) for Covert Channel $CC_{MB4\text{-}1}$ - Padding using a static embedding ratio *er* of 0.8.



(b) Comparison of the effect of embedding ratio aggregated across all datasets on bandwidth (in kilobyte) for Covert Channel $CC_{MB4\text{-}2}$ - Padding using a static capacity of 8 bit.

The evaluation of CC_{MB4} (Padding) is split into two subchannels: $CC_{MB4\text{-}1}$ uses a static embedding ratio $er = 0.8$ and modulates the bandwidth using four different configurations for capacity $8bit, 16bit, 24bit, 32bit$. This translates into a selection of

Table 4.9: Comparison of achievable bandwidth using CC_{MB4} (Padding) with fixed capacity ($CC_{MB4\text{-}1}$) against using a fixed embedding ratio ($CC_{MB4\text{-}2}$).

(a) Comparison of achievable bandwidth (kilobyte per dataset and kilobyte per hour) in case of $CC_{MB4\text{-}2}$ (Padding) with **8 bit** encoding (Capacity fixed at $Cap = 8bit$ and Filter $F_1(all)$).

	Bandwidth (kb)				Bandwidth (kb/h)			
er	0.2	0.5	0.8	1.0	0.2	0.5	0.8	1.0
DS_1	16.28	40.49	64.74	80.77	16.28	40.49	64.74	80.77
DS_2	6.61	16.28	26.12	32.62	2.77	6.83	10.96	13.69
DS_3	0.14	0.35	0.57	0.71	0.025	0.061	0.101	0.126
DS_4	4.82	12.22	19.43	24.25	4.90	12.43	19.76	24.66
DS_5	2.61	6.60	10.40	13.02	2.65	6.71	10.58	13.24
DS_6	17.45	43.49	70.04	87.28	348.94	869.9	1400.88	1745.62

(b) Comparison of achievable bandwidth (kilobyte per dataset and kilobyte per hour) in case of $CC_{MB4\text{-}1}$ (UnitID) using a static embedding ratio $er = 0.8$ and Filter $F_1(all)$).

	Bandwidth (kb)				Bandwidth (kb/h)			
cap	8	16	24	32	8	16	24	32
DS_1	64.74	129.19	193.8	258.07	64.74	129.19	193.8	258.07
DS_2	26.12	52.17	78.53	104.15	10.96	21.89	32.95	43.70
DS_3	0.57	1.16	1.68	2.36	0.10	0.21	0.30	0.42
DS_4	19.43	38.76	58.25	77.58	19.76	39.42	59.24	78.89
DS_5	10.4	20.98	31.31	41.31	10.58	21.34	31.83	42.01
DS_6	70.04	139.80	209.35	279.51	1400.88	2795.96	4187.10	5590.24

80% of all suitable packets. Depending on the *cap* configuration, $CC_{MB4\text{-}1}$ appends 1 to 4 bytes to the Modbus frame. In contrast, $CC_{MB4\text{-}2}$ always adds one single byte ($cap = 8$) to the frame and modulates the bandwidth using the embedding ratio parameter *er*. Table 4.9 describes the difference in achievable bandwidths across the different parameters and datasets. As illustrated in Figure 4.13a, using a static embedding ratio ($er = 0.8$) the bandwidth increases linearly with increasing capacity. However, between datasets, the achievable bandwidth does vary significantly, for example, using $cap = 8$, DS_6 achieves over $70kb$ whereas DS_3 reaches only $0.57kb$. Again, this is simply due to the amount of Modbus frames in the respective datasets. The more Modbus frames are available the more can embedded. These differences are clearly visible in the bar charts of Figures 4.13a and 4.13b. The robustness for all configurations is 1.0 i.e., no hidden data was lost. Overall, and compared to the other covert channels, CC_{MB4} (Padding) provides consistenly both most capacity and bandwidth across all datasets. As this covert channel is applicable to each Modbus frame, independent of direction, performable by clients, servers and intermediate nodes, it is the most versatile covert channel in this selection. In a passive hiding senario (*man-in-the-middle*-like) it can also be used in a reversible

fashion, for example, if intermediate nodes remove the padding from the original packet before forwarding it to its actual destination.

4.1.7.5 Performance Comparison of Covert Channels $CC_{MB1\text{-}4}$

In the previous sections, each covert channel was evaluated separately. Here, the overall performance of the covert channels shall be compared. Before doing so, it is important to reflect on the performance of the parameters first, as these have a drastic influence on the actual performance.

In general, the three parameters, *Packet Filters* (F_{1-3}), *Embedding-Ratio* (er) *and Capacity* (cap) worked in all cases as expected.

The capacity parameter showed the most direct influence on the resulting bandwidth as it affects each object used for embedding. For example, by switching from a 1 bit capacity to an 8 bit capacity, the resulting bandwidth is eight times higher as well. The exception to this rule, is CC_{MB1} (Unused Bits) as it is designed to use the maximum achievable capacity for each single packet.

The embedding ratio er and packet filter F_i can be used independently of the capacity parameter to modulate the bandwidth using cover selection.

In contrast to the capacity parameter, the embedding ratio and packet filter work by *reducing* the maximum amount of bandwidth by a certain amount, mainly to decrease detectability and beeing more inconspicious. The embedding ratio er showed the most consistent results and allows for granular modulation of the bandwidth. Especially the pseudo-random selection of covert objects make detection much more difficult.

The packet filters are highly dependent on the dataset used. This is due to the large differences in frequency of certain packet types between datasets (for reference, see the comparison of the datasets in Table 4.3). For example, using a filter that uses only ReadCoils-packets for embedding is completly ineffective if there are not any ReadCoils packets in the dataset. Therefore, when using packet filters it is more difficult to estimate the effect on the bandwidth and therefore should be used more carefully than, for example, the embedding ratio. Still, the packet filter seems a viable option when hiding in specific network packets either to make detection less likely or to be used as part of the stego-key.

In summary, the parameters provided enough means to modulate the bandwidth required for most attack scenarios in which the bandwidth is adapted to become more stealthy.

In terms of comparison of the performance of the covert channels, some similarities and differences can be identified. Starting with similarities, each covert channel shared the same significant differences in achievable bandwidths between datasets DS_{1-6}. As a general rule of thumb, for the implemented storage channels, the number of carrier frames (i.e. *quantity* of cover objects) is the most significant factor for the resulting bandwidth. The *quality* of the cover object instead does not seem to be a major factor (with respect to bandwidth) for most of these selected covert channels. On the contrary, timing channels, for example, are far more dependent

on the quality of the carrier, e.g., the consistency of interpacket times. This is not the case for the selected covert channels in this study. However, there is one exception to this, which is Covert Channel CC_{MB1} - Unused Bits. In case of CC_{MB1} (Unused Bits) both the number (i.e., *quantity*) of ReadCoils-requests as well as the field which describes how many coils are requested in this particular request (that is, the *quality* of the respective carrier object) have a direct impact on the overall achievable bandwidth. However, this is only due to implementation, as the configuration of CC_{MB1} in the offline framework maximizes the capacity for *each cover object individually* and therefore affects the bandwidth. When using a static capacity instead, the results would be similar compared to the other covert channels.

Table 4.10: Comparison of performance of the selected Modbus/TCP Covert Channels $CC_{MB1\text{-}4}$ in terms of minimum and maximum Capacity, Bandwidth, Robustness and Stealthiness.

	Capacity		Bandwidth		Robustness		Stealthiness	
	min.	max.	min.	max.	min.	max.	min.	max.
CC_{MB1}	1 bit	7 bit	0.00 kb	9.92 kb	1.0	1.0	*high*	*high*
CC_{MB2}	1 bit	8 bit	0.02 kb	87.28 kb	1.0	1.0	*low*	*high*
CC_{MB3}	1 bit	1 bit	0.00 kb	5.46 kb	1.0	1.0	*low*	*high*
CC_{MB4}	8 bit	32 bit	0.14 kb	279.51 kb	1.0	1.0	*high*	*high*

In terms of robustness all four covert channels showed perfect results, i.e., every hidden bit could be retrieved successfully. This is due to fact that only storage channels were used in this investigation which do not experience alterations in the offline evaluation framework as this does not include any noise factors that would affect robustness. Arguably, this is a limitation of the evaluation framework. To realistically evaluate the robustness of the selected covert channels, these would have to be tested in real-world (live) deployments which include intermediate nodes and wholistic processes that include noise added to network packets. However, this would be more complex and requires far more resources in terms of hardware, software and especially (computing) time. As the aim of the offline framework is to have a solid foundation to be used for rapid prototyping and parametrization of covert channels, testing for robustness is not a focus in its design. Testing for robustness requires a separate testing environment, which is out of scope for this particular evaluation due to the reasons mentioned before. Such testing seems advisable for future investigations to get a better understanding on the applicability of such covert channels in real-world scenarios.

When it comes to differences between covert channels, the most significant difference is the bandwidth. For example, when looking at the maximum achievable bandwidth as shown in Table 4.10, CC_{MB4} leads with $279.51kb$, followed far behind by CC_{MB2} with $87.28kb$ (equivalent to $\approx 31\%$ of CC_{MB4}). After CC_{MB2}, follows CC_{MB1} with only $9.92kb$ which is approximatly only $\approx 3.55\%$ of the bandwidth that is achievable with CC_{MB4}. The last is CC_{MB3} with a maximum of 5.46 which is about $\approx 1,95\%$ relative to CC_{MB4}.

In terms of minimum capacity, CC_{MB1} and CC_{MB3} are outliers in the evaluation data as some dataset/filter combinations can lead to the situation that there are no suitable carrier objects left that could be used for embedding. In general though, the parameters *Filter*, *Embedding Ratio* and *Capacity* can successfully modulate the bandwidth to the exact amount the information hider requires, e.g., to decrease probability of detection.

Another difference can be found in stealthiness. Overall, each covert channel can achieve high stealthiness in at least one dataset using a parametrization that utilizes low capacities, strong packet filters, and low embedding-ratios to make detection less likely. However, under certain conditions, covert channels CC_{MB2} (UnitID) and CC_{MB3} (TransID) are easier to detect, though. For example, when using Covert Channels CC_{MB2} (UnitID) with maximum capacity ($cap = 8$), maximum embedding ratio ($er = 1.0$) and Filter F_1 (all), *every* UnitID of each Modbus packet gets exchanged for a piece of hidden information, i.e., each request is using a different UnitID whereas usually only one to three UnitIDs are used. This is very obtrusive and, therefore, easy to detect. A similar case can be found with CC_{MB3} (TransID) in the DS_4 (CRITIS18-1v2) dataset. In this dataset, most Modbus frames use the same Transaction ID ($0x00$) with only a few exceptions of packets that use ($0x01$) as the identifier. With this knowledge, a detection based on the distribution of TransIDs is easy to achieve.

The proper detection and mitigation of these covert channels is discussed next in Section 4.1.8.

4.1.8 Detection & Mitigation

In this section, targeted methods are described to detect and mitigate the threat posed by previously evaluated Covert Channels CC_{MB1-4}. Following Kerckhoffs' principle, it is assumed that the warden who is running the detection algorithm is in full knowledge of the existence and functionality of the previously described covert channels except for the stego-key (which in this case is the parameterization: Embedding Ratio er, Filter F_i, Capacity cap). That means, the warden knows that information is embedded but does not know which covert channel is used, what the covert objects are and how much is embedded.

4.1.8.1 Detection of CC_{MB1} (Unused Bits)

The key to the detection of CC_{MB1} (Unused Bits) is to inspect each pair of ReadCoilsRequest/Response packets to verify the number of requested coils against the number of bits in the response. For example, if two coils are requested, there should also be only two bits set in the response packet. Any true bit in the remaining unused bits is an indicator (in malware-terms *Indicator-of-Compromise, IoC*) for this covert channel as the remaining bits should be zero-filled. From this observation it becomes apparent that a *stateful* inspection method on the application layer (Modbus) is required, i.e. there has to be some kind of memory that can hold the number of requested bits of the last request and then compare it against the actual number of responded bits. Therefore, rule-based systems like *Snort* and *Suricata* seem not sufficient to detect this channel. The next more powerful option is to use a script-based detection system. For a proof of concept, *Zeek* is chosen as the tool of choice, as it provides a powerful scripting language and extensive Modbus/TCP parsers using the Industrial Control Systems Network Protocol Parsers (ICSNPP)[7] published by the US Cybersecurity and Infrastructure Security Agency (CISA). Listing 4.1 shows a simplified version of the detection algorithm in Zeeks native scripting language. Zeek uses an event-based system. In Listing 4.1 two events are hooked which basically represent the two steps mentioned above: (1) in case of a ReadCoilsRequest packet the number of requested coils is saved in a variable. (2) in case of a ReadCoilsResponse packet, the algorithm checks for any true bits in the unused bit section.

Table 4.11 shows the detection rates for each usable dataset with different parameterizations for the embedding ratio er. Figure 4.14 illustrates the difference in detection rates between the datasets.

The detection algorithm performed best on DS_1 (*Generated Dataset*) with an arithmetic mean of 99.61% *true-positive* detections of cover objects with hidden information, followed by DS_5 (*Lemay-run8*) with a mean of 98.83% and last DS_2 (*Cyberville*) with a mean of 93.04%. In all cases, the *false-positive* rate was zero.

Unexpectedly, the embedding ratio only marginally affects the detection rate. Instead of an increasing detection rate with increasing embedding-ratio, the detection rate is stable with only minor fluctations ($\sigma \leq 0.076$ across all datasets).

[7] https://github.com/cisagov/ICSNPP

```
1
2  global requested_coils = 0;
3
4  event modbus_read_coils_request(c: connection,
5  headers: ModbusHeaders, start_address: count,
6  quantity: count)
7      {
8          requested_coils = quantity;
9      }
10
11 event modbus_read_coils_response(c: connection,
12 headers: ModbusHeaders, coils: ModbusCoils)
13 {
14     local unused_bits = (8 - requested_coils) % 8;
15
16     if (unused_bits != 0)  {
17         local true_bits = 0;
18         for ( index, coil in coils[-unused_bits:] )
19             {
20                 if (coil == T)
21                     true_bits += 1;
22             };
23         if (true_bits != 0)
24             {
25                 # Covert Channel identified
26             };
27     };
28 }
```

Listing 4.1: Simplified proof-of-concept detection algorithm for CC_{MB1} (Unused Bits) in Zeeks native scripting language.

Table 4.11: Detection rates in percentage of CC_{MB1} (Unused Bits) in Datasets DS_1, DS_2, DS_5 in context of the corresponding embedding ratio er. For better readbility the percentage symbol (%) is ommited.

		Embedding Ratio er					
		0.2	0.5	0.8	1.0	mean	σ
DS_1	*Generated Dataset*	99.50	99.66	99.75	99.54	99.61	0.0011
DS_2	*Cyberville*	91.91	93.39	93.34	93.53	93.04	0.0076
DS_5	*Lemay-run8*	98.81	98.88	98.94	98.69	98.83	0.0011

The other unexpected aspect is the difference between datasets. In case of DS_1 and DS_5 less than 2% of cover objects with hidden data were not identified, while in DS_2 the false negative rate lies between $6 - 8\%$.

From these results two questions arise: (1) Why does the embedding ratio not affect the detection rate? In theory, a lower embedding ratio should decrease the detection probability. (2) Why is the detection probability significantly lower in DS_2?

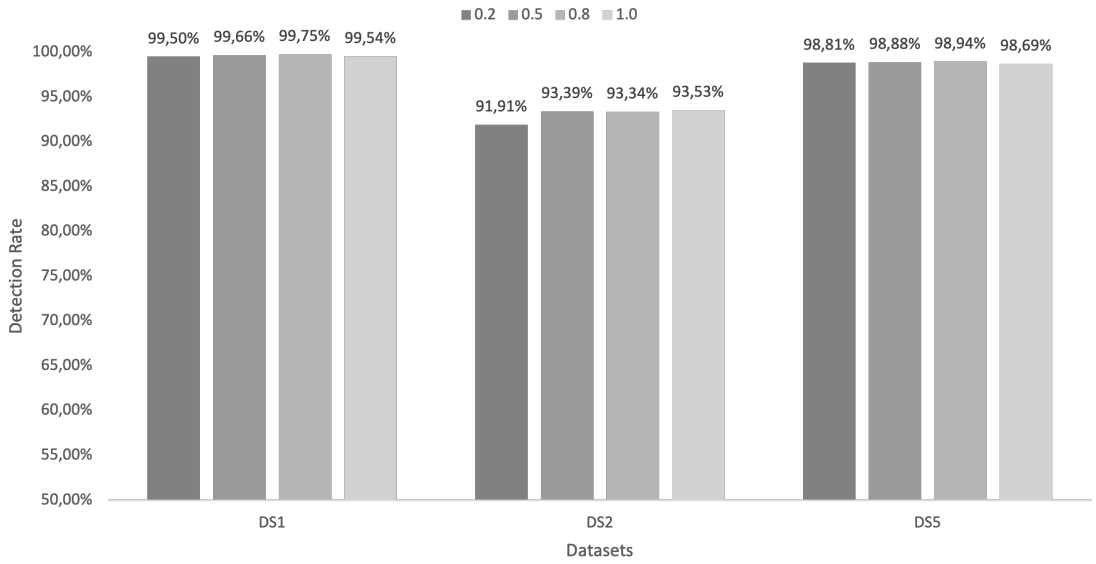


Figure 4.14: Detection rates in percentage of CC_{MB1} (Unused Bits) in Datasets DS_1, DS_2, DS_5 in context of the corresponding embedding ratio er.

The answer to the second question can be found in the dynamic capacity method used by this covert channel. For each packet, the embedding algorithm calculates the number of unused bits and uses all of them. As shown in Table 4.6, the capacity per cover object varies between the datasets.

DS_2 has the lowest capacity per cover object. This is due to the circumstance that in this particular datatset the Modbus client requests the status of 77 or 5 coils. This leaves 3 bits unused in the response from the server. In contrast, in DS_5 each time 4 coils are requested, leaving 4 bits unused.

With each additional bit used by the covert channel a detection is statistically more likely. This is due to the fact that the detection algorithm searches for any true bits (binary one) in the unused bits section (which should be binary zero). However, the detection algorithm cannot detect a part of a hidden message that consists only of zeros, as that resembles a regular packet with unused bits. Under the assumption that the hidden message is split across multiple packets and zeros and ones are equally distributed, in case of the 4 unused bits, the likelyhood of a non-detectable packet is $\frac{1}{2} * \frac{1}{2} * \frac{1}{2} * \frac{1}{2} = \frac{1}{16}$, i.e. statistically every 16th packet cannot be detected by the algorithm.

In case of DS_2 with only three unused bits the probability is significantly higher for an undetectable packet: $\frac{1}{2} * \frac{1}{2} * \frac{1}{2} = \frac{1}{8}$, that is, statistically every eighth packet is not detectable by the algorithm. Therefore, the detection rate is significantly lower for DS_2.

This circumstance explains the difference in detection rates between DS_2 and DS_5.

The case for DS_1 is similar. As shown in Table 4.6 the algorithmic mean capacity in DS_1 is between $4.40 - 4.45$ bits. This is due to the circumstance that in DS_1 either four coils are requested or only one coil. Therefore, the number of unused bits

varies between 4 and 7 bits. Especially packets with 7 unused bits have the highest detection probability $(1-(\frac{1}{2})^7 = 1-\frac{1}{128} \approx 0.992$. Due to this additional packet with seven unused bits, the detection rate in DS_1 is even higher than in DS_5.

The answer to the question, why the embedding ratio does not significantly impact the detection rate, is found in the detection algorithm. The detetion works by evaluating each single packet individually, looking for any true bits that should be false. Therefore, the number of packets that are used to embed the whole secret message does not affect the individual detection of packets containing hidden bits. As described before, *capacity* is deciding factor for the detection rate.

In summary, these results show that CC_{MB1} (Unused Bits) can be detected by Zeek reliably across multiple datasets and parametrizations.

4.1.8.2 Detection of CC_{MB2} (Unit ID)

Under the assumption that the warden is in knowledge of the legitimate Unit IDs for each dataset, CC_{MB2} (Unit ID) can be detected with 100% precision using rule-based systems simply by filtering packets that use a Unit ID that has not been previously whitelisted.

In case of the datasets used in this study, the Unit ID remains always $0x01$. Therefore, any deviation from this Unit ID is easy to spot and used as an indicator for CC_{MB2} (Unit ID).

One of the simplest ways for detecting this channel is to use wireshark. In wireshark one can use a filter to find any packets that have a Unit ID different from the default: `mbtcp.unit_id!=1`.

To automate this process in a proof-of-concept, again Zeek is used for better comparability, making use of a simple detection script that resembles common detection rules found in rule-based systems like Suricata and Snort. In case of the datasets used in this study, such rule-based systems are sufficient to detect this channel. A simplified version of detection script for Zeek is shown in Listing 4.2.

```
1
2  event modbus_message(c: connection, headers: ModbusHeaders,
3                                                 is_orig: bool)
4  {
5      local unit_id = headers$uid;
6      if (unit_id !in ModbusCovertChannels::allowed_unit_ids)
7          {
8              # Potential Covert Channel identified;
9          }
10 }
```

Listing 4.2: Simplified proof-of-concept detection algorithm for CC_{MB2} (Unit ID) in Zeeks native scripting language.

In all of these cases the detection script provided a 100% detection rate without any false positives. This his been validated with wireshark as well. This also holds true for both capacity variants with 1 bit ($cap = 1$) and 8 bit ($cap = 2$).

While in these datasets the Unit ID is static, in practice, a warden might face the situation that not only one, but several different Unit IDs are used. In that case, the information hider might use this cirumstance by chosing an embedding scheme (encoding) which only makes use of the legit Unit IDs to encode a message. In other words, the alphabet would be made out of the Unit IDs that occur in the target environment. In that case, the detection methods described above would not be sufficient and would require statistical means for detection. One approach would be to calculate the frequency distribution of occuring Unit IDs in the target environment in a specific time frame (window) when it is in a non-compromised state. An alternative approach would be to use a sliding window.

With this regular state, the warden can perform anomaly detection on the frequency distribution. Again, the proper time frame (or detection window) is vital for successful detection. Further tests are required to validate this approach.

4.1.8.3 Detection of CC_{MB3} (Trans ID)

As described above, the individual use of the Transaction Identifier varies between datasets: DS_2 (Cyberville) and DS_6 (chinese-ctf) are linearly incrementing, DS_3 (DEFCON23) uses seemingly random transaction identifier, DS_4 (CRITIS18-1v2) uses only 0 and 1, DS_5 (Lemay-run8) is incrementing, yet with many duplicates. In DS_1 (Generated Dataset) the TID is lineary incrementing from 0-255 (8 bit only) and starts over again. From this observation, there seems not to be one distinct detection method that suits all cases. In case of DS_4 (CRITIS18-1v2) any deviation from the default 0 or 1 can be identified using the same approaches as described above for CC_{MB2} (Unit ID) using a rule-based detection or using frequency distribution as a feature to detect anomalies. In case of DS_1, DS_2 and DS_6 which are linearly increasing, a script-based system like Zeek can be used to verify the linear progress of the transaction identifiers used. Any deviation from linearity would be an indicator for the covert channel.

In DS_3 (DEFCON23) and DS_5 (Lemay-run8) the covert channel is significantly more difficult to detect due to the randomness in the IDs. In DS_3 (DEFCON23) the Transaction ID usage appears to be random, rendering any of the prior discussed detection methods useless. Here, advanced Machine-Learning techniques might provide a way for detection.

DS_5 (Lemay-run8) is a special case as the TransIDs are incrementing, yet contain many duplicates. Potentially, an anomaly detection-based method using the frequency distribution of duplicate TransIDs might pave a way for proper detection.

Such deeper investigations into such advanced detection methods are out of scope for this thesis, yet provide a perspective for future research topics.

In general, CC_{MB3} (Trans ID) shows to be signifcanlty more difficult to detect due the differences experienced in different target environemts or datasets, respectivly.

4.1.8.4 Detection of CC_{MB4} (Padding)

CC_{MB4} (Padding) is an interesting case regarding detection. Although the covert channel is simple in its structure, common IDS systems like Snort and Zeek fail to

detect it due their ability to parse Modbus/TCP. What seems contradicting at first, is a loophole, created by the ability to properly dissect application layer protocols like Modbus. By parsing the packet and dissecting it, both Snort and Zeek fully ingore the padding as it does not belong to the Modbus frame. Therefore, when the detector has no direct way of detecting or accessing the padding with tools mentioned. This leaves a detection loophole for this specific covert channel using common open-source IDS systems. To circumvent this issue, the python library `scapy` is used for a proof-of-concept implementation. Scapy is a Python library, that provides several means for packet inspection, dissection and manipulation and is not affected by the limition mentioned above. It is also the same tool used to implement the covert channel itself.

The python code for detecting this channel is listed in Listing 4.3. The detection script looks for any paddings after the Modbus Application header. By this, in this evaluation a 100% detection accuracy is achieved across all datasets and parameterizations with zero false-positives.

In summary, with knowledge of this detection loophole for paddings after protocol parsings in common IDS systems, CC_{MB4} (Padding) is fairly easy to detect using a customized Python script utilizing open-source, public libraries.

4.1.8.5 Design of an Active Warden for Mitigation of Covert Channels $CC_{MB1\text{-}4}$

After the discussion of detection methods, another question is whether the covert channels can be prevented or mitigated without prior detection using an *active Warden*.

Indeed, Covert Channels $CC_{MB1\text{-}4}$ can be prevented or mitigated using a *protocol normalizer* acting as an *active Warden*:

In pure Modbus/TCP environments, the Unit ID is not required and therefore can be normalized to a specifc value, e.g. 1. This can be enforced by a protocol-normalizer acting as *man-in-the-middle* between Modbus server and clients (by that resembling an *active warden*). The normalizer, or warden overwrites the Unit ID in each packet, erasing any hidden information and by that countering CC_{MB2}. This can also be combined with a prior detection (i.e., derivation from the default ID) with the detection methods discussed before. The same procedure is possible for CC_{MB3} with the Transaction ID. The normalizer replaces the Transaction ID in each packet with a linear increasing counter. In case of CC_{MB1} (Unused Bits), the warden keeps track of the requested Coils. In the response from the server, the warden overwrites the unused bits with true bits. By using true bits, the warden can eliminate even hidden information that is encoded as zero (see discussion on detection of CC_{MB1} for details). However, if this leads to any conflicts, the warden can also write false bits (i.e. a binary zero) to the unused bit section.

CC_{MB4} (Padding) can be mitigated using the same method that is used for detection. If any padding is identified it can simply be removed from the packet before forwarding it to its original destination.

All of these methods used for mitigation can be implemented using scapy, the python library used to implement the covert channels.

```
1
2  from scapy.utils import PcapNgReader
3  from scapy.contrib.modbus import *
4  from scapy.packet import Padding
5
6  pcap = "modbus-pad-cc-sample.pcap"
7
8  byte_counter = 0
9  frame_counter = 0
10 contents = ""
11
12 with PcapNgReader(pcap) as frames:
13     for frame in frames:
14
15         # filter for modbus frames only
16         if frame.haslayer(ModbusADURequest) or frame.
               haslayer(ModbusADUResponse):
17
18             # check for some unusual padding
19             if frame.haslayer(Padding):
20                 # save its content for further analysis
21                 load = frame[Padding].load
22                 decoded_hex = load.decode("utf-8")
23                 contents += decoded_hex
24
25                 # save some stats
26                 byte_counter += len(load)
27                 frame_counter += 1
28
29 if frame_counter != 0:
30     print(f'Found {frame_counter} modbus frames with
           unusual padding.')
31     print(f'{byte_counter} bytes were identified.')
32     print(f'This makes an average {byte_counter /
           frame_counter} bytes per packet.')
```

Listing 4.3: Proof-of-concept detection algorithm for CC_{MB4} (Padding) using scapy.

In summary, Covert Channels $CC_{MB1\text{-}4}$ can be successfully prevented or mitigated using an active warden, that is capable of dissecting and manipulating Modbus/TCP frames between Modbus client and server.

4.1.8.6 Summary of Detection and Mitigation of Covert Channels $CC_{MB1\text{-}4}$

As the results discussed above show, in most cases a reliable detection and/or mitigation is possible for the investigated covert channels. In most scenarios, open-source rule-based or script-based IDS/IPS systems can be leveraged for successful detection. However, the results also show some loopholes as well as the significant impact of the target environment or dataset, respectively, on the acutal performance. In some cases, mainly depending on the target environment, statistic-based approaches like anomaly detection and further technologies of machine learning might be required for reliable detection. In a well-defined environment, together with the usage of protocol normalizer and wardens capable of properly dissecting the application layer protocol, all investigated covert channels can be prevented or detected. Overall, the results highlight the necessity of preparation against such covert channels, as without targeted measures, these covert channels would stay undetected.

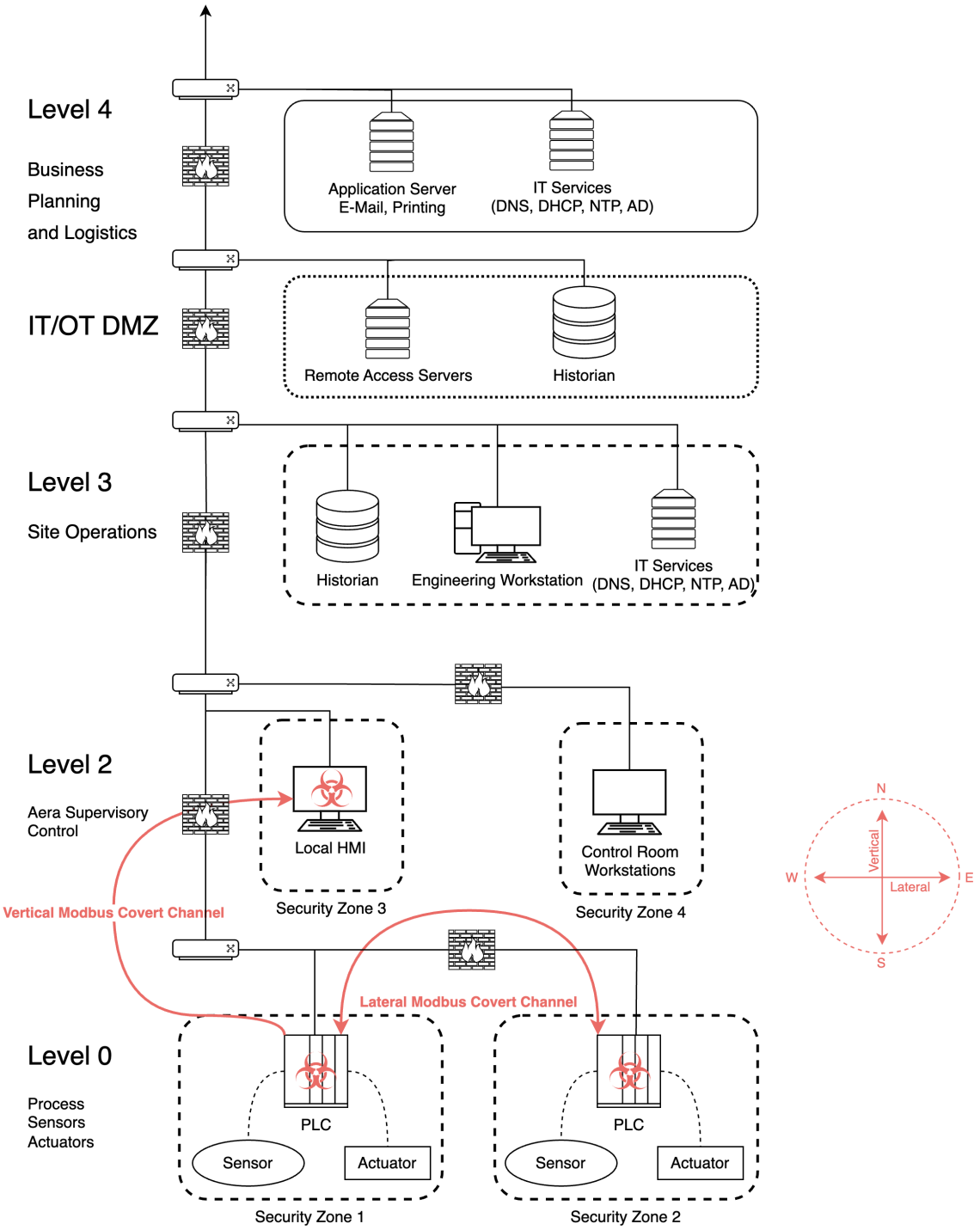


Figure 4.15: Potential threat scenario involving adversarial use of Modbus/TCP covert channels at the example of the Plausible Reference Architecture *PlauRA-3*.

4.1.9 Adversarial Application of Modbus/TCP Covert Channels

As Modbus/TCP is an OT-specific protocol, its use is limited mainly to PERA levels 1-3 (see Figure 3.1 for reference). Therefore, threat scenarios using Modbus/TCP mainly revolve around covert communication between compromised PLCs, HMIs and Engineering Workstations. Other components like protocol-converter and (Historian-) databases may play a role as well. In general though, covert communication seems mainly relevant for *covert lateral communication* in the same security zone (*intra-zone* or between security zones *inter-zone*). Depending on the architecture, the covert channels in Modbus/TCP could also be used to covertly move *vertically* from PERA Levels 0-2 to Level 3, which mainly incorporates Site Operations (see Figure 3.2). From there, the adversary might pivot into the IT-levels or vice versa using covert channels in other protocols.

4.1.10 Summary of Modbus/TCP Case Study

Modbus/TCP is a widely used non-proprietary ICS protocol that can be expected to be used in adversarial attacks within Operational Technology. This case study focuses on the evaluation of covert channels by applying the patterns of the Network Information Hiding Taxonomy to the specification of Modbus. A testbed is designed to generate realistic, representative, and reproducible datasets that are used as a baseline in combination with public datasets for the analysis of covert channels. From these identified covert storage channels, a subset is selected for implementation and further evaluation in realistic testbests and real-world datasets. The evaluation shows that the introduced covert channels are highly dependent on the environment, yet can achieve high bandwidths (between 2 kb/h and 10 kb/h) based on the specific parameters of the environment/dataset. In fact, the target environment and its use of the procotol features has the most significant impact on the performance of the covert channels, than the parameterization. Therefore, an information hider will be required to use prior reconnaissance to get an estimate on the applicability of covert channels and their potential bandwidth to verify its sufficiency for the attack or required covert communication, respectively. In case of the implemented covert channels the embedder can directly optimize bandwidth using the evaluated parameters (*Packet Filter F_i, Capacity cap and Embedding Ratio er*) without loss of robustness. Even though the bandwidth can be modulated (with the aim of decreasing detection probability), at least for the datasets in this study targeted countermeasures and detection schemes could be designed that provide realiable detection and mitigation against these covert channels. Due to its wide spread use and possibilities for covert channels, Modbus/TCP can be expected to be one of the most probable carrier for hidden information in OT networks.

4.1.11 Key Insights from the Modbus/TCP Case Study

This section summarizes the key findings derived from the Modbus/TCP case study. These key findings are relevant for the field of Information Hiding in CPS in general or have implications for other case studies. The following selected insights can be drawn from the case study:

Application of Information Hiding Patterns for Covert Channel Discovery The application of known information hiding patterns (in this case the Network Information Hiding Taxonomy by Wendzel et al. [MWC18]) to the specification of network protocols can aid in the systematic identification of *potential* covert channels. These might not be not exhaustive, yet give a good indication of what is potentially possible with respect to the specification.

Implication of real-world environments on performance and feasibility Due to significant differences in the public datasets/environments used in this case study, the practical results show that some of the identified *potential covert channels* are generally not feasible or highly dependent on the target environment, including different devices, firmwares, hardware, and differences in implementation. As the specification leaves room for interpretation and implementation, software from different vendors can show significant differences in their behavior, which, on the other side, can have direct impact for the use of covert channels or even *enable* covert channels that are not apparent from the specification, i.e., the **difference between specification and implementation** is of special importance. These differences between environments can result in lower bandwidths, higher detectability, less robustness, or even infeasibility of the application of certain covert channels. For example, in one environment a covert channel is statistically undetectable whereas in another environment the exact same covert channel is so obtrusive that is easily detectable by eye (when inspecting the network traffic). This is an important insight as it underlines the importance for the secret sender and receiver to be aware of the specific cover characteristics of the environment that is used as cover. That means that prior reconnaissance is mandatory. This is an important fact for adversarial behavior modeling, as well as detection and mitigation strategies as it requires the adversary to analyze the cover i.e., taking actions that might be detected even *before* covert channels are established. The same holds true for the warden, who needs specific knowledge of the cover characteristics for successful detection and mitigation.

Cover Selection and Bandwidth Modulation In the practical evaluation, three means for cover selection and bandwidth modulation were tested that are also part of the stego-key: A packet filter, that selects only certain types of packets as cover, an embedding-ratio which pseudo-randomly selects a subset of the prior filtered packets for embedding (i.e., the secret message is pseudo-randomly distributed across a large number of packets) and a capacity parameter, that enables the secret sender to change between different embedding capacities. All these three have an impact on the achievable bandwidth. The packet filter showed to be of special significance at certain types of packets might not

even occur in certain environments/datasets, making certain covert channel-parametrization combinations infeasible. The capacity parameter worked as intended and provides a predictable impact on bandwidth. The embedding-ratio showed to be a feasible approach for distributing the secret message pseudo-randomly across a large set of cover objects, aimed at making detection more difficult.

Detection & Mitigation Despite the previous mentioned efforts to modulate the bandwidth to decrease the probability of detection, at least for the practically tested covert channels, promising detection methods could be identified and partially verified. With the exception of the covert channel leverging transaction identifiers as cover, rule-based or script-based IDS systems like Snort and Zeek are sufficient for detection of the implemented covert channels. Depending on the dataset or imlementation of the Modbus Client/Server, anomaly-based or other ML-based techniques might be required for reliable detection of the Transaction ID-based covert channel. Indeed, for all four practically tested covert channels, active Wardens can be designed to prevent these covert channels without prior detection.

4.2 *Case Study CS_2*: Process Data Transmission

Parts of this case study have been peer reviewed and published in part within the scope of the following publication as joint work with the co-authors Jana Dittmann, Christian Krätzer and Tom Neubert: [LNK+21]. Kevin Lamshöft was the originator of the covert channel discussed in this paper and was involved in its conception and implementation. Christian Krätzer contributed to the Introduction and State-of-the-Art sections of the named paper and was the source of the idea for the synchronization method. Jana Dittmann improved the structure of the paper, formalized the approach, and provided the basis for the discussion of embedding and retrieval keys. Tom Neubert is the originator of the detection approach described in Section 4.2.5

In this case study, the long-term storage of process data is used as a cover for stealthy exfiltration. The communication between a PLC and a (historian)-database is leveraged to establish a covert channel.

4.2.1 Threat Scenario & Participants

In this threat scenario, a PLC has been compromised. From this PLC data shall be exfiltrated in a stealthy way. Figure 4.16 illustrates this scenario and its participants in hidden communication at the example of the Plausible Reference Architecture, introduced in Section 3.1. As shown in the figure, the core idea of the covert channel proposed for this scenario is to leverage the communication between PLCs and (historian-) databases which store process data. The investigated covert channel embeds hidden information into values that the PLC receives from one of its sensors. This modified process data then gets transmitted to the historian database to be stored. The retrieval takes place by accessing the database, e.g. by an engineer.

Although the focus here is on the exfiltration part, the proposed covert channel can also be used as part of a more complex Command & Control (C&C) channel when paired with a covert infiltration channel. To be able to exfiltrate this information, a device on the embedder side has to be compromised that is part of the communication flow between sensors that collect data about a physical process. These data are collected and processed by Programmable Logic Controllers (PLCs) and aggregated and stored in a process database or historian, with potential intermediate stops. Depending on the network architecture, components such as PLCs, network elements (e.g. firewalls), Human-Machine Interfaces (HMIs), or other components, like data aggregators or protocol converters, may be present. On the receiving side, any device that has (potential) access to the historical data, such as the database server itself or engineering workstations, can be potential receivers/retrievers of the hidden message. For instance, an engineer who has access to the historical data can act as an internal threat by, for example, infecting his computer with malware that is capable of retrieving the hidden information or writing such code directly at his workplace. In the context of MITRE ATT&CK®ICS (see Section 2.2.5.2), this scenario is part of the *Initial Access* phase and is reflected in the tactics of *Data Historian-*, *Engineering Workstation-* and *Supply Chain- Compromise*. The previously mentioned scenario of internal threat, developing malware at work, is not currently reflected in MITRE ATT&CK®ICS.

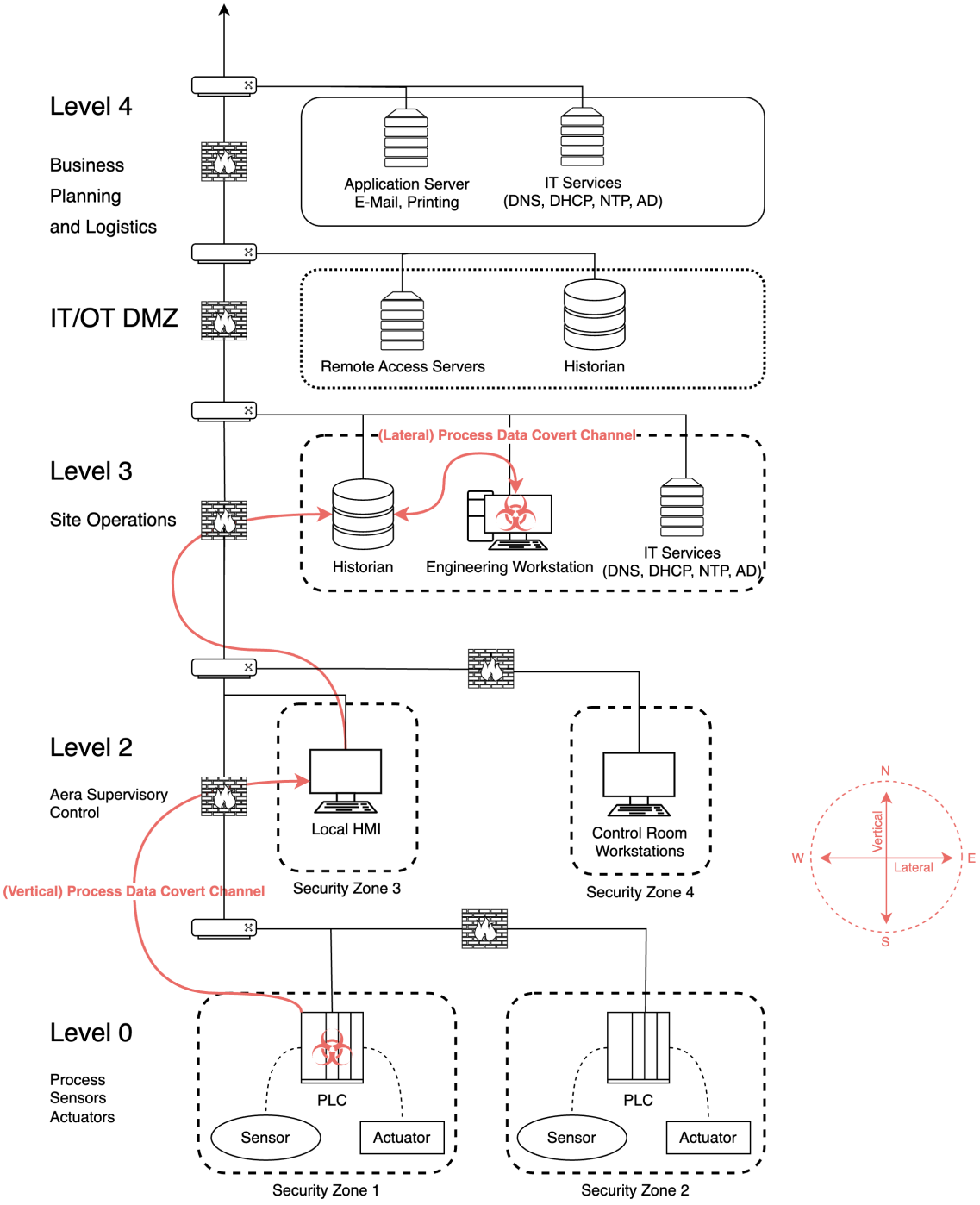


Figure 4.16: Potential threat scenario involving adversarial use of process data covert channels at the example of the Plausible Reference Architecture (*PlauRA-3*).

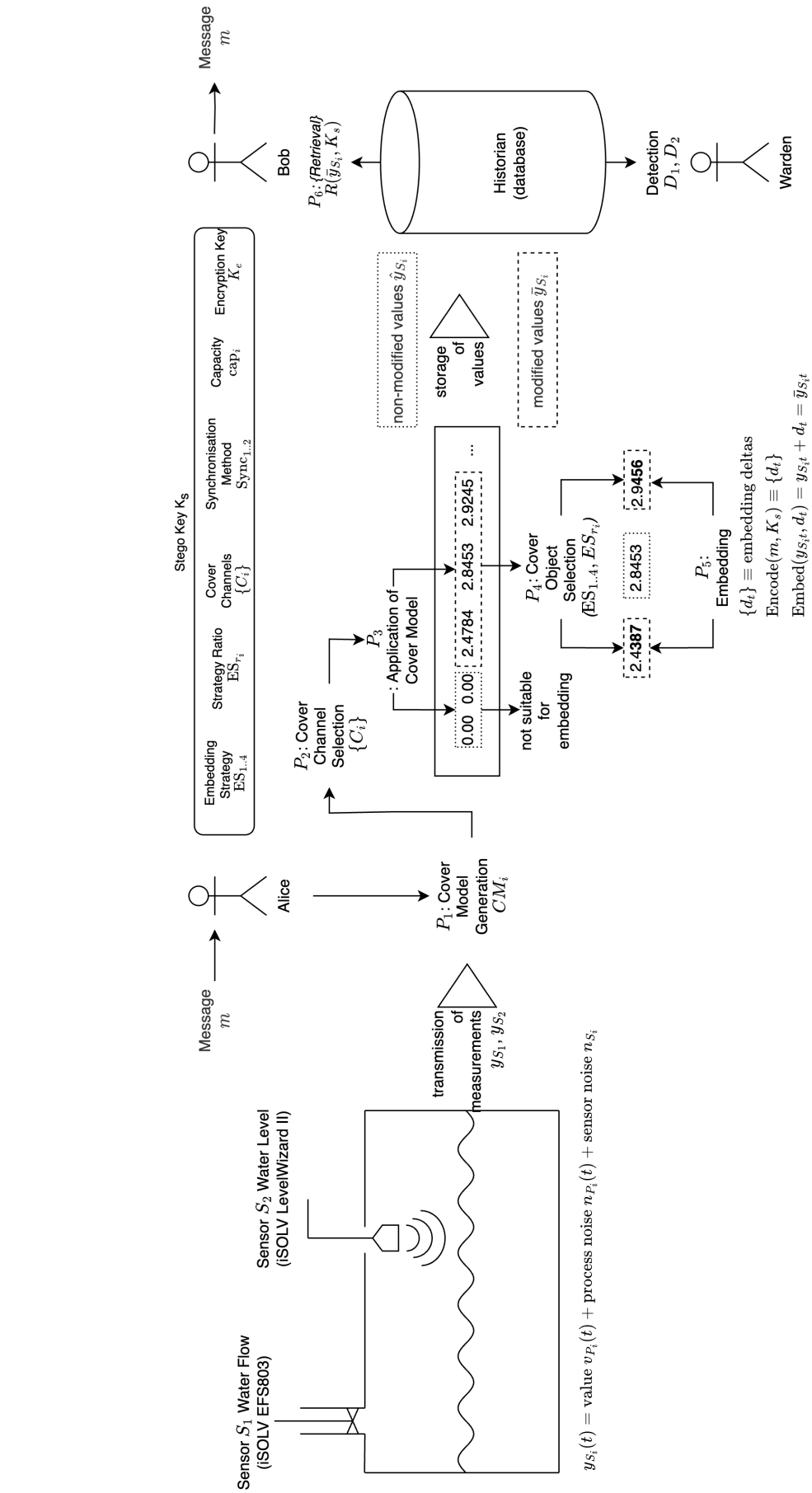


Figure 4.17: Overview on the threat scenario and covert channels leveraging the transmission of process data. Modified figure based on [LNK+21].

4.2.2 Attacker Model

The level of knowledge of the attacker is not a major factor in this situation, as the aim of the data exfiltration strategy is to gain more insight and could be used as part of the reconnaissance for further attacks. Nevertheless, prior infections are necessary to set up the covert channel, which necessitates a certain set of prior information. The attacker model assumes that the attacker has the capacity to deploy supply chain attacks and inside threats, which require a great deal of effort and are usually conducted by actors with considerable resources, such as nation state actors or advanced persistent threats (APT). Therefore, the attack scenario and attacker model are based on the assumption of a highly targeted attack with almost unlimited resources in terms of time, money, and personnel. Nevertheless, on the technical side, there are limited resources and capabilities, as restrictions arise in terms of computational power and data storage when embedding and retrieval is done in a discreet manner on components with limited resources, such as PLCs or firewalls.

4.2.3 Experimental Setup & Dataset

For evaluation of this threat scenario a public dataset containing process data is used. This dataset provided by iTrust [GAJM17] is named *Secure Water Treatment (SWAT)* and is commonly used for anomaly detection in CPS [MW19, IYC+17, SFL18, BBT22]. This dataset contains process data collected over multiple days from an experimental water treatment facility. For the evaluation, two sensors are selected from the dataset: water flow (`FIT101` in the original dataset, here S_1) and water level (`LIT101` in the original dataset, here S_2), from the A1 and A2 collections from December 2015. These sensors are chosen as they represent two common but distinctive types found in such systems. A subset of 24855 datapoints from the SWAT dataset is used, which reflects around 6 hours of the physical process. The first two hours (8285 values) are used to build the cover model (see Section 4.2.4.1) for each sensor (and can be used as a non-stego set), the other four hours (16570 values) are used for embedding and detection.

Figure 4.17 illustrates the attack scenario, in which a PLC modulates received values from its sensors. The performance of four different embedding strategies are measured and evaluated for each sensor (S_1, S_2) as well as the resulting effect on detection performance.

4.2.4 Covert Channel Design (Embedding & Retrieval Procedures)

As illustrated in Figure 4.17, the strategy for embedding and retrieval is based on six Process Steps P_i, which can be summarized as follows: P_1: Generating a Cover Model, P_2: Choosing a Cover Channel, P_3: Implementing the Cover Model, P_4: Selecting the Object, P_5: Embedding, and P_6: Retrieval. The following sections provide a more detailed description of each phase.

4.2.4.1 Cover Model Generation (P_1)

The characteristics of sensor measurements can vary significantly depending on the type of sensor, the physical process being monitored, and the current state of that

process. For example, the water flow sensor S_1 returns a constant value of 0.00 when no water is passing through it, making it impossible to embed any data into such values. To determine if a measurement is suitable for embedding, a cover model CM_i is created by observing measurements over a given period of time, such as a day. This cover model establishes constraints which indicate whether the current measurement is suitable for embedding. To create these constraints, two methods are combined: (1) a simple states model is used to describe the current state of the physical process and (2) the noise of the sensor is estimated for this state. Additionally, sensors can show recognizable patterns over time (e.g. reaching a peak level after filling a water tank). These patterns are used to enhance the cover model and to define constraints when values are used for synchronization or embedding the actual message. Figure Figure 4.18 illustrates the cover models for sensors S_1 and S_2. Taking into account that physical processes can be simplified into a states model with two states (steady and transient), the attacker is limited in the computational resources available on the target system. Therefore, a representative cover model is created.

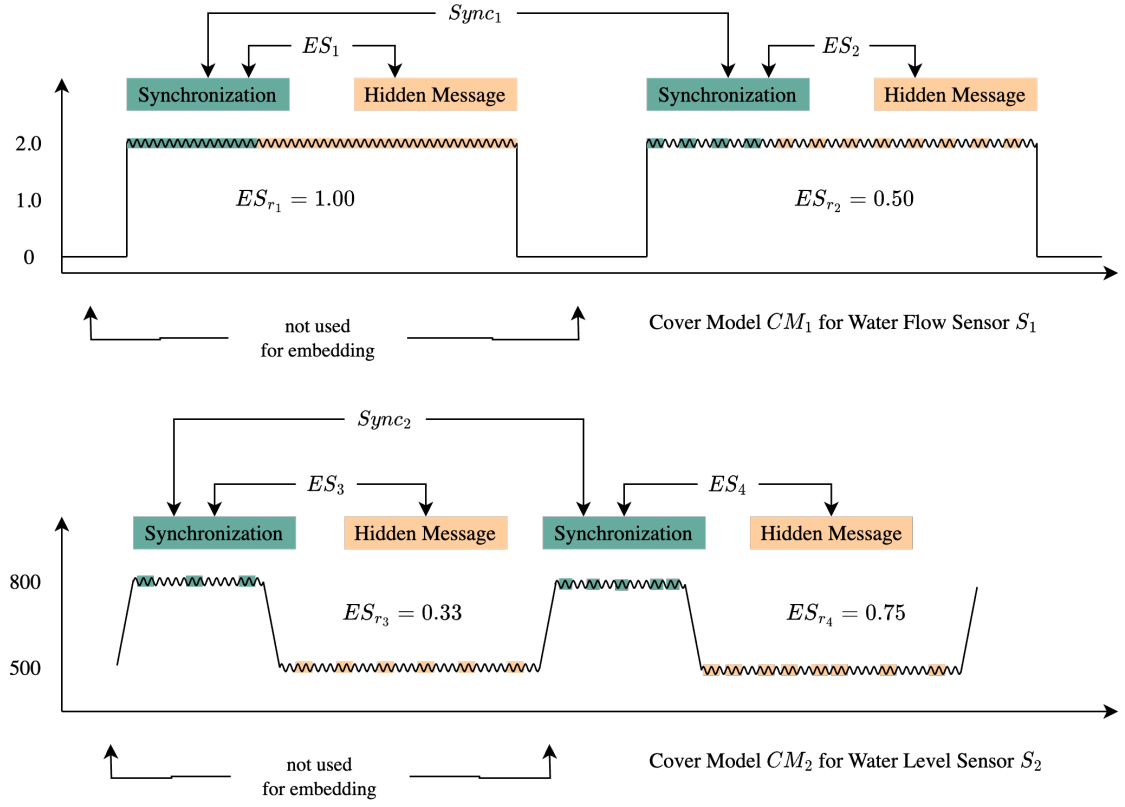


Figure 4.18: Cover Model CM_1, CM_2, Synchronisation Method $Sync_1, Sync_2$ and Embedding Strategies ES_1, ES_2, ES_3, ES_4 for Sensors S_1, S_2) (Water Flow, Water Level). Modified figure based on [LNK+21].

The values of steady states can either remain constant or vary around a certain value, while transient states refer to the transition between steady states. By determining the current state, a simplified model of the physical process can be constructed by noting the average duration of each phase, its mathematical description (static/-linear/exponential/logarithmic increase/decrease) and the estimated noise for that state. As shown in Figure 4.18, two methods $Sync_1, Sync_2$ are used for synchroniza-

tion for both Sensors S_1 and S_2. For the water level sensor S_2, the steady state above 812 centimeters is used for synchronization and the linear transient state (which appears to be steady, but is actually linear increasing) around 500 centimeters is used for embedding the actual message ($Sync_2$). The synchronization method $Sync_1$ for the water flow sensor S_1 is different, as there are two steady states, in which the water flows or not. Embedding is only possible when water flows through the sensor. For synchronization, the transition between not-flowing and flowing is used. The first 128 values after the sensor starts to read a flow are used for syncing and the rest for embedding the actual message. To get comparable results, the same hidden message is used for both sensors, as well as the same parameters for the embedding strategies. Additionally, the states model is useful when employing a dynamic capacity strategy by calculating the noise per state to use different capacities (hidden bits per cover object) in relation to the noise, which in theory can make detection more difficult. The following section describes the estimation of sensor and process noise.

As indicated in [KLG15], the sensors of Cyber Physical Systems can display different types of noise, depending on the physical property being measured and the type of sensor. According to [AZM18], two sources of noise are generated: (1) sensor noise, which is caused by the (in-)accuracy of the sensor measuring a physical property, and (2) process noise, which are fluctuations of the physical process, e.g. a moving liquid. The sensor measurement y_{S_it} can be expressed as the combination of the physical value v_t and the noise n_t, which is the sum of the process noise n_{pt} and the sensor noise n_{S_it} at a given point in time t:

$$y_t = v_t + n_t = v_t + n_{pt} + n_{S_it}$$

To embed a hidden message in the sensor measurement, the attacker adds a δ_t (which may be positive or negative) to the sensor measurement value (see Figure 4.17). The modified value which is transmitted and stored in the historian can therefore be expressed as:

$$\bar{y}_t = y_t + \delta_t = v_t + n_t + \delta_t$$

In order to remain undetected, the aim is to keep modifications so small that a) values are still within the limits and distribution of usual noise and b) a receiver is still able to retrieve the hidden information. In a dynamic capacity scenario, the information hider has to solve the optimization problem of finding the highest possible capacity at a given point in time while staying within the limits and distribution of the usual noise. Assuming the usual distribution of the combined noise of sensor and process noise is $n_t \in P$ the distribution of the sensor reading with the embedded message should be within this usual noise distribution $n_t + \delta_t \in P$ to avoid detection. To calculate the noise, a modified approach inspired by [AMO20] is used and local noise is determined by using the standard deviation on fixed window sizes. In addition, the distribution of the last digits is used as a second feature for noise calculation. The window size is selected individually for each sensor, as transient and steady states of the underlying control process may vary in length. To obtain the potential steganographic capacity per sensor value in bits, the doubled standard deviation of the values for a given window X is taken to the logarithmic base of 2 to receive the deviation in bits: $cap_{max}(X) = log_2(2\sigma(X))$. This gives the maximum number of bits

which can potentially be modified while still staying within the usual fluctuations of the values.

4.2.4.2 Cover Channel Selection (P_2)

In this case study, the covert sender is a Programmable Logic Controller (PLC). This means that the list of potential cover channels is determined by the sensors connected to the PLC. If a compromised network element, such as a protocol converter or firewall, is present, the number of available cover channels is two, namely S_1 and S_2. In [WMH17a], the authors suggest using a scatter hoarding strategy to hide a message across multiple unused registers of sensors in a smart building. This technique is based on the concept of achieving a high bandwidth while remaining undetectable, as the message is spread across a large set of cover objects. The debate between hiding a message in a small number of cover objects (larder hoarding) or scattering it across a larger set of cover objects (scatter hoarding) is discussed in Ker's 2007 paper on batch steganography and steganalysis [Ker07]. This work argues that it is, in theory, the best strategy to embed only in a small number of cover objects, at least if the warden is using a detection method specifically designed for such pooled steganalysis. This leads to the conclusion that, in order to choose an appropriate strategy for an information hider, they must consider the warden, which puts them in a game theoretic problem, as described by Schöttle and Böhme in 2013 [SB13]. Consequently, the potentially best strategy for the hider is dependent on their knowledge of the system and the warden, their capabilities, their position in the system, and thus their access to cover objects and cover channels.

In the natural world, rodents such as squirrels move their food around to stop it from being stolen by other animals. This same concept can be applied to Information Hiding. where ihas two implications. Firstly, the hidden elements (in this case, a hidden message) may have different lifespans. Depending on the covert channel, a hidden message may only be accessible in real-time, or it may be retrievable even years later. This is known as the temporal aspect of the persistence of a covert channel or hidden message. Secondly, the general idea of relocating is reflected in Information Hiding as changing the embedding position or cover selection. In [LNK+21], we proposed four main types of relocating, each with multiple options to trigger the proccess:

1. *Cover Channel* Selection,
2. *Covert Channel* Hopping,
3. *Embedding Position* Change,
4. *Cover Object* Selection.

Each of these relocating processes can be triggered by certain events, in [LNK+21] we proposed the following four:

1. *recache-by-default* (constant change as part of the embedding algorithm)

2. *triggered-recache* (e.g. triggered by a control message sent to the covert sender)
3. *conditional-recache* (triggered when a pre-defined condition is met e.g., entropy of cover object below threshold)

The notion of asynchronous covert channels is essential to this exfiltration approach. This is due to the fact that in CPS, it is typical to store process data for extended periods of time in a historian database. This is used to construct a covert channel where the embedded information can be accessed without taking into account the time it was embedded. Consequently, these channels can only be established if the cover objects are long-lasting or persistent. In the case of Cyber Physical Systems, there are three levels of durability that differ depending on the system.

1. *long-term persistence*, similar to dead drops e.g., historians (months to years)
2. *mid-term persistence* e.g., log files (days to weeks)
3. *short-term persistence* (minutes to hours)

In contrast, a *synchronous covert channel* makes use of an *ephemeral cover channel* e.g., direct TCP/IP communication between an HMI and PLC (assuming that communication is not logged). The data exfiltration strategy proposed here uses long-term persistent storage of process data as a cover channel. This distinction reveals that there are potentially a large number of other attack scenarios that leverage different *levels of persistence*. This differentiation is also important when designing a warden, as it is essential to place wardens in the right positions to detect such covert channels.

4.2.4.3 Cover Model Application (P_3)

The purpose of Cover Model Application (P_3) is to determine for each Cover Channel C_i, whether the current cover object (i.e. the measurement value y_{S_it}) being sent on that channel is used to conceal a hidden message, synchronize sender and receiver, or not used at all (e.g. due to low noise). This decision is based on the chosen synchronization method $Sync_i$ and the cover model CM_i, which set the parameters for whether a cover object is suitable for embedding a message, and whether it is part of a synchronization or embedding.

4.2.4.4 Cover Object Selection (P_4)

In the wild, animals use different sizes and amounts of food caches to prevent or reduce losses due to theft. In the field of Network Information Hiding, two types of steganographic payload variation are commonly used: Capacity Modulation, which is the steganographic payload per cover object, and Bandwidth Modulation, which is the steganographic payload over time. These two options are especially useful for Cyber-Physical Systems, as they allow for different strategies to avoid detection. Capacity modulation can be beneficial in scenarios where the characteristics of the cover channel vary, such as when the noise of sensor values is different in transient states from steady states. Bandwidth modulation can be used to adjust the payload

over time in relation to the physical process, thus reducing the detection probability by a warden. Both strategies, and especially when used together, can help an attacker to optimize the steganographic payload against detection probability. To modulate the bandwidth in this approach, four Embedding Strategies (ES_i) are applied to drive the Cover Object Selection (P_4). For evaluation of this approach, four different strategies are used which employ different Embedding Strategy Ratios (ESr_i), as shown in Table 4.12.

Table 4.12: Embedding Strategies ES_i and correlating Embedding Strategy Ratios ES_{r_i}

Strategy	Description	ES_{r_i}
ES_1	every suitable value	1.0
ES_2	every second suitable	0.5
ES_3	every third suitable value	0.33
ES_4	pseudo-random method	0.75 (*can be arbitrary set*)

The embedding strategy ratio ES_{r_i} is a measure of how many of the values that are suitable for the cover model CM_i are used for embedding. In ES_1, all suitable values are used, while in ES_2, only every second suitable value is used, and in ES_3, every third suitable value is used. ES_4 uses a pseudo-random function to decide which values are used, and a seed is part of the stego key K_S to enable synchronization. The aim of these strategies is to reduce the bandwidth in order to make detection less likely. The embedding ratio r takes into account all values, including those that are not suitable for embedding due to low noise, and is calculated as $r = \frac{\bar{y}_{S_i}}{\hat{y}_{S_i}+\bar{y}_{S_i}}$.

The embedding ratio r shows how many values were changed ($\bar{y}$) to embed a message in comparison to all values (non-modified $\hat{y}$ due to CM_i and ES_i and modified $\bar{y}$). The strategy ratio ES_{r_i} determines how many values are used for embedding the hidden message, and is a direct factor of the resulting embedding ratio r and bandwidth. The pseudo-random method allows for fine-tuning of the embedding ratio r by giving the function a probability to decide whether a value is used or not. This makes detection and suppression more difficult, as the embedding appears random to an outside viewer. The embedding strategies are applied to both the synchronization sequences and the sequences used to embed the message, as shown in Figure 4.18.

4.2.4.5 Embedding (P_5)

In P_5, the insertion of hidden bits for a given value is carried out, while the Cover Model CM_i of Process Step P_1 determines whether they are part of the synchronization sequence or taken from the actual message. Consequently, both the synchronization sequences and the secret message are encrypted, encoded, divided into embedding blocks, and embedded in the target value (see Figure 4.19). In line with the attack scenario, where an attacker is assumed to be attempting to exfiltrate sensitive information, a realistic hidden message is generated with detailed information about the hardware and firmware of the target system. The Linux command `lshw` is used to generate a comprehensive hardware list. For the testing environment, this message is approximately 13.5 kilobytes in size. For the sake of simplicity AES-ECB

is used, but other modes are possible as well, as the synchronization methods are specifically designed for this use case.

As shown in Figure 4.19, the embedding process can be divided into five steps. In the first step, the cover object (value) is split into two parts: head and tail. The size of the parts is given by the capacity parameter cap_i. The capacity is calculated (dynamically chosen) using the cover model CM_i in order to set a capacity in relation to the current noise, or a static capacity cap_i is used (as part of the stego key K_S).

For the two sensors S_1 and S_2, capacities between 1 and 12 bits are tested. The algorithm splits the last four digits (tail) from the rest of the value (head). These last four digits are then treated as a 16-bit integer. The tail is then transcoded to a binary string. The next block in the size of the chosen capacity cap_i is retrieved from the encrypted message queue, which will then replace the last bits of the tail. This bit-block replaces the corresponding last bits of the original. The new binary string is then encoded back to an integer and joined with the original.

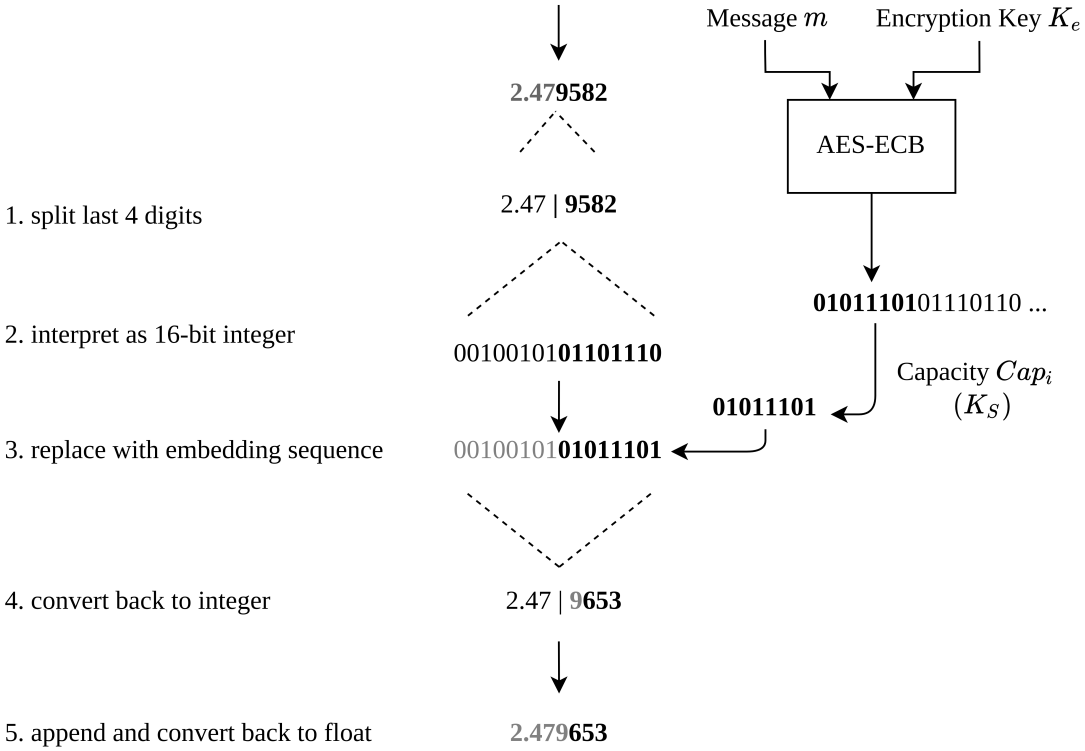


Figure 4.19: The five-step process for embedding secret messages in the last digits of sensor values. Figure based on [LNK+21]

4.2.4.6 Retrieval (P_6)

A covert receiver requires access to the (historian-) process database. An engineer, for instance, may access the database for monitoring or maintenance purposes. To be able to retrieve the message, the retriever must be aware of (1) the algorithm, (2) the cover channels (sensors) the covert sender has access to, and (3) the stego key K_S. As illustrated in Figure 4.17, the **Stego Key** K_S consists of six components: (1) Cover Channels (Selection Method) $\{C_i\}$, (2) Embedding Strategy ES_i, (3)

(Embedding) Strategy Ratio ES_{r_i}, (4) Synchronization Method $Sync_i$, (5) Capacity Cap_i and (6) the Encryption Key K_e. In the context of the attack scenario, a static stego key is assumed, e.g. as part of a supply chain attack. At first glance, the retrieval may seem like searching for a needle in a haystack. However, the stego key K_S is designed in such a way that the search is successful: With knowledge of the strategy used to select cover channels $\{C_i\}$ for embedding, the retriever knows where to look for hidden messages. With that knowledge and the corresponding synchronization method $Sync_i$, the receiver can create a cover model. With that model and the knowledge of the chosen embedding strategy ES_i, and embedding ratio ES_{r_i}, the receiver can identify the synchronization sequences, which contain information about where the hidden messages are stored. To decode and decrypt these, the receiver needs to know the capacity cap_i and the encryption key K_e.

4.2.4.7 Performance Comparison

Table 4.13: Embedding Strategy Comparison (sorted by Embedding Strategy Ratio ES_{r_i})

Sensor		Parameterization				Bandwidth		
S	er_{max}	*Embedding Strategy*		ES_{r_i}	*cap*	*er*	bit/sec	kb/h	kb/day
S_1	0.35	ES_1	*all*	1.00	8 *bit*	0.35	2.84	1.28	30.67
		ES_4	*random*	0.75		0.27	2.12	0.95	22.90
		ES_2	*alternating*	0.50		0.18	1.41	0.63	15.23
		ES_3	*thirds*	0.33		0.12	0.94	0.42	10.15
S_2	0.53	ES_1	*all*	1.00	8 *bit*	0.53	4.28	1.93	46.22
		ES_4	*random*	0.75		0.40	3.21	1.44	34.67
		ES_2	*alternating*	0.50		0.27	2.14	0.96	23.11
		ES_3	*thirds*	0.33		0.18	1.42	0.64	15.34

To evaluate the effect of the four embedding strategies ES_i on the resulting bandwidth per sensor, a static capacity of 8 bit ($cap_{1,2} = 8$) is used for both sensors S_1, S_2. Table 4.13 shows that the water flow sensor S_1 does not have the same potential bandwidth as the water level sensor S_2. The maximum embedding ratio r_{max} describes how many of all measurements can be used for plausible embedding, and for the water level sensor S_2 it is 53%, while for the water flow sensor S_1 it is only 35%. This directly affects the maximum achievable bandwidth per sensor. Within the constraints of the cover model, the maximum achievable bandwidth for the water level sensor S_2 is about 4.2 bit per second and 2.84 bit for the water flow sensor S_1 when using every suitable value for embedding (ES_1). This translates to 370kb (water level) and 245kb (water flow) per day. Although this may not seem like a high bandwidth, it is comparable to the bandwidths used in exfiltration attack scenarios ([HLD+20], [LD20]). For example, the hidden message (which contains detailed hardware and firmware information about the target system) is only 13.5kb in size and therefore could be transmitted between 18 and 27 times depending on the sensor per day. Even when using an (embedding) strategy ratio of only 0.33 ($ES_{r3} = 0.33$) on the lower performing water flow sensor S_1, the message could be transmitted six times (81kb) in the course of a day. A dynamic capacity based on

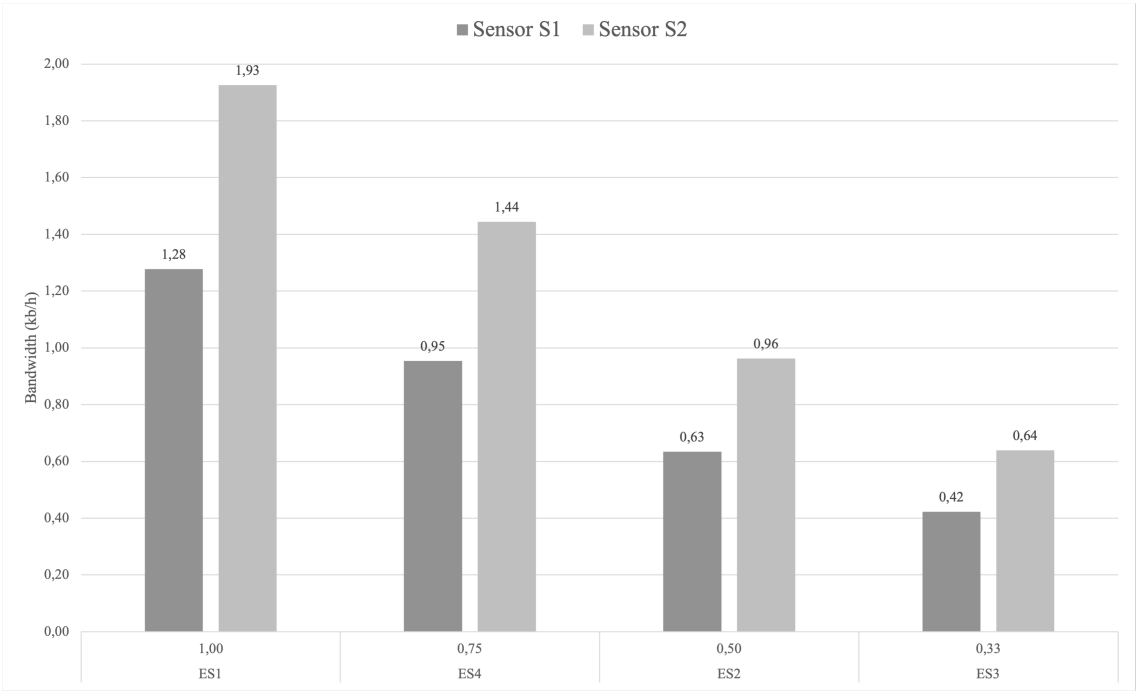


Figure 4.20: Embedding performance (bandwidth in kb/h) for the proposed covert channel in process data. Figure based on data from [LNK+21].

the local noise of the sensor could potentially increase the bandwidth while also improving the stealthiness.

4.2.5 Detection & Mitigation

In the joint publication [LNK+21], T. Neubert proposed two detection approaches based on a J48 two-class anomaly classifier. J48 was chosen due to its performance, while Logistic model trees (LMT), support vector machines (SVM) and multilayer perceptrons (MLP) were also tested. The first detector, D_1, uses an 8-dimensional feature space that takes the last three decimals into account. The second detector, D_2, only uses one feature (Std_{x3}) which was selected using WEKA's *InfoGainAttributeEval* with its *Ranker* method [LNK+21]. The results are shown in Figure 4.21. Both detection approaches (D_1, D_2) achieved good detection results ($TPR > 80\%$ and $FPR \leq 0\%$ for D_2) for datasets with embedding ratios greater than 0.95. However, the less information that is embedded, the more difficult it becomes to detect stego objects. For example, with an embedding ratio of 0.33, which means that every third value contains hidden information, the TPR is reduced to only 42.3% (D_1) and 40.6% for D_2. This leads to the conclusion that such means of bandwidth modulation (the embedding strategy/ratio) can be very effective in reducing detection probability (at least in the case of the tested J48, SVM, LMT and MLP classifiers). To mitigate the evaluated covert channel, the resolution of sensor values can be reduced. Therefore, it is up to the operator to decide which resolution is actually required to run the process. For example, in this case study, the resolution of the water flow sensor values is very high, comprising six decimals, whereas the data suggests that two decimals would be sufficient. Another option is to use active wardens which calculate the noise for each sensor and replace the least significant

bits using random bits (noise). When such an active warden is placed correctly, the investigated covert channel can be mitigated.

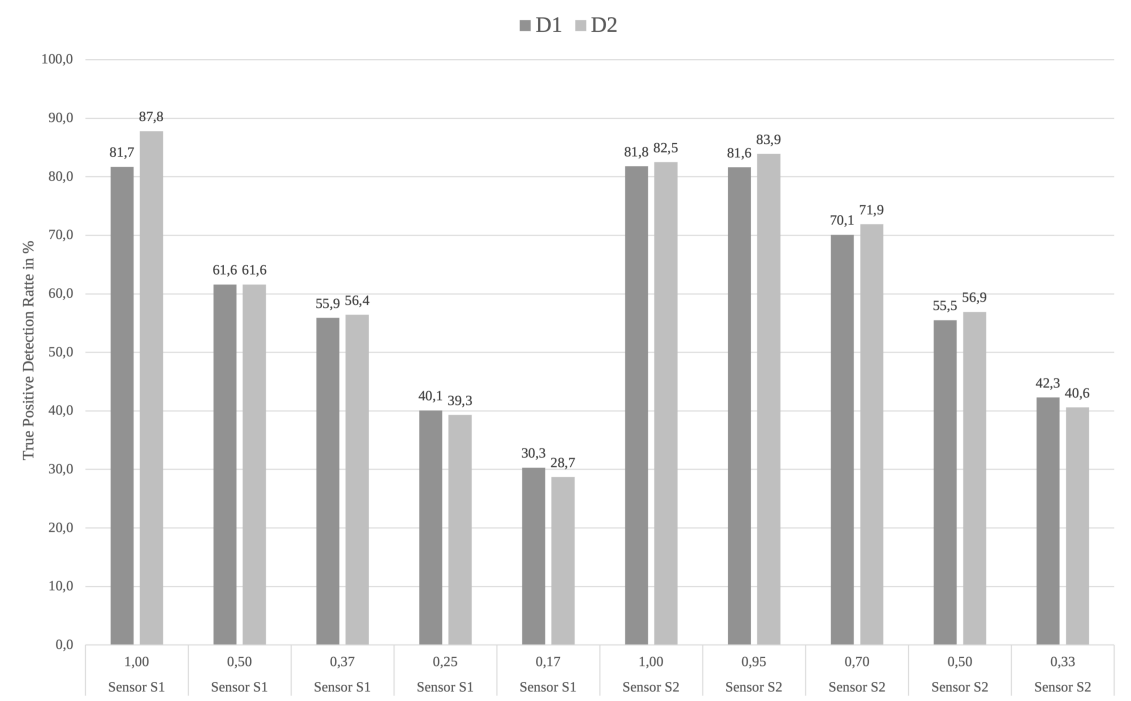


Figure 4.21: Comparison of *True-Positive* detection rate between detectors D_1 and D_2 proposed by T. Neubert in [LNK+21] in dependency of the used sensor (S_1, S_2) and embedding ratio. Figure based on data published in [LNK+21].

4.2.6 Key Insights from the Process Data Transmission Case Study

This section summarizes the key findings derived from the Process Data Transmission case study. These key findings are relevant for the field of Information Hiding in CPS in general or have implications for other case studies. The following selected insights can be drawn from the case study:

Process Data as Cover Object (Lifecycle, Locations, Persistence & Robustness) In contrast to common network covert channels, in which the cover object (i.e., a network packet) is sent from a sender to a receiver, cover objects in Cyber-Physical Systems (i.e., process data) are processed multiple times at different *locations* and points in *time*. For example, at first a sensor measures a physical property and provides an encoded signal towards a PLC. On the PLC this signal is interpreted and processed. At some point, this value might be sent to an Historian database. To do so, the value is again potentially encoded in another format and packed into the payload of an application-level protocol. This frame is sent across the network towards the receiver with potential interstops and procotol conversions (in which the measurements gets re-encoded). On arrival, the database interprets the network packet and stores the measurement values (potentially again in a different format/encoding). This example illustrates the different lifecycles of process data as cover objects. The first aspect is location. Process data is processed at different locations, which enables multiple options where the embedding and retrieval can take place. The other aspect is time. On the PLC the sensor value might be available for only one cycle time, on the network the packet is ephemeral whereas in the historian database the cover object might be retrievable even years after embedding. This enables for entirely different embedding- and retrieval scenarios in comparison to common network covert channels. Another relevant factor is the conversions the cover object is facing. For example, conversions from float to integer (which are, for example, very common in environments using Modbus/TCP) might destroy the embedded message. Another example are sensors, which typically can provide much more precision than is required for long-term storage in historian databases. In that case, the values often get trimmed down, e.g., to a certain amount of decimals. In case of an LSB-based covert channel, this would result in loss of the secret message. In general, it can be expected that such covert channels must be designed in a more robust way than, for example, covert network storage channels.

Noise as Cover The core idea of the covert channel in this case study is to use the noise in process data, not the process data itself as cover. This is an important difference. This case study shows that is possible, that instead of simply applying a Least-Significant-Bit/Byte method (LSB) to a sensor measurement, a covert channel can leverage the noise induced by the physical process itself and inaccurities in sensor readings. By calculating this noise, it is possible to make modifications (i.e., to embed hidden information) into a measurement while at the same time preserving its noise signature. This is possible due to the randomness of such noise, which provides high enough entropy to hide secret messages.

Process data for Stego-Key Exchange & Synchonization In this case study, recurrent patterns in the process data are used to synchronize the covert sender and receiver. Using key-derivation functions on certain parts of process data can be leveraged for additional (stego-) key exchanges.

Cover Selection, Bandwidth and Capacity Modulation In this case study, a dynamic method for cover selection based on the suitability (in this case noise measured by entropy) for each cover object is proposed and successfully evaluated. Moreover, based on the noise in the process data, the capacity per cover object can be dynamically adjusted to keep the modifications within the bounds of usual noise. Similarly to the Modbus/TCP case, the bandwidth is further modulated using an embedding-ratio parameter aimed at and distributing the secret measures pseudo-randomly across a large number of cover objects. In fact, this approach was shown to be effective against the proposed machine learning-based detection approaches proposed in the corresponding publication ([LNK+21]).

Detection & Mitigation The detection approaches proposed in [LNK+21] were shown to be effective against the investigated covert channel when using high embedding ratios (and therefore high bandwidths). In low-bandwidth scenarios though, many cover objects with hidden information remain undetected. Another important factor is that the proposed approaches are two-class classifiers, i.e., these are especially trained for this specific covert channel. It remains to be further investigated how a generalized approach could work against a broader set of covert channels leveraging process data as cover. However, by reducing the accuracy/resolution of the values, most covert channels can be limited in their capacity/bandwidth or completely eliminated. Another option are active wardens, randomizing least significant bits in noisy measurements to overwrite any hidden bits (blind supression).

Magic Triangle & Steganographic Cost In this case study, the relationship between bandwidth and undectability is observable as described in the *magic triangle* (see Section 2.1.7). With increasing bandwidth and capacity, the carrier gets more and more degraded (steganographic cost), making them easier or more likely to detect. The results show a direct linear impact of the embedding ratio (and by that the achieved bandwidth) on the true positive detection rate.

Implications for Wardens As mentioned above, process data is processed at multiple locations. Each of these inter-stops is a potential embedder of hidden information (covert sender). Therefore, when designing countermeasures, the placement of the warden is of particular significance.

4.3 *Case Study CS_3*: Time Synchronization Protocols

Parts of this case study have been peer reviewed and published in part within the scope of the following publications as joint work with the co-authors Jonas Hielscher and Christian Krätzer ([HLKD21, LHKD22]) and Jana Dittmann ([HLKD21, LHKD22, LD22]):

Time synchronization protocols, such as, for example, the *Network Time Protocol* (NTP) [MBKM10] , the *Precision Time Protocol* (PTP) [IEE08] and the *Network Time Security* (NTS) [FST+20] Procotol, are vital, yet highly underrated protocols when it comes to their perception as potential threat to network security. Time synchronization protocols are required in many networks for proper operation of network encryption (e.g., SSL/TLS) and in process control for coordinated procedures. While the *Network Time Protocol* (NTP) is the most prevalent protocol for synchronizing clocks in IT networks, the Precision Time Protocol (PTP) is found primarily in Industrial Control Systems (ICS) networks that comply with higher demands regarding accuracy and resolution. *Network Time Security* (NTS), specified in 2020, is an addition to NTP, providing cryptographic means for securing the time synchronization process and is set to become the new standard for IT networks. These time synchronization protocols were selected for further examination and analysis due to the fact that preliminary research revealed that they possess certain distinctive characteristics that make them susceptible to covert channels which can be used to secretly communicate across security zones and -levels:

(1) Time synchronization packets contain multiple high-resolution timestamps. As shown in previous work, timestamp fields in network packets can contain high entropy, allowing the embedding of hidden information that is hard to detect [HLD+20].

(2) Time synchronization protocols use a hierarchy with central servers, resulting in network traffic passing through network (security-) zones and levels (especially vertical North-South communication, see Section 3.6). This might be used by adversaries to establish covert communication channels across segregated networks.

(3) Infrastructure protocols and procedures are potentially neglected in security considerations and might pass firewalls and IDS-systems (e.g., due to missing traffic inspection and/or protocol dissectors).

This case study is an aggregation of three publications with varying co-authors, covering the three mentioned time synchronization protocols:

Covert Channels in NTP [HLKD21] In this paper, in similar fashion to our Modbus/TCP analysis, the Network Pattern Taxonomy [MWC18] is used to systematically identify potential covert channels in the specification of *Network Time Protocol*. Furthermore, in a practical analysis with different Operation Systems and NTP-client combinations the covert channels were tested in terms of feasibility, bandwidth, and plausibility in the corresponding environment (*contextual plausibility)*.

Threat of Covert Channels in NTP and PTP [LHKD22] This is an extended journal paper based on [HLKD21], covering threat scenarios imposed by covert channels in time synchronization protocols and, in addition to NTP, also covers which of the previous identified covert channels in NTP might also be applicable in the same way to the *Precision Time Protocol (PTP)*.

Covert Channels in NTS [LD22] This paper is dedicated to *Network Time Security* (NTS) and covers a theoretical covert channel analysis for both sub-protocols NTS-EF (*NTS Extension Fields*) and NTS-KE (*NTS Key Establishment*) as well as practical evaluation of one specific covert channel.

The evaluations of these publications are extensive in length and detail and mainly focus on IT networks. Therefore, here, a broader perspective is taken on how the identified covert channels fit into the threat landscape of Industrial Control Systems and provide lessons learned, for example the role of active wardens in mitigation.

The following section introduces the Network Time Protocol (NTP) and briefly describes relevant covert channels. It then puts these into perspective by discussing potential threat scenarios in the Plausible Reference Architecture of Section 3.1. A brief overview of Precision Time Protocol (PTP) is provided and compared to NTP. Lastly, covert channels in Network Time Security (NTS) are discussed, as NTS introduces additional challenges by incorporating cryptographic measures into NTP.

4.3.1 Technical Background

This section provides the neccessary technical background for the *Network Time Protocol* (NTP), *Precision Time Protocol* (PTP) and *Network Time Security* (NTS).

4.3.1.1 Network Time Protocol (NTP)

NTP is a UDP-based protocol at the seventh layer of the OSI model, designed for time synchronization. The current version is NTPv4 [MBKM10], which is compatible with NTPv3. It usually works in a client-server mode, where clients regularly request time from servers, usually three. There are also two other modes, broadcast and peer-to-peer. NTP servers are organized in a hierarchical stratum architecture. A server with direct access to a time source (e.g. atomic clock) is a reference server with a stratum value of 0. Other servers on the first layer can request time from this server and so on, up to a maximum depth of 15. An NTP client is considered synchronized when it has requested enough reliable time information and adjusted its local clock accordingly. Every NTP packet contains at least 13 fields (48 bytes). Whether a field is used or filled with zeros depends on the mode of operation. An extension with one or two extension fields or a Message Authentication Code (MAC) field is possible but not common. In the context of covert channels, the most promising fields are the four 64-bit timestamps (reference, origin, receive, transmit) used for round-trip delay and time calculation. The upper 32 bits represent a second counter. The lower 32 bits of the timestamp fields resolve one second (down to the picosecond level). These bits - which represent more accuracy than the acutal system clock of the sender is capable of - should be set to random values [MBKM10].

4.3.1.2 Precision Time Protocol (PTP)

The Precision Time Protocol (PTP) was initially specified in IEEE 1588-2002 (also known as Version 1). This was superseded in 2008 by IEEE 15588-2008 (also known as Version 2) and recently updated with IEEE 1588-2019 [IEE08]. PTP is an industry-standard protocol that enables clock synchronization over Ethernet with sub-nanosecond accuracy using special hardware. It is a master/slave protocol with different types of clocks, such as ordinary clocks, which are single port devices that can work as master/slave, and boundary clocks, which are two port devices that can work as master and slave simultaneously. PTP is a hierarchical protocol with one system-wide grandmaster clock, whose time is distributed downwards the hierarchy, as illustrated in Figure 4.22. The master clocks distribute their time using `Sync` and `Follow_Up` messages. For slaves to accurately calculate the correct round trip delay, they send `Delay_Req` messages, which are answered by the master clock with `Delay_Resp` messages containing the precise timestamp when the message was received. Using those timestamps the slave then is able to calculate the correct time with sub-nanosecond precision.

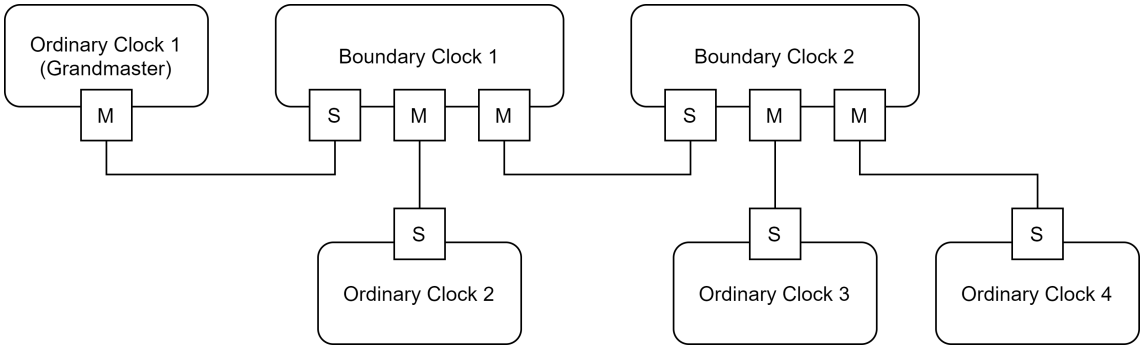


Figure 4.22: Exemplary PTP architecture with one grandmaster clock distributing time downwards the hierarchy via boundary clocks. Figure based on [LHKD22].

4.3.1.3 Network Time Security (NTS)

In 1992, the first security measures for NTP were implemented by adding cryptographic features to NTPv3 [Mil92] to protect the transmission of timestamps from manipulation. However, due to the need to distribute pre-shared keys, this feature was usually only used in small internal networks. In 2010, Autokey was introduced to NTPv4 in RFC5906 [MH10], but was later deprecated in RFC8633 [RSS19] due to inherent vulnerabilities in the architecture. Finally, in October 2020, the IETF introduced Network Time Security (NTS) in RFC8915 [FST+20], which uses Transport Layer Security (TLS) [Res18] and Authenticated Encryption with Associated Data (AEAD) [McG08] to provide cryptographic security for the client-server mode of the Network Time Protocol (NTP). NTS is based on two loosely coupled sub-protocols, the NTS Key Establishment (NTS-KE) which handles initial authentication and key establishment over TLS, and the NTS Extension Fields for NTPv4 (NTS-EF) which handles encryption and authentication during NTP time synchronization via extension fields in the NTPv4 packets. The core concept is to use the TLS-based key establishment only occasionally (ideally only once) and then mainly NTPv4 packets with the NTS Extension fields, which provide signed and partially encrypted time requests and responses.

4.3.2 Covert Channels in the Network Time Protocol (NTP)

The work in [HLKD21] is many-fold: In a theoretical analysis using the pattern-based Network Steganography Taxonomy, 49 potential covert channels were identified in the NTP protocol specification. This large amount of potential covert channels illustrate the susceptibility to covert channels for such complex yet vital and broadly used infrastructure protocols. Covering all of the identified covert channels is out-of-scope for this work. Instead, the focus is set to the two implemented and pracitcally evaluated covert channels for NTP and how these fit into the threat lanscape of common ICS systems: The theoretical results are compared in terms of (contextual-) plausibility in real-world OS NTP implementations, by testing different Operating Systems (OSs), namely Windows, macOS, Ubuntu, iOS and Android (using different NTP clients). The findings show that some implementations are not NTP specification compliant, leading to more heterogeneous NTP traffic. Again illustrating the neccesity of proper reconaissance and adaptability of covert channels and their parameters to a target environment before deployment and use. Two of the identified covert channels are implemented and evaluated in detail. One covert channel makes use of randomized bits in a timestamp field, the other one uses the stratum field to encode hidden information across multiple layers in the stratum architecture. In the following, both implemented covert channels are desribed briefly and then put into perspective for ICS and corresponding threat scenarios.

4.3.2.1 NTP Timestamp Covert Channel

This covert channel is implemented in the last 16 bits of the fraction part of the transmit timestamp. The payload is encrypted with AES in ECB mode, using a dynamic key. The idea is that the encryption creates an entropy level similar to the randomized bits in the timestamp. According to [Cac04], a covert channel is considered undetectable if the entropy distributions of overt traffic and traffic containing the hidden information are equal. This could be verified in an experiment with 10,000 NTP packets per server fetched from public NTP servers [HLKD21].

4.3.2.2 Stratum Covert Channel

The other covert channel implemented is based on the manipulation of the stratum value in server packet fields (as seen in [HLKD21]). It is assumed that downstream servers (those with higher stratum numbers) will adjust to changing stratum numbers and calculate their own stratum number accordingly. In [LHKD22], it was tested if it is enough to manipulate only one of the three upstream servers to send secret information embedded in the stratum field (by changing its value) through multiple stratum layers. The results showed that it is possible to send hidden information through different network layers, even if only one of the upstream NTP servers is malicious: The requesting client will adjust its own stratum number to the highest it received. Therefore, the secret information can be passed on as long as it is encoded with higher stratum numbers than those of the overt servers. A client will alter its own stratum value right away whenever it receives a new value from an upstream server (as long as the client is synchronized). The alteration of stratum values is not treated as an error by the client software (chrony). Thus, the time calculation is not affected. The channel can only be used reliably if the stratum

values encoding the hidden information are not changed after every request of the intermediate server. Since the timing of its request is not synchronized with those of the malicious client, information might be lost. The bandwidth of the channel is comparatively low and depends on the polling interval of the intermediate server (less than 1 bit/min in the case of a synchronized client).

In the following these covert channels are put into perspective discussing potential threat scenarios alongside the Plausible Reference Architecture of Section 3.1.

4.3.3 Threat Scenarios

As pointed out above, the main threat of covert channels in time synchronization protocols is their potential of traversing segregated networks unnoticed carrying hidden information. In this case study, this threat is illustrated with the example of the Plausible Reference Architecture of Section 3.1.

These specific threat scenarios are mainly designed with NTP in mind, however with minor modifications their are applicable to PTP and NTS as well. The selected base scenario revolves around an *outside* adversary infiltrating information into the network across all levels towards a compromised PLC. This infiltration channel can be used in combination with further covert channels to establish a full covert Command-and-Control (C2) channel e.g., to send instructions or further malware components to a compromised system within a target network perimeter.

In the following, this generalized threat scenario is further split into two smaller scenarios covering multiple configuration options, representing a broad set of potential architectures:

In scenario (A), the time synchronization takes place using an external time synchronization server or pool of servers that has access to an atomic clock (Stratum 1). This timing information is then passed down the architecture, eventually reaching the PLCs. In this threat scenario, the adversary is able to compromise this external time server (pool).

In scenario (B), representing a different approach to time synchronization, a time server in the OT domain synchronizes time using radio signals. Equipped with external antennas, such time servers are capable of using time signals distributed over Global Navigation Satellite Systems (GNSS), like GPS[8], GLONASS[9] or Galileo[10]. Another option is to use reference times from radio transmissions, for example DCF77[11] in Germany or WWV[12] in the United States.

In real-world deployments, a combination of both approaches seems plausible e.g., to use an internet-based synchronization in the IT domain and a radio-based synchronization in the OT. For the threat analysis, both approaches are discussed separately but apply in the same way to a deployment that uses a combination of both.

Figure 4.23 illustrates both threat scenarios alongside the Plausible Reference Architecture of Section 3.1.

[8] https://www.faa.gov/about/office_org/headquarters_offices/ato/service_units/techops/navservices/gnss/gps

[9] https://www.glonass-iac.ru/en/about_glonass/

[10] https://www.gsc-europa.eu/system-service-status/constellation-information

[11] https://www.ptb.de/cms/ptb/fachabteilungen/abt4/fb-44/ag-442/verbreitung-der-gesetzlichen-zeit/dcf77.html

[12] https://www.nist.gov/pml/time-and-frequency-division/time-distribution/radio-station-wwv

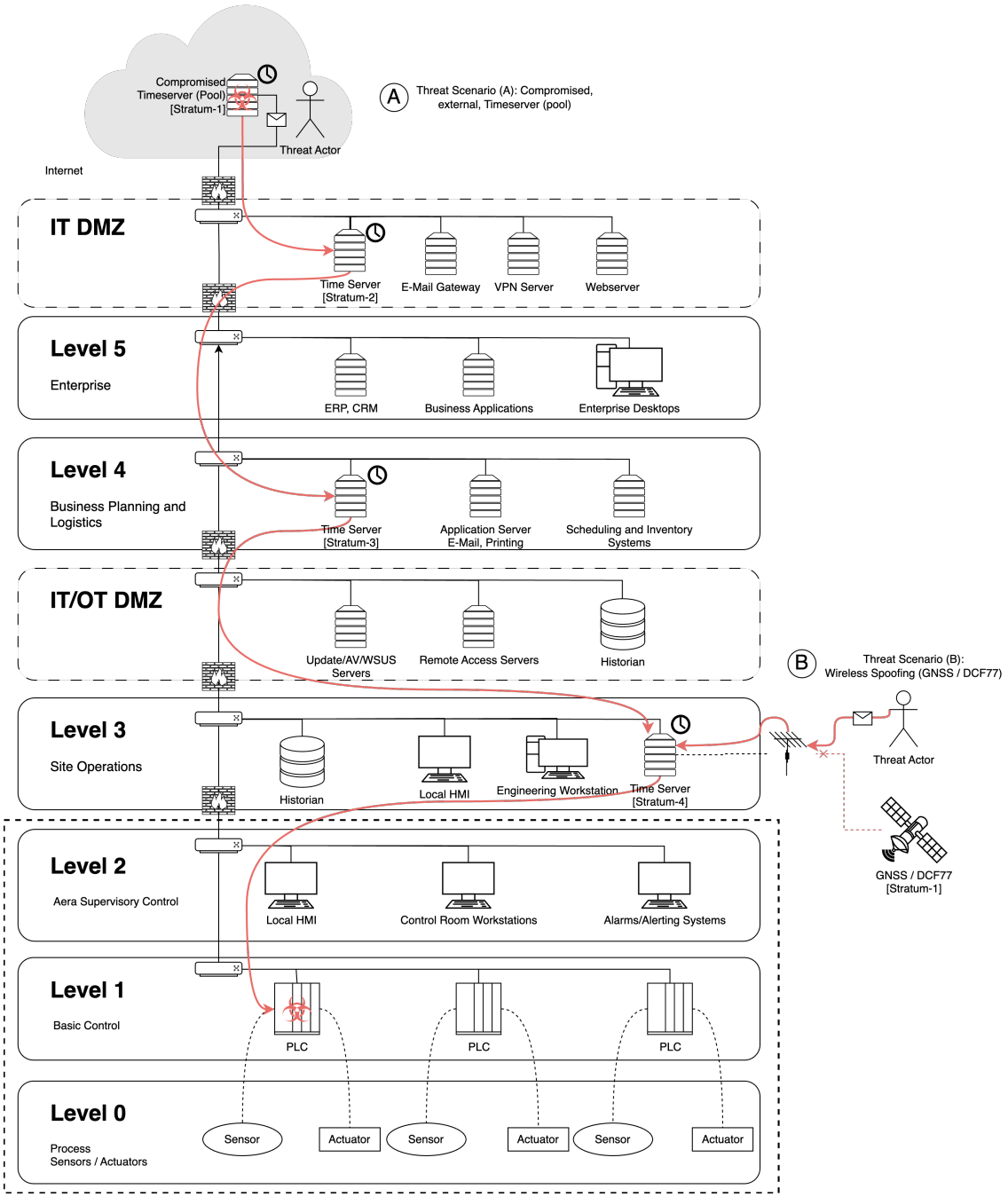


Figure 4.23: Potential threat scenarios (A) and (B) using time synchronization processes as carrier for hidden information. (A) compromise of an external time server. (B) Wireless spoofing attacks on communication with GNSS or radio signals transmitting time signals of atomic clocks, e.g., DCF77. Illustration of covert adversarial communication at the example of the Plausible Reference Architecture (*PlauRA-1*). Threat scenarios based on [LHKD22].

4.3.3.1 Threat Scenario (A): External Internet-based Synchronization

In this scenario, the time synchronization takes place using an external time synchronization server that has access to an atomic clock (Stratum 1). As illustrated in Figure 4.23, a central corporate time server (Stratum 2) in the IT DMZ, synchronizes its clock with this external server. Another time server is found on PERA-level 4 and/or 5 (depending on the architecture) in the Enterprise IT domain (Stratum 3), providing a local time source for servers and computers on the corresponding levels. This local time server is required, so that clients of Level 4 and Level 5 are not required to communicate with the central corporate time server in the IT-DMZ, by that reducing vertical network traffic providing better security. In Figure 4.23, it is exemplary placed on Level 4. From there, the OT time server on Level 3 can synchronize its clock (Stratum 4), providing a local time source for all OT components on Levels 1-3. Another option is to place an additional time server in the IT/DMZ to synchronize the time servers on IT and OT segments. For these threat scenarios this option has no significant impact and therefore left out in the scenarios illustrated in Figure 4.23.

In the illustrated threat scenario, a PLC is compromised on Level 1 in the architecture. The next step for the adversary is to compromise the used time server pool e.g., by adding own malicious NTP servers to a public pool or by taking control over them directly. This scenario involves proper reconnaissance to be able to tell which NTP pool is used by the corporate time server. Moreover, this scenario requires a full scale attack on the selected NTP server pool. These circumstances render this scenario rather difficult and require high amounts of resources. Though, as such capabilities are often found in the context of Advanced Persistent Threats and are not unusual in real-world incidents (as such a scenario is quite similar to the scale of known supply-chain attacks in the past) the described threat scenario seems plausible in that case, too. For example, the solar storm supply chain attack[13] is quite similar in terms of capabilities and scale and illustrates the imminence of such threats. Using one of the deep layer covert channels described above, the adversary is then able to secretly infiltrate information into the perimeter network, the time servers acting as unwitting proxies, passing down the information within the synchronization information towards the compromised PLC on Level 1. The communication between each time server is legitimate, so firewalls would not interfere.

The primary cause of this potential threat is the implicit trust between the servers. This architecture is problematic as it can unintentionally transmit unsafe data through the network. This is a common risk in defense-in-depth structures and must be taken into account when designing the architecture. Time synchronization processes are faced with the additional challenge that, as operating one's own atomic clock is too expensive and complex, external clocks and communication are usually necessary. Therefore, Scenario (B) covers an alternative approach to achieve time synchronization without relying on information from the Internet and proxied through the network.

[13] https://unit42.paloaltonetworks.com/solarstorm-supply-chain-attack-timeline/

4.3.3.2 Scenario (B): Radio-based Synchronization

In this scenario, the time server on the OT network is equipped with modules and antennas to receive radio signals containing timing information derived directly from atomic clocks (Stratum 1). In general, there are two options: Option (1) is to use time signals from satellites of Global Navigation Satellite Systems (GNSS), such as GPS, GLONASS, and Galilieo. Option (2) is to use terrestrial radio signals, for example, DCF77 in Germany or WWV in the US. Some devices include both options as a fail-save. Figure 4.23 illustrates this scenario with the corresponding attack to infiltrate covert information. Although radio-based time synchronization does not require any Internet access, it is susceptible to attacks on radio transmissions. What on the first glimpse seems to be a more secure option turns out to require even less resources and capabilities to attack. To jam the legitimate radio signal and spoof the radio timing signals, the adversary only has to be in the vicinity of the target facility and use cheap and publicly available hardware and software. For example, a parking lot in front of the target building would be fully sufficient to jam the radio signal originating from global navigation satellite systems or radio stations. As such radio systems use common frequencies, cheap jamming devices are publicly available and can be implemented using low-cost software defined radios (SDRs)[14]. By jamming the original signal, the adversary achieves that the original reference time is not reaching the corporate time servers. Instead, the adversary encodes a hidden message in the timestamp (e.g., covert channel UV1c of [LHKD22]) and emits the modified signal. By jamming the original signal and transmitting the modified signal near the antenna, the adversary can ensure that only the modified signal is received [Ebi21]. Once the OT time server is synchronized with the injected time signal, this information containing the hidden information is passed unwittingly to all clients on the OT network, in this scenario including the compromised PLC.

[14] https://greatscottgadgets.com/hackrf/

4.3.4 Covert Channels in the Precision Time Protocol (PTP)

NTP and PTP have many similarities on a functional level, but their protocol fields and communication flows differ significantly. The main similarity between them is the transmission of high-resolution timestamps. The main difference is that PTP has better accuracy and precision due to specialized hardware and high accurate clocks, with an estimated accuracy of 200 nanoseconds in the best case scenarios. Fluctuations in the 32-bit nanosecond timestamp follow a Gaussian distribution [End16], leaving an adversary with 7 bits of fluctuations to embed hidden information. Keeping the modifications small and within usual fluctuations and inaccuracies makes it hard to detect, as previously shown for NTP. In the case of PTP, staying within the usual distribution of inaccuracies is essential, as improperly synchronized clocks can have an impact on Operational Technology and the controlled processes.

The bandwidth that results from the intervals between messages used for embedding is typically 7 bits per second or 48kb per minute when the `Sync` messages are sent every second. This may seem like a low bandwidth, but it is enough to transfer malware components into a target network within a day, which is suitable for many attack scenarios, especially in the OT domain, which can take weeks, months, or even years. Other covert channels, such as storage channels in the header fields, may be possible, but when analyzing publicly available PTP traffic (e.g. [Wir21]), it is evident that modifications to the `sourcePortIdentidity` would either disrupt the communication or be easily detectable.

Timing channels may be a viable option for covert communication, such as the modulation of message timing in `Sync` messages. However, these channels are mostly protocol-agnostic and implementation-dependent, and they usually have a low bandwidth, such as 1 bit/second in the case of `Sync` messages. Additionally, timing channels can interfere with the time synchronization process, as they may introduce non-deterministic delays in the communication between clocks. In conclusion, PTP is vulnerable to covert channels, similar to NTP. The use of high-resolution timestamps makes PTP attractive for covert message injection, but the higher accuracy of PTP can also be affected by covert channels if not implemented carefully by an adversary. In many instances, the use of covert channels in PTP necessitates prior reconnaissance to tailor the covert channel to the actual implementation. To remain undetected, an adversary must observe PTP traffic and evaluate which PTP fields can be used for storage channels and how much alteration of the timestamps can be done without disrupting the time synchronization process. Consequently, such covert channels are most likely to be employed in situations with a great deal of resources, such as Advanced Persistent Threats (APT).

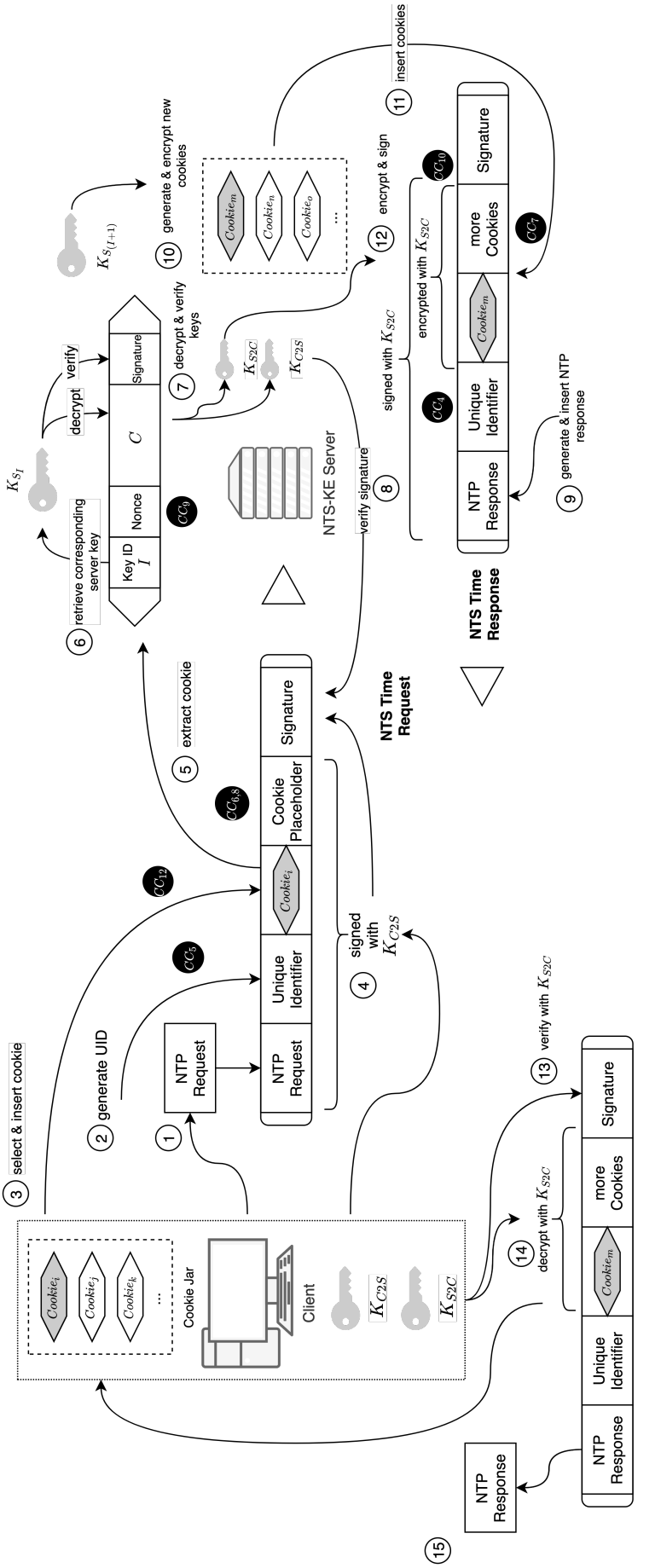


Figure 4.24: Overview of the NTS Time Synchronization Process and NTS time request and response packet structure based on RFC8915 [FST+20, Net20]. Black circles reference to the covert channels in Table 4.14. Figure from [LD22].

4.3.5 Covert Channels in Network Time Security (NTS)

As NTS is basically an extension to NTP and wraps around NTP frames, the previously identified covert channels for NTP are also applicable to NTS. Therefore, in the following the focus is set to NTS-sepcific covert channels. Figure 4.24 illustrates the method of operation for NTS and which features can be used to establish covert channels. In a theoretical analysis, published in [LD22], in total 12 storage covert channels and one stego key exchange method are discussed in both sub-protocols NTS Key Establishment (NTS-KE) and NTS Extension Fields (NTS-EF) with three covert channels in NTS-KE and nine in NTS-EF. From these 12 covert channels, one is additionally practically verified in [LD22] and four more by a team of students [FGCK23].

The results of these investigations are summarized in Table 4.14. The capacity of the identified covert channels ranges drastically from 3 bit up to 6.4kb per packet. However, similar to NTP the bandwidth in general is limited due to the circumstance that the time synchronization process is commonly only run every 60 seconds. Therefore, for the NTS-EF covert channels the capacity tends also to be the bandwidth per minute. In case of the NTS-KE sub-protocol (NTS-KE), the bandwidth is even smaller, as the key establishment is (typically) performed only once per boot-up. This significantly reduces the available bandwidth for both NTS-EF and NTS-KE covert channels. On the other hand, as these covert channels leverage encrypted features, they tend to be very difficult to detect and mitigate and require the deployment of active wardens, which will be discussed later in the section on mitigation.

In the following, the covert channels that were implemented and validated are described in more detail.

4.3.5.1 Covert Stego Key Exchange ($Sync_1$)

The NTS-KE sub-protocol was found to have an interesting feature that could be exploited for the covert exchange of stego-keys and other key material. This is based on a TLS1.3 connection with a handshake for key exchange. A stego-malware could take advantage of this key exchange to obtain key material for its own use. The TLS Key Export, as defined in RFC5705 [Res10], is used to generate a client-to-server key K_{C2S} and a server-to-client key K_{S2C}. According to RFC8915 [FST+20], the NTS server and client may use this function to derive further keys for additional functionality. This could be abused to derive a shared secret (stego key) K_{Steg} to control the covert embedding and retrieval functions of other covert channels. As this key export is a key derivation performed only by the server and client, a warden cannot detect or prevent this key exchange. This covert method for key exchange is illustrated in Figure 4.25. In a practical proof-of-concept implementation [FGCK23], this method could be successfully validated to be effective and undetectable for an outside warden monitoring network traffic. This insight is very important, as it is not limited to NTS but **any protocol incorporating TLS 1.3 Key Export implementing RFC7505 [Res10]** as this covert technique is inherent and required for the key export and therefore cannot be mitigated.

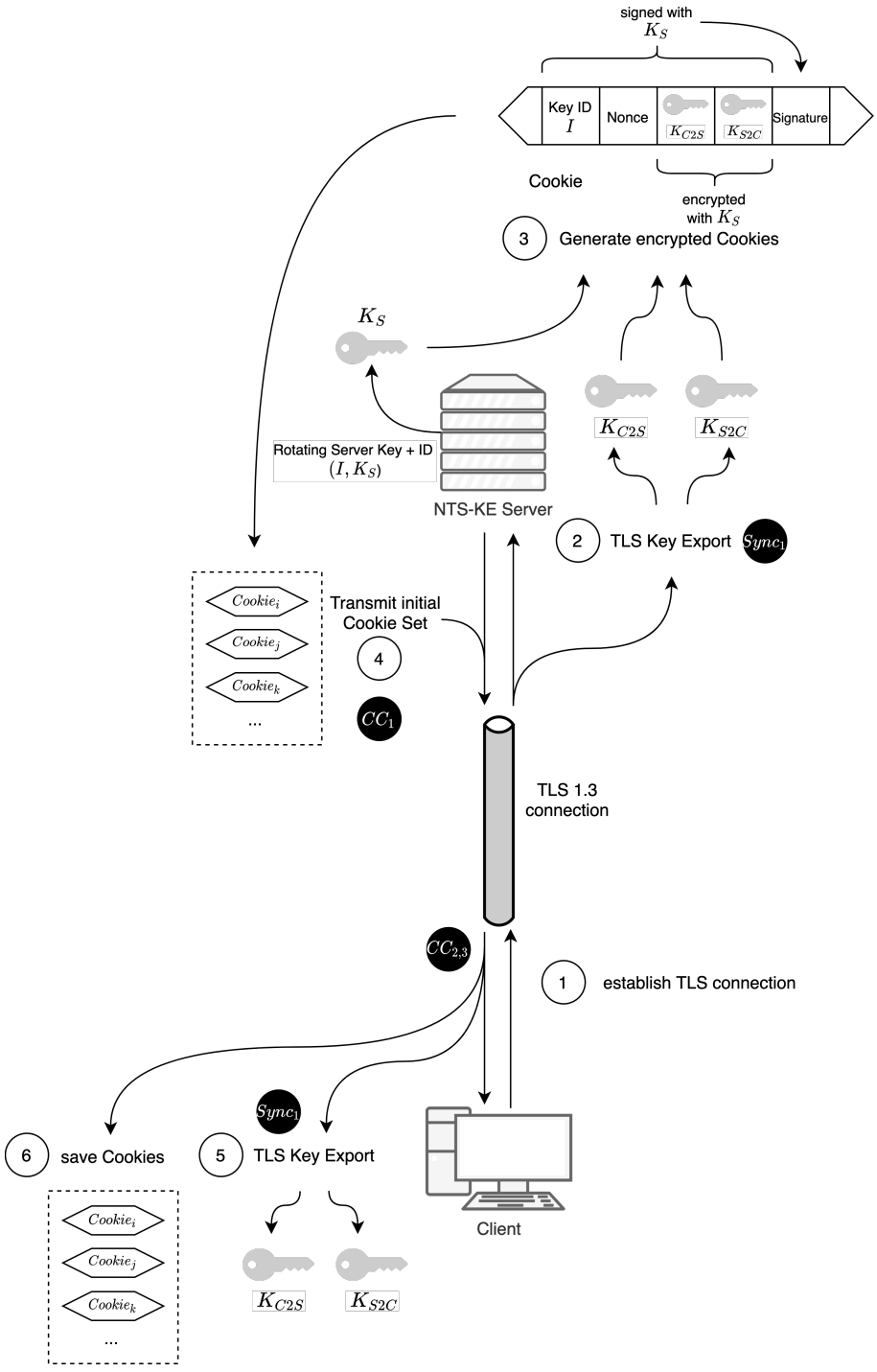


Figure 4.25: Overview of the NTS Key Establishment Process and cookie structure based on RFC8915 [FST+20, Net20]. Black circles reference to the covert channels in Table 4.14. Figure from [LD22].

Table 4.14: List of identified Covert Channels in NTS (RFC8915 [FST+20]). The Sub-Protocol column describes whether the covert channel is applicable in the NTS Key Establishment Phase (NTS-KE) or to NTS Extension Fields (NTS-EF). The direction column describes whether the covert channel is applicable for infiltration ($\rightarrow$), exfiltration ($\leftarrow$) or Command and Control ($\rightleftarrows$). The related Pattern column describes which pattern of the revised network covert channel pattern taxonomy fits the covert channel [WCM+21]. The Protocol-Compliance column indicates whether the covert channel is protocol-compliant. Bold row indicates practically validated covert channel.

Covert Channel ID	Sub-Protocol	Direction	Description	related Pattern	Capacity (per packet)	Proto.-Compliance
CC_1	**NTS-KE**	$\rightarrow$	**Initial Cookie Set Modulation**	$EN3$ **Enumeration**	**3 bit**	✓
CC_2	NTS-KE	$\leftrightarrows$	NTPv4 Server Modulation	$EN4$ Value Mod.	2024 bit	✓
CC_3	NTS-KE	$\leftrightarrows$	NTPv4 Port Modulation	$EN4$ Value Mod.	16 bit	✓
$Sync_1$	**NTS-KE**	$\leftrightarrows$	**TLS Stego Key Export**	-	-	✓
CC_4	**NTS-EF**	$\leftarrow$	**Client-side UID Modulation**	$EN4.2$ **Random**	**256 bit**	✓
CC_5	NTS-EF	$\rightarrow$	Server-Side UID Modulation	$EN4.2$ Random	256 bit	✗
CC_6	**NTS-EF**	$\leftarrow$	**Requested Cookies Modulation**	$EN3$ **Enumeration**	**3 bit**	✓
CC_7	NTS-EF	$\rightarrow$	Responded Cookies Modulation	$EN3$ Enumeration	3 bit	✗
CC_8	NTS-EF	$\leftarrow$	Cookie Placeholder Overwrite	$EN4.1$ Reserved	$\sim$ 800 bit	✗
CC_9	NTS-EF	$\leftrightarrows$	AEAD Nonce Modulation	$EN4.2$ Random	76-104 bit	✓
CC_{10}	NTS-EF	$\leftarrow$	Encrypted Content Modulation	$EN4$ Value Mod.	$\sim$ 6400 bit	✗
CC_{11}	**NTS-EF**	$\rightarrow$	**Client-readable Cookies**	$EN4$ **Value Mod.**	$\sim$ **800 bit**	✓
CC_{12}	NTS-EF	$\leftarrow$	Client-side Cookie Modulation	$EN4$ Value Mod.	$\sim$ 800 bit	✗

4.3.5.2 Initial Cookie Set Modulation Covert Channel (CC_1)

In the NTS-KE phase the server provides an initial set of cookies (see Figure 4.25, Step 4) for the client. This initial amount can be modulated to encode hidden information. When varying between one and eight cookies, this results in a *capacity of three hidden bit*. According to RFC8915 the amount of cookies is implementation-dependent. In practice, eight cookies seem to be the standard[15] to provide the client with a proper amount of cookies while at the same time avoiding UDP fragmentation. This assumption could be verified in a practical evaluation [FGCK23]. A deviation from this standard may raise suspicion or could trigger anomaly detection systems. However, as the communication is encrypted using the TLS connection, the amount of cookies is not directly observable without intercepting the encrypted communication. Therefore, a passive warden without TLS inspection is not able to detect the covert channel. The direction in this case, is from server to client (*infiltration*) and the channel itself is *protocol-compliant* as the amount of cookies is implementation-dependent. In terms of the pattern-based taxonomy, this covert channel is reflected by the *EN3 Elements/Features Enumeration* pattern.

4.3.5.3 NTS-UID Covert Channel (CC_4)

This covert channel makes use of the NTS Unique Identifier Field, which is randomly set by the client. The embedding and retrieval process of this specific covert channel is illustrated in Figure 4.26. The core concept is to use a deterministic pseudo-random-number-generator (PRNG) with the stego key K_{Steg} as the seed (1) to generate a pseudo-random NTS-UID (2) consisting of 32 random bytes. Step (3) involves splitting the secret message into 32 byte-sized chunks and embedding it into the pseudo-random UID using XOR (4). This results in a stego-key seeded pseudo-random UID containing the embedded message. This UID is then inserted in a NTS time request and sent to the server (5). To transfer the next message chunk, the PRNG is reseeded (6) with the previously generated pseudo-random UID from Step (2). This resembles the Output-Feedback mode (OFB), known from block ciphers, and allows for the provision of fresh pseudo-random IDs based on the initial secret. In Step (7) the next message chunk is then embedded by XOR into this new UID and sent within the next NTS time request. For retrieval on the server side, the server seeds the PRNG with the shared secret K_{Steg} (8) and generates a pseudo-random ID as well. To retrieve the plaintext, the server extracts the UID from the client's NTS time request (9) and performs an XOR operation on the extracted UID and the pseudo-random ID it generated with the stego key (10). To retrieve the next message block, the server seeds the PRNG with the previous pseudo-random ID (11), generates a new pseudo-random ID, extracts the next UID from the client's time request and performs another XOR operation on both the UID from the client and the newly generated UID from the server (12).

A different approach to this covert channel would be to not apply XOR to the entire UID, but only to certain parts of it. This would enable the secret message to be concealed in various places within the random UID (thereby providing a second parameter or stego key) and further reducing the chances of detection.

[15] see chrony (https://chrony.tuxfamily.org/index.html), ntpsec (https://www.ntpsec.org/) and [Cho21]

This design of a covert channel has been validated using an open-source Python implementation. Two data sets were created: DS_{true} which contains 10,000 cryptographic secure random UIDs based on Pythons `os.urandom` function, and DS_{pseudo} which contains 10,000 pseudo-random UIDs with embedded messages. The covert channel was implemented using Pythons PRNG implementation from the `random` module, which implements a Mersenne Twister PRNG algorithm. The PRNG was initially seeded with a static string (100 characters) as K_{Steg} and generated 10,000 pseudo-random IDs with Caesar's De Bello Gallico[16] embedded as a hidden message using the method described before. The complete transfer of the text (169kb) took roughly 12 hours in the test environment (256 bits per UID, 62 seconds between time requests using chrony). Statistical methods (entropy, runs-test, chi-squared) and rand-test[17] were not able to differentiate between the original and embedded UIDs.

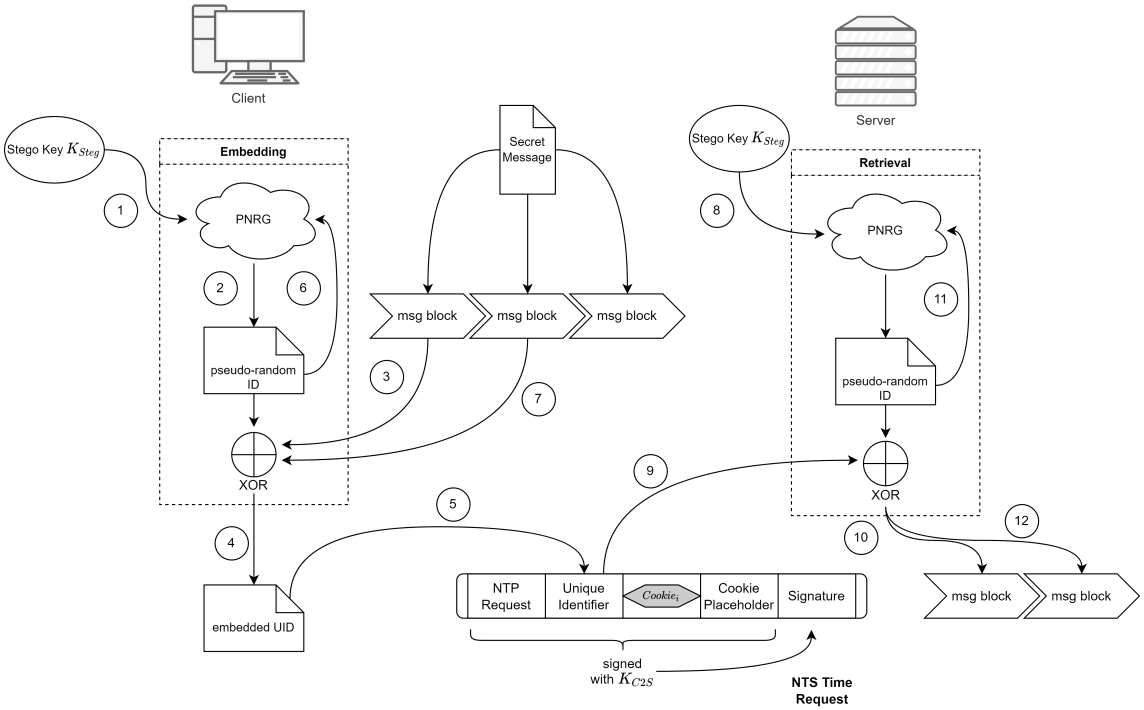


Figure 4.26: Overview of the embedding and retrieval processes for the NTS Covert Channel *Client-side Unique Identifier Modulation*. Figure from [LD22].

4.3.5.4 Requested Cookie Modulation Covert Channel (CC_6)

To avoid having to run the NTS Key Establishment again, clients can request new cookies from the server by including zero-filled Cookie Placeholder fields. In the testing environment, the default are eight cookies. In a similar way to CC_1, a covert sender can vary the number of requested cookies to encode hidden information. By ranging from zero to eight cookies, *three bits* of data can be encoded. This covert channel is *protocol-compliant*, as the specification does not explicitly state the amount of cookies. To detect this channel, a (passive or active) warden is needed to monitor the amount of commonly requested cookies to identify any anomalies.

[16] https://www.gutenberg.org/ebooks/218
[17] https://github.com/sudo-rushil/randtest

4.3.5.5 Client-readable Cookies Covert Channel (CC_{11})

As demonstrated in Figure 4.24, NTS-cookies contain the client-server keys and are only readable by the server as they are encrypted using the secret server key K_S. A covert channel between a malicious server and client can be established when the server encrypts the cookies with a shared secret between the two, such as a stego key derived in the NTS-KE (Key Establishment) as described in $Sync_1$ and inserts the hidden message within the encrypted cookies. Alternatively, the cookies can be encrypted with one of the symmetric client-server keys (K_{S2C}, K_{C2S}). This allows for a covert channel with 800 covert bits per cookie (*6.4kb per packet*) from the server to the client. As the cookies are encrypted using a key only known to the rogue server and client, even active wardens capable of TLS-inspection are unable to detect or prevent this covert channel. This channel is represented by the *EN4 Value Modulation* pattern. Although this covert channel clearly violates the specification, it can be considered *protocol-compliant* in the sense that an observer/warden inspecting the NTS packets would not be able to differentiate such stego-cookies from regular cookies. The concept of this covert channel and the resulting capacity could be practically verified by a team of students [FGCK23].

4.3.6 Detection & Mitigation

As previously described and discussed in [LHKD22], detection for covert channels is yet to be investigated in more depth in case of NTP. However, from a theoretical standpoint and first experiment it appears to be a non-trivial task with high efforts as most covert channels make use of high-entropy or even random fields. Therefore, here, we focus on potential mitigations using protocol normalization.

4.3.6.1 Protocol Normalization for NTP

It has been previously suggested in [FM19] that the stratum, Root Delay, Root Dispersion, Reference ID, Reference timestamp, Origin timestamp, and Receive timestamp in NTP client packets should be set to zero. This, combined with the findings of [HLKD21] regarding chrony's default behavior, indicates that NTP client packets can be normalized. The version, mode, and transmit timestamp fields are the only ones that need to be present in NTP client requests, and all three can be set to static values (chrony sets the transmit timestamp to a random value, but this can be changed). This means that a traffic normalizer can be used to prevent covert channels in one direction. Normalization of server (and broadcast) packets is more limited: the root dispersion, root delay, reference timestamp, and origin timestamp are necessary for accurate time calculation. All other fields can be normalized, and at least the last bits of the timestamp fields can be normalized without significantly affecting the time calculation. This reduces or eliminates the high entropy. NTP packets should be normalized to a size of 48 bytes, and extension fields and the MAC field should not be allowed, further reducing the attack surface. This is also true for NTP control queries [HLKD21].

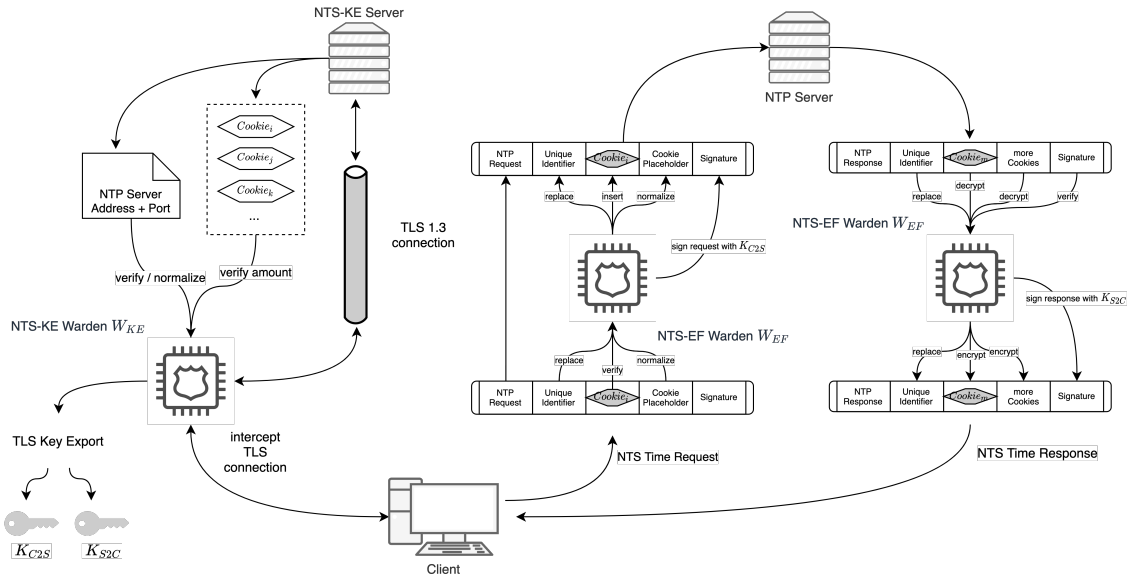


Figure 4.27: Overview on the processes of the active warden mitigating the described covert channels in NTS. Figure from [LD22].

4.3.6.2 Active Warden Design for NTS

In [LD22], an active warden is proposed to address the majority of identified covert channels. This concept is illustrated in Figure 4.27 and consists of two components, one for each sub-protocol. The NTS-KE warden (W_{KE}) acts as a man-in-the-middle to validate the amount of initial cookies (CC_1) and NTPv4 Server address (CC_2) and UDP port (CC_3) using anomaly detection or normalization. Normalization would involve verifying that the server responds with the configured address and UDP port as well as the configured amount of cookies. If there is any deviation from these defaults, the warden would overwrite the affected values, eliminating any hidden communication. The NTS-EF warden (W_{EF}) also acts as a man-in-the-middle server. To prevent the UID modulation covert channels CC_4 and CC_5, the warden replaces the UIDs generated by the server and client with true random IDs generated by itself. This requires the warden to sign the NTS request and response using the respective client-server keys (K_{C2S} and K_{S2C}) obtained from NTS-KE interception. To detect the channel, the warden needs to run randomness tests such as DieHard [Ala10], DieHarder [BEB18] or TestU01 [LS07]. Alternatively, PRNG attacks can be used to clone the PRNG and predict the next pseudo-random number. In [Kop20], it could be shown that 624 consecutive outputs are sufficient to clone a Mersenne Twister PRNG. The Cookie Amount Modulation Covert Channels CC_5 and CC_6 can be mitigated by enforcing a specific amount by the warden. CC_8 can be mitigated by normalizing the cookie placeholder fields to all zeros (any deviation would indicate a potential covert channel). CC_{10} Encrypted Content Modulation can be mitigated by using the server-to-client key K_{S2C} to decrypt the contents of the NTS response and to verify its content (it should only contain cookies). To mitigate CC_{12}, the warden has to keep track of which cookies have been issued by the server (requiring K_{S2C}) and which cookies have already been used by the client, verifying that cookies are only used once.

This design for a warden has yet to be tested in actual deployments. Although it appears to be a viable option, it is both costly and complex. The fact that the entire cryptographic system must be broken by the warden demonstrates the difficulty of defending against covert channels in the encrypted domain. This emphasizes the need for further research into simpler warden designs, as well as methods for detection on encrypted streams to avoid the need to break the used cryptography.

Covert Channel ID	Sub-Protocol	Description	Proto.-Compliance	Detectable by W_{PV}	Detectable by W_{KE}/W_{EF}	Overall Detectability	Mitigation by W_{KE}/W_{EF}
CC_1	NTS-KE	Initial Cookie Set Modulation	✓	○	⦿	⦿	✓
CC_2	NTS-KE	NTPv4 Server Modulation	✓	○	⦿	⦿	✓
CC_3	NTS-KE	NTPv4 Port Modulation	✓	○	⦿	⦿	✓
$Sync_1$	NTS-KE	TLS Stego Key Export	✓	○	○	○	✗
CC_4	**NTS-EF**	**Client-Side UID Modulation**	✓	○	○	○	✓
CC_5	NTS-EF	Server-Side UID Modulation	✗	●	●	●	✓
CC_6	NTS-EF	Requested Cookies Modulation	✓	⦿	⦿	⦿	✓
CC_7	NTS-EF	Responded Cookies Modulation	✗	●	●	●	✓
CC_8	NTS-EF	Cookie Placeholder Overwrite	✗	●	●	●	✓
CC_9	NTS-EF	AEAD Nonce Modulation	✓	○	○	○	✓
CC_{10}	NTS-EF	Encrypted Content Modulation	✗	○	⦿	⦿	✓
CC_{11}	NTS-EF	Client-readable Cookies	✓	○	○	○	✗
CC_{12}	NTS-EF	Client-side Cookie Modulation	✗	○	⦿	⦿	✓

Table 4.15: Comparison of mitigiation methods for the identified Covert Channels in NTS. Detectable by W_{PV} indicates the warden-compliance against a passive warden. Detectable by W_{KE}/W_{EF} indicates the warden-compliance against the active wardens with the same symbols as for W_{PV}. The Overall Detectability column gives an indication on how hard the covert channel is to detect based on the combination of active and passive warden-detectability. The last column indicates whether mitigation is possible using blind normalization performed by the active wardens W_{KE}/W_{EF}. A white circle indicates that detection is not possible, a black circle indicates that a rule-based detection scheme is sufficient to detect the channel, ⦿ indicates that anomaly-/ ML-based detection is required. Bold line indicates implemented and tested channel.

4.3.6.3 Proposal of a Time Sychronization Architecture resilient to Covert Channels

This proposal outlines an architecture for a time synchronization process, exemplified by the Plausible Reference Architecture (PlauRA-1), that is designed to be more resilient to covert channels. This architecture is illustrated in Figure 4.28. One key aspect is to diversify the use of protocols and use all prior discussed protocols: NTS is used to communicate securely with time servers on the Internet, NTP is used for time synchronization of IT systems (instead of NTS due to easier inspection methods). PTP is used for OT components, given an extra line of separation to the IT components using NTP. These protocol conversions break some covert channels and remove the implicit trust between the servers. The architecture is not only built on diversity but redundance for detection purposes as well. In the following the process steps are described:

Beginning with the IT side, the time server in the IT-DMZ obtains its reference time from a reliable, public Stratum-1 server pool using NTS over the Internet. This time signal is authenticated by both a GNSS source and a radio source, similar to the safety architecture of airplanes, which eliminates outliers using a majority vote. This process does not prevent covert channels, but makes them more difficult to establish and operate without detection. In the second step, the time synchronization between the time server of the IT-DMZ and the NTP server located in Levels 4 and 5 of the reference architecture is examined and normalized, eliminating potential covert channels in the NTP fields. In Step 4, the central OT time server receives its clock reference from two independent sources (GNSS and radio) and verifies these time signals against the time used in the IT network using a dedicated neutral time verification server located in the IT/OT DMZ (Step 5). This verification server constantly checks that the time signals of IT and OT are consistent. In Steps 6 and 7, PTP communication, which is subject to packet inspection and normalization, is used to distribute the time into the different OT zones. By this, the IT and OT time synchronizations are decoupled yet used to verify each other.

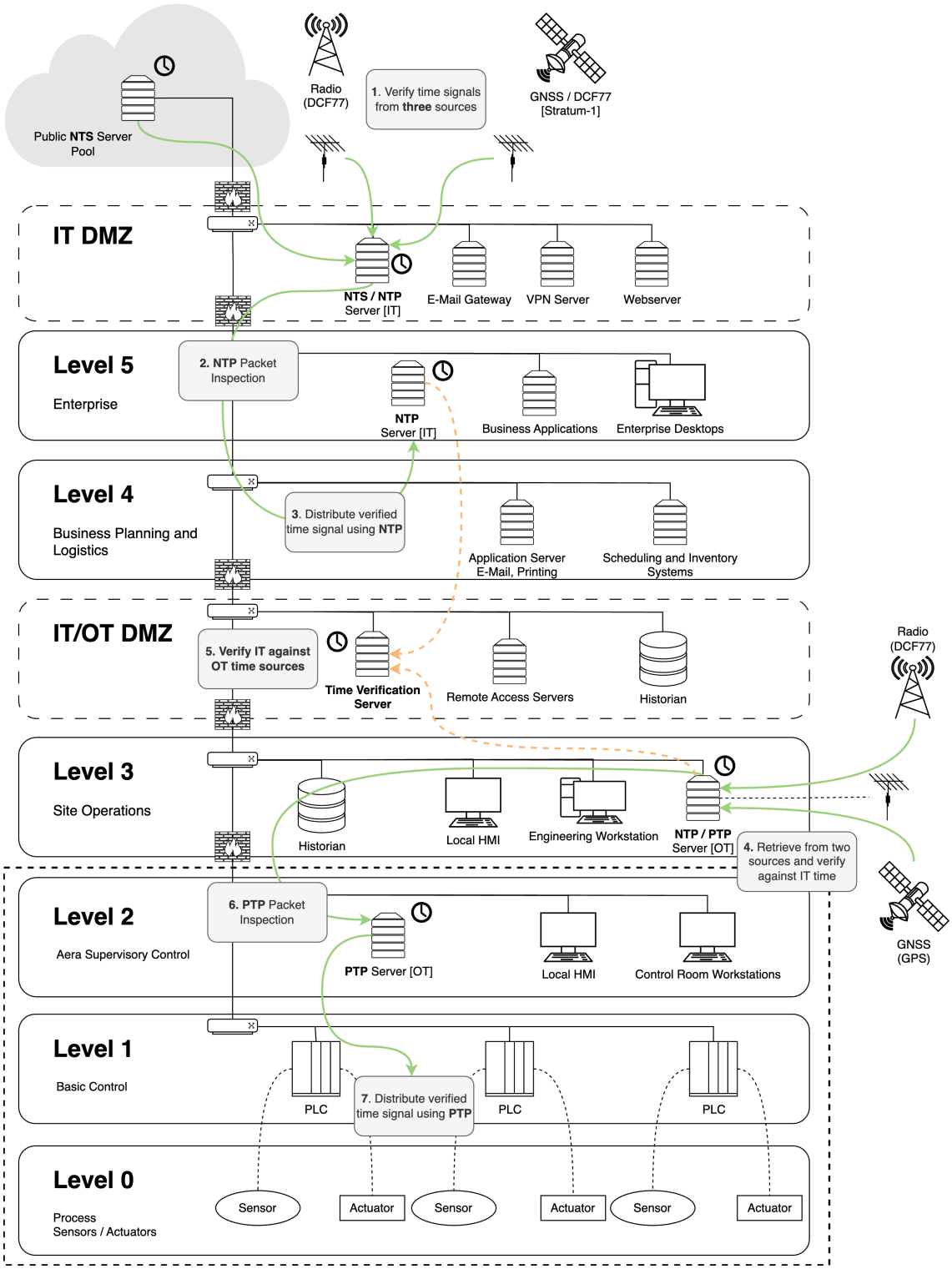


Figure 4.28: This proposal outlines an architecture for a time synchronization process, exemplified by the Plausible Reference Architecture (*PlauRA-1*), that is designed to be more resilient to covert channels.

4.3.7 Key Insights from the Time Synchronization Case Study

This section summarizes the key findings derived from the Time Synchronization case study. These key findings are relevant for the field of Information Hiding in CPS in general or have implications for other case studies. The following selected insights can be drawn from the case study:

Time Synchronization Protocols as Cover This case study demonstrates the potential for hidden information to be transmitted through seemingly innocuous infrastructure protocols such as NTP, PTP and NTS. Time synchronization is a complex process that requires the transmission of high-entropy data, often with more precision than the hardware is capable of, leaving enough room to establish covert channels in timestamp fields. The hierarchical structure and implicit trust between time servers also contribute to the threat of covert channels, as they can be used as proxies to traverse firewall-segregated networks. Furthermore, such infrastructure protocols are typically not monitored closely by IDS/IPS systems, as they are not commonly used in conventional attacks. This highlights the importance of paying attention not only to obvious carriers such as process communication and data, but also to seemingly harmless infrastructure communication.

Encrypted Protocols as Cover and the Encryption Paradox The case study of NTS demonstrates the substantial effect of cryptography on IH-based threats. The purpose of introducing encryption to a protocol is to secure communication and protect against certain threats. For instance, it reduces the risk of Man-in-the-Middle attacks, which also includes covert channels (Passive or Hybrid Information Hiding). However, as NTS has shown, it can also create a plethora of possibilities for covert channels in an A-B situation, i.e. covert channels between the endpoints using encrypted communication (Active IH). Even with a warden capable of TLS-inspection, some covert channels remain undetectable. This is a common pattern in encrypted protocols as these usually require some kind of randomness, which can be used for information hiding. At first glance, introducing cryptography to a protocol may appear to be a viable solution for thwarting covert channels. However, this case study revealed the opposite to be true. While it is true that certain covert channels may be more difficult to set up when communication is encrypted, the additional overhead can create new opportunities for establishing covert channels. This presents a paradoxical situation, as the integration of cryptographic measures to protect against threats can create opportunities for other threats, such as covert channels. It is essential to recognize that the implementation of security measures can have unintended consequences and side effects that must be taken into account when conducting a risk assessment. This is a key factor to consider when designing security measures and analyzing the associated risks.

TLS Key Export as a general method for Stego-Key Exchange The RFC5705 TLS Key Export [Res10], a key feature of TLS 1.3, has been demonstrated to be an effective way of providing shared key material for a secret sender and receiver to generate secure stego keys. This has implications not only for NTS,

but for any protocol that incorporates such key exports based on TLS. The architecture of this protocol makes it impossible to stop this type of stego key exchange.

Detection & Mitigation This case study has revealed that it is often difficult or requires a lot of effort to detect covert channels in time synchronization. Normalization of the NTP protocol can be a good starting point, but it cannot protect against more complex channels in the timestamps. It is recommended to create a secure architecture that limits the potential for covert channels. Such an architecture is proposed in Section 5.2.3. This architecture should involve decoupling, breaking the implicit trust between servers, using each protocol for a different purpose (diversity) and providing redundancy (e.g. radio, satellite and Internet synchronization at the same time). This redundancy can then be used to detect any irregularities. For NTS, the cryptographic features require more advanced inspection techniques and the use of active wardens to reduce the number of identified covert channels.

5. Conclusion & Key Insights from the Case Studies

This chapter provides a summary on which key insights can be drawn from the case studies and also gives a brief comparison between the case studies.

5.1 Methods of Information Hiding in CPS

This section provides the key insights that can be drawn from the case studies and puts them into perspective.

5.1.1 Distinctive Features of CPS with impact on Information Hiding-based Threats

As discussed in Chapter 3 and demonstrated in the case studies, Cyber-Physical Systems (CPS) are intricate systems that bring together components and communication from multiple domains. This complexity allows for the application of various forms of Information Hiding, such as network covert channels in communication between CPS components, media steganography in image transmissions (e.g., surveillance cameras), Out-of-Band Covert Channels using side-effects of physical processes as carriers, steganography in process data, and text steganography in log files. The classification of Information Hiding methods in Cyber-Physical Systems (cf. Chapter Section 3.3) provides a structure to this complexity and highlights the potential threat of Information Hiding in CPS.

In terms of Information Hiding, the different types of devices and components represent a whole set of potential covert sender and receiver, allowing for various active, passive and hybrid information hiding scenarios (cf. Section 3.5.1.1).

In the Operational Technology (OT) domain, the communication between components is more uniform than in the Information Technology (IT) domain due to the cycle-based computation on logic controllers. This can be beneficial for malicious

actors as it is simpler to predict the communication in a target environment. However, this also means there are fewer possibilities for embedding. From a security standpoint, the homogeneous and less complex communication is easier to monitor and detect anomalies. Despite this broad statement, the Modbus/TCP case studies also demonstrated that the actual use of (actually simple-structured) protocols features in target environments can be so varied from system to system, that prior reconnaissance and adaptation of the covert channel and its parametrization for the specific target is necessary. This also applies to the defensive measures and detectors which have to be adjusted for the environment.

The operational technology (OT) domain has a much longer lifespan for its components than the information technology (IT) domain. Whereas IT components are typically replaced after a few years, OT components can be used for decades. This can create a heterogeneous environment when older, legacy components are operated alongside newer components. It is not uncommon for OT to operate devices with known security vulnerabilities, as patches may not be available or deployed in order to avoid disruption of operations. Because of the presence of legacy components or to ensure compatibility between devices and components from different manufacturers, unsecured protocols are frequently employed. These factors not only facilitate cyber attacks (and the use of Information Hiding) but also make detection a lot harder, as these differences between devices and their behavior can be leveraged to establish covert communication channels.

Apart from the process communication, the case studies also demonstrated that infrastructure protocols also play a major role for covert channels, yet are underrated in common IT security considerations. Another factor is the inherent trust between some components that can be leveraged, as shown as well in the case studies.

5.1.2 Application of Information Hiding Patterns for Covert Channel Discovery

The application of known patterns of Information Hiding methods (in case of the Modbus/TCP and NTP studies the Network Information Hiding Taxonomy by Wendzel et al. [MWC18] and for NTS the revised patterns [WCM+21]) to the specification of network protocols has demonstrated to be a reliable method to identify potential covert channels in protocol specifications. The resulting list of identified covert channels might not be exhaustive and subjective, yet give a solid foundation for further analysis which can verify the potential covert channels and their limitations. These can also spark ideas for other, previously unidentified, covert channels.

5.1.3 Differences between Specification and Implementation

It became evident in the case studies that some of the potential covert channels identified are not feasible or are highly dependent on the target environment, which includes different devices, firmwares, hardware, and implementation variations. The specification leaves room for interpretation and implementation, so software from different vendors can have significant differences in their behavior, which can have a direct effect on the use of covert channels or even enable covert channels that are not

apparent from the specification. It could be shown that these differences between environments can lead to lower bandwidths, higher detectability, less robustness, or even the impossibility of using certain covert channels. Therefore, the covert sender and receiver need to be aware of the specific cover characteristics of the environment they are using. This means that prior reconnaissance is essential. This is an important factor as well for detection and mitigation strategies, as it requires the adversary to analyze the cover, i.e., take actions that may be detected even before covert channels are established. The same applies to the warden, who needs specific knowledge of the cover characteristics for successful detection and mitigation.

This difference between specification and implementation sparked the idea for *contextual plausibilty*, explained in the following.

5.1.4 Contextual Plausibility of Cover Objects in CPS

The contents of this section have been peer reviewed and published in part within the scope of the following publications as joint work with the co-authors Jana Dittmann, Jonas Hielscher, Christian Krätzer: [LD20, HLKD21, LD22].

In [HLKD21] we further diversified the concept of plausibility with the addition of **contextual plausibility**, which describes the plausibility of a covert channel in a given context:
A (network) covert channel is considered **contextually plausible**, if the modified packet or flow is **(1) syntactically and (2) semantically correct in the given context**.

While *protocol-compliance* describes the syntactical correctness, *contextual-plausibility* considers information of the context, e.g., vendor-specific implementations, network environment configurations and the point in time.
This is especially important, for example a network packet or (CPS-)event might be syntactically correct (or - in case of an event - expected), yet may raise suspicion, if it is semantically wrong or it occurs to a wrong point in time.

For example, in in the covert channel analysis of the Network Time Protocol (NTP) [HLKD21] (as part of Case Study CS_3), we identified covert channels that leverage the behavior of specific NTP-client implementations. For example, the client-implementation of *chrony* randomizes the *Transmit Timestamp* field, whereas other implementations use the actual transmission time. If this covert channel is used on overt traffic from other implementations, i.e. the transmit timestamp is modulated in the expectation of a randomized value, the covert channel becomes obvious, making detection trivial. Another interesting point in this example is, that the covert channel in both cases is protocol-compliant as the specification is not clear on this point, yet the implementations do differ. This might represent a general pattern as it occurred in multiple investigations, for example in the Modbus/TCP case study CS_1 and [LD20] and in the analysis of Network Time Security (NTS) [LD22] (as part of CS_3).

This also highlights the relationship of (contextual-) plausibility, protocol-compliance and the resulting warden-compliance. A covert channel might be protocol-compliant, yet in specific contexts is easier to detect than in others, leading to different levels of warden-compliance.

Therefore, it is important to distinguish, if a covert channel is dependent on specific contextual features. Another example from CPS-Steganography, is the modulation of sensor measurements. One implausible covert channel, for example, is the modulation of water flow measurements while the actual valve responsible for letting water flow through the sensor is closed. Such inconsistencies can make detection trivial, however, the warden needs to be capable of identifying such logical mistakes/inconsistencies (see Section 5.2.4 regarding details on the requirements for effective wardens in ICS).

5.1.5 Techniques for Distribution and Relocating/Recaching of Hidden Information

In the course of the process data transmission case study (CS_2) and in the corresponding publication ([LNK+21]), a categorization of techniques for distributing hidden information is proposed:

Cover Channel Selection/Change If multiple **cover** channels exist, one or more can be selected and changed over time.

Covert Channel Selection/Hopping If multiple **covert** channels are applicable to the cover object/channel, a covert channel can be selected or hopped between different ones. Similar to the pattern hopping approach discussed in [MWC18].

Embedding Position Selection/Change If multiple options exist, change the embedding position within a cover object.

Cover Object Selection Selection of a different set of cover objects e.g., using a seeded Pseudo-Random-Number-Generator (PRNG).

Furthermore, in process data transmission case study (CS_2) and in the corresponding publication ([LNK+21]), following triggers for such relocations are proposed:

recache-by-default constant change/hopping between cover/covert channel, embedding position or cover object selection driven by the embedding algorithm (e.g., as part of the stego key)

triggered-recache manually triggered change (for example by a control message from the secret sender, typically part of covert protocol (cf. [MWZ+16], Ch.4)

conditional-recache triggered when a pre-defined condition is met (for example entropy of cover objects gets below a certain threshold)

The cover object selection process has been studied in greater detail throughout the case studies, primarily to modulate the bandwidth. In the following, both cover object selection and bandwidth modulation is discussed.

5.1.6 Cover Object Selection

In the course of the case studies, methods of cover object selection have been investigated. In the Process Data Transmission Case Study CS_2, an entropy-based approach was used to estimate the noise in sensor data. This approach was to select the cover channel (i.e., the sensor with more noise) and to decide for each cover object on that channel if its used for embedding or not, i.e. it has to match certain quality criteria to be used for embedding, in this case entropy. The main purpose of this selection method is to select cover (objects) providing the most chances of staying undetected. This process can be considered to be a *dynamic* (algorithm-based) method for cover object selection as it decides for each cover object to be used or not.

In contrast, in the Modbus/TCP case study CS_1, a *static* packet filter as part of the stego key and paramater for the embedding process was used to define a subset of cover objects based on a *static rule*.

Additionally, a second parameter, the embedding ratio, was used to further select certain cover objects from this subset. Embedding ratio is a parameter, that defines the ratio of used and unused cover objects for embedding. In case of the Modbus/TCP Case Study, a Pseudo-Random-Number-Generator (PRNG) which is seeded with a secret derived from the stego-key used to pseudo-randomly decide whether a cover object is used for embedding or not. The usage of PRNGs is also a common technique in Image Steganography.

In the Process Data Transmission Case Study CS_2 this random selection method was used as well in the evaluation of multiple selection methods or strategies: In addition, to the pseudo-random selection, simpler rules were tested as well: One strategy was to use every second cover (i.e. embedding ratio = 0.5), the other tested strategy was to use every third cover (ratio = 0.33). The advantage of pseudo-random selection is that the embedding ratio can be arbitrarily set and is harder to reconstruct by a warden.

Cover Object Selection methods are also a way for (dynamic) modulation of the resulting bandwidth, which will be covered in the next section.

5.1.7 Bandwidth Modulation

The relationship between bandwidth, capacity, robustness, and undetectability is described by the *magic triangle* in Section 2.1.7. In the course of the case studies, one objective was to investigate methods for bandwidth modulation with the general idea of sacrificing bandwidth to achieve (more) undetectability (or theoretically robustness instead). In the case studies, the bandwidth modulation has been realized by the aforementioned methods of cover selection as well as the use of a dynamically adjusted capacity per cover object in case of the process data case study (CS_2).

In case of process data, it could be shown, that the bandwidth and capacity can be dynamically adapted to the current process and noise of the cover channel. This process allows to directly optimize the bandwidth against detectability. The detection results, indicate a linear impact of the bandwidth (and embedding ratio) on the

true positive detection rate, i.e. the lower the bandwidth the lower the detection probability.

For network covert channels, packet filtering, pseudo-random selection of cover objects and of course changes to the capacity per cover object could be shown to be effective and predictable parameters for influencing the bandwidth. However, in contrast to the process data case study, in the Modbus/TCP study a decreased bandwidth did not directly resulted in lower detection rates.

This is due to way the different detectors worked in the case studies. This discrepency is explained in the following.

5.1.8 Steganographic Cost & Detection

The idea of the prior discussed bandwidth modulation is to sacrifice bandwidth for undetectability. Behind this strategy stands the concept of steganographic cost, i.e. the idea that with increasing bandwidth or capacity, the cover object or carrier gets more and more degraded, making the hidden information easier or more likely to be detected. This concept derives mainly from the domain of media steganography. A common example would be the modulation of Least-Significant-Bits (LSB) of pixels. The more pixels are modified, the carrier (the image) is degraded in quality and by that the embedded information is easier to detect or might at least raise suspicion. However, depending on the covert channel and the detection method employed by the warden, this degradation might not even occur, or this degradation might not affect the detection rates. For example, in the Modbus/TCP Case Study, common IDS systems are used to detect covert storage channels. These IDS work on a packet-by-packet basis and are able to detect for each cover object (packet) if it contains hidden information or not. Therefore, in that case, the bandwidth has no significance on the detection rate, as the detector will inspect each single cover object independently. For covert storage channels this can also apply to capacity. For example, in case of CC_{MB4} (Padding) the capacity did not affect the detection rate, as the detector uses as a feature just the fact that a padding exists. This might also apply to other covert channels.

In contrast, in the Process Data Case Study, the proposed detection mechanisms do not inspect each value (cover object) individually to decide whether it contains hidden information or not. Instead, a (sliding) window is used to select a subset of the transmitted sensor values (process data). In that case study, the bandwidth had a linear impact on the detection rate.

This is an important insight, for both secret sender/receivers and wardens: For the secret sender the choice of the carrier and covert channel can influence if a modulation of capacity and bandwidth is reasonable. On the wardens side, it is important to have an understanding of the shortcomings of the chosen detection method.

5.1.9 Location and Persistence of Cover Objects

The contents of this section have been peer reviewed and published in part within the scope of the following publications as joint work with the co-authors Tom Neubert, Jonas Hielscher, Christian Krätzer and Jana Dittmann: [LNK+21, LNH+22]

5.1.9.1 Temporal Aspect and Persistence

One recurrent pattern in the case studies is the fact that cover objects in CPS environments may have different life times or levels of persistence depending on its current carrier and the fact that cover objects might be passed from one carrier to another carrier.

One example is the modulation of sensor values in Section 4.2. In this case study, sensor readings are modulated by a PLC and transferred to a historian.

The core idea of this exfiltration strategy is the fact, that in CPS it is common to store process data for longer periods of time in an historian database. This circumstance is leveraged to establish an asynchronous covert channel, where the hidden information is retrieved independently from the point of time when it got embedded. Therefore, such **asynchronous covert channels** are established if the *cover object* is *persistent*. Such persistence can be further differentiated.

Based on our initial proposals in [LNK+21] and [LNH+22], four **levels of persistence** are derived:

L0 no persistence (ephemeral, synchronous channel)

L1 **short-term persistence** (minutes to hours)

L2 **mid-term persistence**, e.g., log files (days to weeks) [LNH+22]

L3 **long-term persistence** e.g., historians (months to years) [LNK+21]

L2 mid-term and L3 long-term persistence is the foundation of *cache-based* (dead-drop like) hiding methods.

In contrast to that a **synchronous covert channel** makes use of an **ephemeral cover channel** (L0), e.g. direct TCP/IP communication between an HMI and PLC (under the premise, that communication does not get recorded), i.e. such are to be expected when using techniques of network steganography.

In case of the data exfiltration strategy of the case-study, the long-term persistent storage of process data acts as a cover channel.

5.1.9.2 Location of persistent Cover Objects in CPS environments

The process data case study (and the analysis of Syslog as a carrier in [LNH+22] have both) revealed that hidden information may be distributed to multiple locations. In CPS, data is often processed and transmitted at different stations. A common example for CPS are sensor readings: a sensor measures a physical property, which is then interpreted, encoded and sent to a PLC. The PLC receives, processes and

forwards the data to another PLC, an HMI display for human interaction and a data storage, such as a historian. The sensor readings are sent over the network, passing through network switches, protocol converters, firewalls, unidirectional gateways, etc. At each of these stations, the network packet is interpreted and processed. In terms of covert channels, each of these stations is a potential covert sender and/or receiver. This is especially important when considering the placement of wardens. Depending on where network traffic is collected, a warden may be unable to detect covert channels between certain points. Therefore, it is important to conduct a risk analysis to be aware of which channels are detectable and which are not. On the other hand, since the data is accessible from multiple redundant points, these redundancies can be used to identify modifications to transmitted data and, in turn, detect covert channels.

5.1.10 Derived Classification of (Stego-) Key Distribution Schemes

The contents of this section have been peer reviewed and published in part within the scope of the following publications as joint work with the co-authors Tom Neubert, Jonas Hielscher, Christian Krätzer and Jana Dittmann: [LD20, LNK+21, LNH+22, LD22].

As pointed out in the description of the Prisoners' Problem in Section 2.1.5 as well as in the Hidden Communications Model of Section 3.5 Covert Channels require a shared secret between covert sender and receiver (stego key) to drive the embedding and retrieval algorithms.

In practical scenarios and with regard to the investigated threat scenarios, the question arises how such key exchanges take place.

In the course of the case studies presented in Chapter 4, some methods were touched on. In the following, these key distribution schemes are categorized, and further methods are derived from additional publications not covered by the case studies.

Pre-Shared Key Scheme (PSK) Hardcoded pre-shared keys are a common assumption in both covert channel research and real-world applications (Stego-malware [BKD23]). In this case, both sender and receiver have hardcoded secrets that can be used as a Stego-Key or to derive key material for the covert channel and potential encryption, for example by using Key-Derivation functions on the hardcoded secret. The benefit of this is that it is easy to implement and does not require any additional communication. The downside is that the key is included in the source code, making it vulnerable to reverse engineering or disassembly by the warden. If the warden discovers the key, they may be able to inject or manipulate stego-messages (malicious warden). The Pre-Shared Key Scheme can be further differentiated into sub-forms:

PSK_{hard} A hardcoded pre-shared key driving the embedding and retrieval. Secret message is embedded the same way for each cover object.

PSK_{param} Instead of using a proper shared secret, the set of parameters contrucsts the key. Two sub-forms for the parametrization approach:

PSK_{static} the embedding alogrithm is paramterized once and after that not changed, therefore can be easier reverse-engineeringed in steganalysis

$PSK_{dynamic}$ uses a dynamic parametrization e.g., changes after a certain amount of time or is only used as initialization and then a new key exchanged over the established covert channel (cf. recaching methods in Section 4.2.4.2 [Process Data Case Study CS_2]).

Pre-Shared Algorithm for Key Derivation Scheme (KDF_{PSA}) In this scheme, both sender and receiver have a common algorithm to derive a shared secret key from a specific context or public information, e.g., by deriving key material from the current time. The advantage here is that the key is not static, i.e. embedding will be different, making detection more difficult. On the other hand, this type of keys can be reverse engineered.

Cyber-Physical Key Derivation Scheme (KDF_{CPS}) This scheme is a sub-form of the *Pre-Shared Algorithm for Key Derivation Scheme (KDF_{PSA})* and is specific to Cyber-Physical Systems. The idea behind this scheme is to use high-entropy data that both sender and receiver have (read-)access to, e.g. process data. From this data pool, data points can be selected to derive key material using key derivation functions. This scheme has been used in the case study on process data (sensor measurements) as cover [LNK+21].

Asymmetric Key Exchange Scheme (KE_{asym}) The asymmetric key exchange scheme leverages common asymmetric encryption to securely exchange key material. In this scheme, the sender (Alice) either has the public key ($K_{P_{Bob}}$) of the receiver (Bob) beforehand or is able to retrieve it from some public location. The sender then generates the Stego Key (K_{Steg}) and encrypts it using Bob's public key ($c = E(K_{Steg}, K_{P_{Bob}})$. This encrypted key can then be transmitted to the receiver (Bob) either using an overt communication channel or using a different (pre-shared key based) covert channel to increase stealthiness). The receiver can then decrypt the Stego Key using his secret key ($K_{Steg} = D(c, K_{S_{Bob}})$, that only he has access to. We used this scheme has in a publication on covert channels in Syslog [LNH+22]. The advantage of this scheme is the security of the stego key. The disadvantage is the additional communication. Additionally, asymmetric encryption requires more storage and computation power which might prevent its use in some use cases.

Covert Key Exchange Scheme(KE_{covert}) This scheme makes use of common key exchange protocols, for example the Diffie-Hellman(-Merkle) Key Exchange, in this case over a predetermined (hard-coded) covert channel. The security of this key exchange is limited to the detectability of the covert channel i.e., the assumption is that the key exchange is not detected/monitored by a warden.

Overt Key Exchange Scheme (KE_{overt}) This scheme uses innocuous traffic to exchange the stego key using a common key exchange protocol. For example, in the NTS Case Study ([LD22]) the TLS-Key-Export functions are used to derive a stego key from a TLS-Key-Exchange. Using encrypted traffic makes is harder for wardens to detect such key exchanges. In case of the NTS case

study, this key exchange is even undetectable for wardens monitoring network traffic.

Dead Drop Key Retrieval Scheme ($KE_{deaddrop}$) In this scheme the stego is fetched key from a dead drop, e.g., public reachable source. For example, in a publication on covert channels that leverage Syslog, we used a feature of Github to publicly store an public RSA key, which can be retrieved using one single https call to github.com [LNH+22]. The advantage is, that the stego key does not have to be delivered with the stego-malware. The disadvantage is, that the key needs to be retrieved. If this is not possible, the covert channel cannot be established (unless a hard-coded fallback is available). Another disadvantage is that this key retrieval can be detected by a warden.

Out-of-Band Key Exchange Scheme (KE_{OOB}) This concept is based on the use of an Out-of-Band Covert Channel, such as audio, to exchange the stego key in either a unidirectional or bidirectional manner. This idea was inspired by the use of audio-signals to provide WiFi-Passwords from a user's smartphone to an IoT device (e.g., vacuum cleaning bots or security cameras). This exchange scheme can provide a stealthy way for key exchange, but it is only applicable in certain scenarios where the sender and receiver are close and have the capability to transmit/receive such OOB signals.

5.1.11 Tactical Deception

In the process data case study CS_2 and our publication on covert channels in Syslog embedded by port scans [LNH+22], we proposed the use of tactical deception to further conceal the covert channel. This involves encoding hidden information in the order of ports that are scanned within a port scan against a firewall, which then logs these security events and creates and communicates corresponding syslogs. Another real-world example of such deception tactics was an attack on an Indian nuclear plant in 2019 [BeI19], where the adversaries covered their actual attack by using an off-the-shelf crypto miner. The defenders thought they had mitigated the attack when they removed the crypto miner, but the adversaries were actually still deep into their networks, remaining undetected for a long time. This tactic is especially useful for covert channels, as innocuous-looking attacks may contain hidden communication. Another option, proposed in the corresponding publication of Case Study CS_2 ([LNK+21]), is to operate multiple covert channels in parallel. For example, one idea is to operate one covert channel that is relatively easy to detect, but does not contain any relevant (hidden) information, so that a detection does not harm the adversary, while at the same time operating a more stealthy covert channel, containing the actual secret communication, with the hope of distracting the warden with the easier-to-detect covert channel. This follows the general idea of "making noise to be able to whisper".

5.1.12 Proposal for an Extension of MITRE ATT&CK® for *Industrial Control Systems*

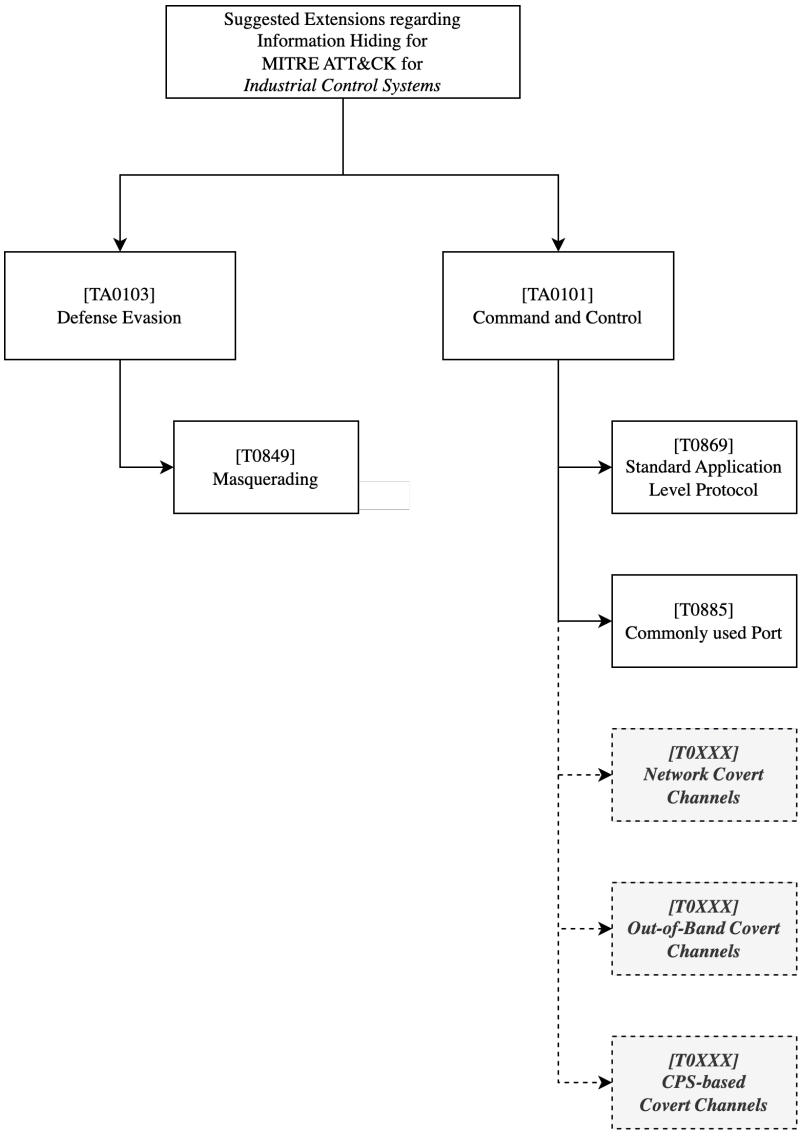


Figure 5.1: Proposal for an Extension of MITRE ATT&CK® for *Industrial Control Systems* with the inclusion of Network Covert Channels, CPS-based Covert Channels and Out-of-Band Covert Channels.

As pointed out in Section 2.2.5.2, MITRE ATT&CK® for *Industrial Control Systems* lacks a detailed inclusion of Information-based techniques - especially in the context of (covert) Command-and-Control `[TA0101]`.

Based on the insights from the derived classification of Section 3.3 and the case studies of Chapter 4, an extension of the framework seems recommendable to be able to cover and describe better the threat imposed by such IH-based methods.

Figure 5.1 shows the proposed extension with the integration of Network Covert Channels, CPS-based Covert Channels and Out-of-Band Covert Channels as novel techniques for `[TA0101]` *Command and Control*. In this proposal these are separate

techniques to highlight their importance and need for further research. An alternative approach would be to include these as sub-techniques for the already included techniques `[T0869]` *Standard Application Level Protocol* and `[T0885]` *Commonly used Port* (cf. Table 2.2 and Figure 2.8).

5.2 Countering Covert Channels in Industrial Control Systems

This section summarizes the insights from Chapter 3 and the Case Studies of Chapter 4 in terms of mitigation of covert channels in ICS.

5.2.1 Proposal for an Extended Mitigation/Countermeasure Model

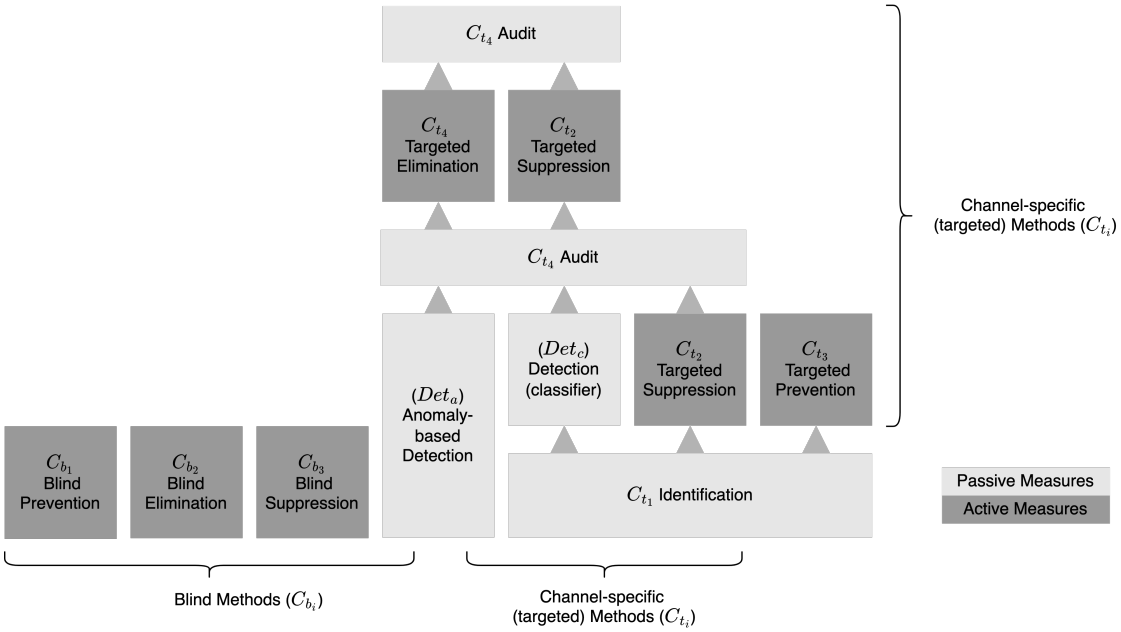


Figure 5.2: Extended Countermeasure Model based on [MWZ+16].

As described in the State of Art in Section 2.1.10 and shown in Figure 2.4, Mazurczyk et al. differentiate between the following sets of countermeasures against network steganography [MWZ+16]:

Identification formal or adhoc methods for discovery of covert channels

Prevention Prevention of covert channels in protocol/system design or implementation e.g., by using a different protocol

Elimination traffic or protocol normalization

Capacity Limitation limiting the capacity e.g., by slowing down system- or protocol mechanisms; introduction of noise

Detection using trained classifier for identified covert channels or anomaly detection

Auditing when a covert channel can be detected yet not be eliminated or its capacity be limited its use should be monitored (in case of identification of covert sender and/or receiver actions might taken against them directly)

Based on the insights from the case studies, in this work, an extension to this model is proposed for a more targeted description of countermeasures in the context of Cyber-Physical Systems. Figure 5.2 illustrates these extensions in detail.

This extended model incorporates three blind methods (in Figure 5.2 enumerated and labeled as C_{b_i}), two detection approaches ($Det_{a,c}$), and four Channel-specific (targeted) methods (C_{t_i}).

This distinction of *blind* and *targeted* methods is one of the major significant changes to the model of [MWZ+16]. *Blind* methods work against covert channels without any knowledge of the existence or presence of specific covert channels. These methods are:

C_{b_1} ***Blind* Prevention**: General design and implementation based methods to prevent a broad range of covert channels, e.g., use of zero-trust architectures, or use of protocols known to be robust against covert channels.

C_{b_2} ***Blind* Elimination**: Elimination of covert channels by active, untargeted methods which modify the overt channel, e.g., protocol conversions.

C_{b_3} ***Blind* Suppression**: Limitation of the bandwidth or robustness of covert channels, e.g., by introduction of noise.

In [MWZ+16] *Prevention, Elimination, Capacity Limitation* as well as *Detection* using trained classifier, require prior *Identification* (see Figure 2.4 for reference). These methods which counter known, i.e., prior *identified* covert channels, are here, in this extended model, named *targeted* methods.

As shown in Figure 5.2 the first step of this *targeted* approach is

C_{t_1} **Identification**: formal or adhoc methods for discovery of covert channels.

Once a covert channel is identified, it can be detected using a trained classifier (Det_c), suppressed (C_{t_2}) or prevented (C_{t_3}):

Det_c **Detection** (*trained classifier*): Covert Channel specific detection methods using trained classifier. These can be machine-learning based approaches or static rule sets.

C_{t_2} ***Targeted* Suppression**: Targeted limitation of bandwidth, capacity or robustness of a specific covert channels using targeted actions, e.g., the introduction of noise.

C_{t_3} ***Targeted* Prevention**: Targeted actions that prevent the establishment of a specific covert channel, e.g., by architectural changes, use of protocol converters, et cetera.

As shown in Figure 5.2, a follow up step to a covert channel that is detected by Det_a *Anomaly Detection* or an trained classifier Det_c, is the ***Audit*** C_{t_4}. The aim of C_{t_4} *Audit* is to provide details on the use of covert channels and the potential *covert sender* and *covert receiver*. This allows to take targeted actions in a next step (***Targeted Elimination*** C_{t_5} or ***Targeted Suppression*** C_{t_2}) against the covert channel or -sender/receiver. After these actions, it is important to run the audit again to measure the success of elimination or suppression.

Furthermore, this model differentiates in the same way as [MWZ+16] between *active* and *passive* measures. ***Active measures*** modify overt or covert objects or both, whereas ***passive measures*** only monitor the cover objects (e.g., network traffic). Therefore, this distinction, basically follows the definition of *active* and *passive* Wardens (cf. Section 2.1.5).

In this extended model, all three blind methods ($C_{b_{1,2,3}}$) as well as C_{t_2} *targeted Suppression*, C_{t_3} *targeted Prevention* and C_{t_2} *targeted Elimination* are **active** measures.

Both detection approaches ($Det_{a,c}$), C_{t_1} *Identification* and C_{t_4} *Audit* are **passive** measures and do **not** modify cover objects.

5.2.2 Considerations on Methods of Detection

Machine Learning based approaches make use of common metrics or features that are established in covert channel detection research. Although several machine learning approaches and features have been investigated in the past (e.g., c.f. [EG22] for an extensive survey) and provide promising results regarding detection rates (accuracy and recall) they come with some drawbacks or limitations, respectively. For example, most approaches are 2-class classifier, i.e., they require proper datasets and therefore can only detect (a-priori) **known** covert channels. Most approaches are specific to one covert channel or a specific set of covert channels, e.g., header fields of one protocol, therefore, several detectors have to be run in parallel, which can cause significant overhead. Approaches that work against a broader set of covert channels tend to be either inefficient or also cause considerable overhead [EG22]. Another important factor are properly created datasets. The creation of proper datasets is a non-trivial and time consuming task and has many potential error sources. Another problem in Machine Learning, and also affecting the detection part in the use cases is the tendencies of some algorithms to overfit and the lack of deeper investations in terms of generalization. Currently, existing datasets suffer from many problems such as oudated content and uneveness, or they are not publicly disclosed or specific to the environment and therefore not generizable [EG22]. Another challenge is the focus in common TCP/IP protocols (especially strong focus on IPv4, IPv6 and DNS). It remains unclear whether and how these detectors can be used on OT application layer protocols. Another issue is that adversaries that have access to the features and metrics might assimilate the statistical properties of the overt traffic. In [EG22] the authors also state that many recent approaches have *high False Positive (FP)* rates, which for ICS would render them useless in productive systems. A general observation is also that published approaches (including our own) focus only on the detection and are not designed to be integrated in a security management system (isolated solutions).

In the Process Data Case Study CS_2, the investigated machine-learning detection approach is able to detect the proposed covert channel with a high accuracy with low false-positives in high bandwidth scenarios. However, with a decrease in capacity/bandwidth, the detection rate drops significantly. With further research, these results might be improved on, though. Still, from a practical standpoint, due to significant overhead, the use of existing architecture and procedures (e.g., IDS/SIEMs etc.), that might also be already in use in the IT parts of the systems seem preferable to custom tailored ML-based detection schemes in productive environments. However, with ongoing and future research, advances in (ML-based) anomaly detection might be efficient and effective to be an additional or even preferable solution.

The detection of covert channels using existing tools for intrusion detection and prevention (IDS/IPS) is the topic of promising past and ongoing research (e.g., cf. [KWJ21, Cav21, Zan17, Gun17, Wen12b]). The Modbus/TCP Case Study CS_1 could also illustrate its effectiveness against known covert channels as targeted measure for detection and elimination. However, such IDS systems also have significant limitations. Most significantly, most systems work signature-based, i.e. the detection is based on certain known indicators (like Indicators-of-Compromise and malware signatures). This also means that they are typically used to detect known threats, or in this case, known covert channels. Furthermore, most systems work statically and on a packet-per-packet basis, limiting the possibility to detect covert channels distributed across multiple packets or in timing features (i.e., there is generally a limitation to the detection of storage channels). Another problem is the current lack of support for OT-specific protocols. Although there are first advances in this area[1], many protocols are yet to be supported. Other advancements are, in contrast to signature-based systems, Intrusion-Detection-Systems that support scripting. One example is Zeek, which also has been successfully used in the Modbus Case Study for detection of certain covert channels which otherwise could not be detected. With such powerful tools, further advancements in detection and migitation are to be expected, as they reduce limitations of previous systems.

In general, a combination of a covert channel limitating, security-focussed architecture together with anomaly-based detection of hidden threats as well as the use of covert channel-aware intrusion detection and prevention systems seem advisable. Another way such IPS systems can be used is to enforce protocol-normalization, eliminating blindly or preventing blindly (yet) unknown or undetected covert channels.

[1] e.g., the ICSNPP protocol parser of CISA (https://github.com/cisagov/ICSNPP)

5.2.3 Derived Design Principles and Measures for Covert-Channel-robust Architectures

In the course of the case studies and further publications, several system design principles and measures could be identified to improve the robustness against covert channels and can act as *blind* mitigation methods for prevention, elimination and suppression:

Decoupling and Segmentation As covert channels require cover communication, one of the most effective measures is to establish a zero-trust architecture with highly-segmented security zones and -levels and also decoupling to break implicit trust in data transmissions. One example approach is the Defense-in-Depth architecture proposed in the IAEA NSS 17-T [IAE21].

Cross-Validation of Data When redundant data access points or data sources are available, these can be used to validate against each other to find differences or anomalies, providing indicators for covert channels. This has been shown effective in the process data case study and the time synchronization case study.

Digital Twins Digital twins can be a method for attaining cross-validation. They can be used to create a ground truth or baseline to detect anomalies in the physical system.

Reduction of Complexity The complexity and corresponding entropy of a system has shown to be the most significant enabler for the establishment of covert channels. To reduce entropy and complexity of the system, all components must be taken into account, including the architecture, systems, and communication. It has been demonstrated that as network protocols become more complex, the risk of covert channels increases (e.g. OPC UA and Modbus/TCP or NTS and NTP). Simpler systems and protocols are easier to inspect, normalize, and monitor for covert channels than complex protocols. Therefore, it is recommended to use less complex communication protocols, and depending on the risk assessment, unencrypted protocols may be used if encryption is not essentially required. This should be taken into account in the risk assessment.

Unidirectional Communication In order to disrupt the implicit trust between agents, unidirectional gateways are a straightforward yet highly effective way to stop a great number of covert channels. This is mainly due to the fact that these gateways are essentially protocol-converters and limit communication to the absolut essential communication (by that eliminating any unneccesary complexity with could be used by covert channels). By using a unidirectional gateway, the system operator must decide which information is essential and who needs it, while at the same time limiting the amount of data sent back to the sender.

Essential Communication Communication between devices should be limited to what is essential to operate the system. Preliminary experiments with PLCs, Engineering Workstations and HMIs of different vendors have shown that many

(comfort-) features are enabled by default, resulting in unncessary network traffic, e.g., the Link Layer Discovery Protocol (LLDP) is used by many vendors.

Essential Data Data should be restricted to only what is essential to operate the system. This includes the kind of data, the quantity of data, and the quality of data. Common examples are the precision of process data (see the process data case study) or the verbosity of log files (see our syslog study [LNH+22]).

Monitored Zone-Transitions The vertical and lateral segmentation of security levels and zones is a critical step in creating a secure architecture that is resistant to covert channels. To be effective, all communication passing through these boundaries (transitions) must be closely monitored using firewalls, IDS/IPS systems, and wardens. Only authorized traffic should be allowed to cross levels or zones (whitelisting).

Visibility It is essential to have visibility into the production network in order to accurately identify potential threats. This requires the use of network sensors, taps, span ports, and other tools to actively monitor network traffic in all segmentations. Additionally, it is important to actively maintain a list of all hardware and software components, including firmware-versions, patch levels, and other related information.

Baselining and Anomaly Detection It has been demonstrated that a combination of traditional intrusion detection/prevention systems and anomaly detection can form a strong basis for the detection of covert channels. For anomaly detection to be effective, it is essential to have an appropriate baseline against which anomalies can be identified. However, creating such a baseline in operational networks presents many challenges. The operator must ensure that the system is uncompromised and that the baseline recording is representative of the system. Edge cases and special situations such as maintenance or the addition of new components must also be taken into account.

Normalization In the case studies, protocol normalization has been demonstrated to be an effective way to combat covert storage channels. To reduce the majority of storage-based covert channels, protocol normalization should be implemented between each security level and -zone. However, similar to baselining and anomaly detection, the learning phase is of special criticality, as disscused by S. Wendzel in [Wen12b].

One example for the application of these principles is the proposed architecture in the time synchronization case study CS_3 (cf. Section 4.3.6.3.)

5.2.4 Derived Requirements for effective Wardens

From the investigations of this work, some requirements for wardens can be derived for wardens to be able to effectively detect and counter covert channels in Industrial Control Systems. In the following, some points that came across in the case studies are touched upon:

Positioning and Data Access The position of the warden in the communication is critical. As shown in the the case studies, it is recommended to implement a defense-in-depth security architecture that consists of security levels and -zones with clearly defined transitions. For each transition must be defined which communication is explicitly allowed, which should be enforced and monitored with IDS/IPS systems, possibly in conjunction with a SIEM and a IH-warden specific to the security level/zone transition. Another factor is data access, i.e. the ability of the warden to access all data required for detection. This might also include data access to historian databases, log servers and other redundant or external data sources to be able to find deviations and anomalies.

Protocol Inspection Another important factor is the ability to fully inspect protocols on all OSI-layers, including the application layer. This is a very important factor, that is often lacked upon in OT environments as common firewalls and IDS/IPS systems might not support all relevant OT protocols.

Cryptographic Features The growing utilization of encryption in network protocols, such as the security features of OPC UA or as demonstrated in Case Study CS_3 for NTS, requires that wardens should be able to intercept encrypted communication for protocol inspections. However, there are exceptions, where it is possible to detect covert channels even in the encrypted domain, for instance in the case of many timing channels. Additionally, some covert channels may exhibit patterns that can be detected even in the encrypted domain. Nevertheless, as demonstrated in the NTS case study, some covert channels are only detectable, when the warden is able to intercept, interpret, and inspect the communication.

Event Aggregation Another approach for detection is to find discrepencies in the correlation of datapoints (in process data or log data) and (cyber-physical or IT-related) events. The idea is to aggreate events and data from different locations to detect inconsistencies, e.g. the rise of water level even though the corresponding valve is closed or user interactions from staff who are actually not at work (cf. our paper [AZL+21]).

Warden Security (or who controlls the Warden?) Each introduction of additional agents like firewalls, IDS/IPS, wardens etc. also open the doors for new threats, i.e. the "security" devices/agents have to be checked and monitored as well, as they might bring in (supply-chain) threats or novel ways for hidden communication (e.g., c.f. [MM20]).

Considerations on Performance, Accuracy, and Reliabilty One of the most important aspects when bringing new security measures into a system is the impact

on performance. As in general Cyber Physical Systems work as real-time systems, it is of special importance that security measures do not impair the performance of the system. Therefore, each detection or mitigation method has to be tested, whether it has an (significant) impact on the performance on the system. Another consideration to make is the accuarcy and reliability of detection methods. For a productive environment, the general aim is to achieve a very low false-positive rate (ideally zero), i.e. to have as few false alarms as possible. As ICS require high availabilty (especially in case of critical infrastructure), false alarms can impair availability as they typically require further investigations, which might impact the whole system, e.g., when certain components have to be taken offline for investigation.

5.3 Summary & Addressed Research Questions

This section provides an overview of the specific contributions of this thesis to address each research question.

Q_1 *How applicable are Information Hiding-based principles and techniques to CPS environments, i.e., what are plausible carriers and how do exemplary covert channels perform in terms of stealthiness and bandwidth?*

For answering this question, at first, a plausible reference architecture (PlauRA) for Industrial Control Systems with three sub-types was derived from the State-of-Art, which allowed for a systematic identification of potential covert sender and -receiver and carrier for hidden information. It could be shown, that the architecture of ICS allow for a variety of potential secret sender and receiver as well as a variety of options for potential carrier. The reference architecture also aids in the illustration of adversarial use of covert channels in threat scenarios for Industrial Control Systems.

In a second step, and based on selected publications of the State-of-Art, a classification for methods of Information Hiding and Steganography *specific to Cyber-Physical Systems* is derived, allowing for a systematic description of covert channels in CPS and providing a better understanding on the variety and spectrum of the application of Information Hiding methods to the specifics of Cyber-Physical Systems. This classification specifically highlights that the complexity of such systems allows for a broad range of steganographic methods and domains to be applied to the variety of carriers found in such systems.

At the example of three selected case studies, three distinctive carriers of ICS were investigated in terms of potential covert channels, their performance, threat scenarios and mitigation. In the course of the case studies, in total 29 covert channels across four carriers in ICS are discussed. From these 29 covert channels, six covert channels were investigated in depth, evaluating and comparing them in terms of capacity, bandwidth, stealthiness, plausibility and protocol- as well as warden compliance.

In the course of Modbus/TCP case study CS_1, it was demonstrated that the application of Information Hiding patterns (in this case the Network Information Hiding Taxonomy by Wendzel et al. [MWC18]) to the specification of network protocols can help in the systematic identification of potential covert channels. The practical results demonstrate that some of the potential covert channels may not be feasible or are heavily reliant on the target environment, which includes different devices, firmwares, hardware, and implementation variations. The specification leaves room for interpretation and implementation, so software from different vendors can have considerable differences in their behavior, which can have a direct effect on the use of covert channels or even enable covert channels that are not apparent from the specification. These differences between environments can lead to lower bandwidths, higher detectability, less robustness, or even the impossibility of using certain covert channels. For instance, in one environment a covert channel may be statistically undetectable, while in another environment the same covert channel may be so conspicuous that it can be easily spotted by simply inspecting

the network traffic. This is an important insight as it emphasizes the need for the sender and receiver to be aware of the specific cover characteristics of the environment they are using. This means that prior reconnaissance is essential. This is a significant factor for adversarial behavior modeling, as well as detection and mitigation strategies, as it requires the adversary to analyze the cover, which may result in them being detected even before covert channels are established. The same is true for the warden, who needs to have specific knowledge of the cover characteristics for successful detection and mitigation.

The case study CS_2 demonstrated at the example of noise in sensor measurements and process data in general, that process data is not only a distinct cover for CPS but also a likely one to be used for the establishment of adversarial covert communication channels. In contrast to the other investigated network covert channels, in which the cover object (i.e., a network packet) is sent from a sender to a receiver, such process data is processed multiple times at different *locations* and points in *time* in a CPS. This temporal (in terms of persistence) and locational aspect of cover objects (cf. Section 5.1.9) is an important insight, as it enables asynchronous covert communication with multiple combinations of potential positions for embedding and retrieval. This feature makes detection and mitigation especially difficult.

In the course of the case studies, methods for cover selection and bandwidth modulation have been successfully investigated. In case study CS_2, a dynamic method for cover selection based on suitability (in this case noise measured by entropy) is used. Based on the noise within the data, the capacity per cover object can be dynamically adjusted to keep the modifications within the bounds of usual noise. All three case studies use methods for modulating the bandwidth and using a pseudo-random distribution of the hidden information across several cover objects. This approach was shown to be effective against the proposed machine learning-based detection approaches proposed in Case Study CS_2.

Case Study CS_3 showed the potential for secret information to be transmitted through seemingly harmless infrastructure protocols such as NTP, PTP and NTS. Time synchronization is a complex process that necessitates the transmission of high-entropy data, often with more accuracy than the hardware is able to provide, leaving enough space to create covert channels in timestamp fields. Timestamps have also been proven to be a plausible carrier in supporting publications on covert channels in the OPC UA protocol (c.f. [HLD+20]). The hierarchical structure and implicit trust between time servers also adds to the risk of covert channels, as they can be used as proxies to traverse networks that are separated by firewalls. Moreover, such infrastructure protocols tend not to be monitored closely by IDS/IPS systems. This emphasizes the importance of paying attention not only to obvious carriers such as process communication and automation protocols, but also to seemingly harmless infrastructure communication.

From the case studies, supporting publications and the State-of-the-Art, a categorization for stego-key distribution schemes is derived in Section 5.1.10,

highlighting the variety of potential methods in the context of Cyber-Physical Systems.

In summary, the results of these investigations illustrate the variety of applicable covert channels to the specifics of Industrial Control Systems that might be leveraged by adversaries to cover secret Command-and-Control channels. It could be shown, that the selected, exemplary covert channels can be successfully applied in ICS, and that stealthy, robust covert channels can be established with sufficient capacities and bandwidth that would allow adversaries to secretly communicate in such systems.

Q_2 *What are plausible threat scenarios for adversarial use of covert channels in ICS?*

Each of the three case studies covered in this thesis discusses potential threat scenarios using the covert channels investigated in the case studies. These threat scenarios illustrate the adversarial use of covert channels for lateral and vertical communication in the context of the prior described plausible reference architecture for Industrial Control Systems (PlauRA). At the example of Modbus/TCP, representing common OT protocols, it could be shown that such OT protocols can be used for covert lateral communication between compromised devices of different security zones as well as vertical covert communication between devices of different security levels. However, since the use of OT protocols is mainly limited to Levels 1-3 of the PERA reference architecture (cf. Section 2.2.3), it can be expected that such covert channels occur only in the OT domain of ICS. In the second case study CS_2, it could be demonstrated, how covert channels can be established within the transfer and storage of process data. As such data is processed and transferred to multiple locations in such ICS, a variety of options for covert embedding and retrieval are available. The third case study CS_3 highlights the threat imposed by network infrastructure protocols, with the example of the time synchronization protocols NTP, PTP and NTS. It could be shown that implicit trust relationships between servers and devices can be leveraged to establish covert channels penetrating deep into the network architecture. Especially the transfer of high-entropy in data fields, in these cases timestamps, enabled covert channels that are hard to detect. At the example of NTS, it was demonstrated that encrypted protocols can actually be counterproductive (in terms of protection against IH-based threats), as they might create the possibility of covert channels that were not possible before. Moreover, the investigated covert channels are hard to detect and require capabilities for intercepting and decrypting the communication to be able to detect these channels, therefore requiring the deployment of active wardens. As a result, when introducing cryptographic features to a communication relationship, it is important to perform proper risk analysis and to have the potential threat of hidden communication in mind, when securing such communications. In summary, the threat scenarios in the case studies illustrate the imminent threat of adversarial use of covert channels in ICS to secretly communicate within such CPS and to create stealthy, hard-to-detect command-and-control channels with potential multiple pivot points, allowing for hidden communication across multiple levels and zones.

Q_3 What challenges can be encountered for detection and mitigation in Industrial Control Systems?

In Section 5.2 an extended countermeasure model is proposed, differentiating between *blind* methods of mitigation as well as channel-specific *targeted* methods. This extended model illustrates the variety of countermeasures available. In the course of the case studies, the focus was set to targeted methods against identified covert channels. For all three case studies, the identification of covert channels has been performed, which is the first step of targeted methods. In the Modbus/TCP case study (CS_1), the open source tool Zeek was chosen, to evaluate the performance of custom-tailored script-based detection methods. It could be shown, that three out of four covert channels could be successfully detected. One covert channel, leveraging the Transaction ID field of Modbus packets, requires further methods for detection, e.g., the use of anomaly-based detection schemes or Machine Learning-based detection with specificially crafted features for this channel. However, in theory, the investigated covert channel can be mitigated using protocol normalization as a method of *targeted prevention*.

In Case Study CS_2, the two-class ML-based detection approach was shown to be effective against the investigated covert channel when using high embedding ratios (and therefore high bandwidths). In low-bandwidth scenarios, though, many cover objects with hidden information remained undetected. It remains to be further investigated how a generalized approach could work against a broader set of covert channels leveraging process data as cover. However, by reducing the accuracy/resolution of the values, most covert channels can be limited in their capacity/bandwidth or completely eliminated. The analysis and calculation of such reductions in precision of sensor data also represents a challenge for productive environments. Active wardens are also an option in this case, e.g., one idea idea is to randomize the least significant bits in measurements to overwrite any embedded hidden bits (*blind suppression*).

Research in case study CS_3 has demonstrated that it can be challenging or require a lot of work to identify hidden channels in time synchronization. Normalizing the NTP protocol can be a useful starting point, but it is not enough to protect against more intricate channels in the timestamps. In Section 5.2.3 an architecture is proposed, that aims at limiting the potential for such covert channels. This architecture involves decoupling, breaking implicit trust between servers, using different protocols for different purposes (diversity) and providing redundancy. This redundancy can then be used to detect irregularities. For NTS, the cryptographic features require more advanced inspection techniques and the use of active wardens to reduce the number of identified covert channels. In summary, the results of CS_3 illustrates the amount of efforts that have to be taken for successfully mitigating the threat of covert channel in time synchronization protocols.

While these methods of detection and mitigation for the selected covert channels exist, they come with some challenges for productive enviroments: It has to be ensured that the use of such active measures, such as protocol normalizer, do not affect the functionality (i.e., availability and integrity) of the system.

Another open question is how to react, when a detector raises alarm. Therefore, response planning is *essential*. How such a response is designed, depends on the environment. However, in general and as already mentioned, availability is one of the biggest concerns in Cyber-Physical Systems, and especially in ICS. Therefore, for all detection methods it is important to have as few false-postivies as possible and for mitigation to have defined methods with enough redundancy to investigate alarms without affecting the functionality of this system. This is one of the main identified challenges for mitigation in CPS. In summary, it could be shown that there is not one method that works against all methods, instead a variety of countermeasures is required to counter the threat of covert channels in such systems.

6. Selected remaining Future Work

The research conducted for thesis represents a first foray into the novel field of Information Hiding and Covert Channels in Cyber-Physical Systems and can only cover certain aspects. Therefore, this work motivates further investigations in this domain. In the following, selected aspects are described.

Further Investigations of Carrier in ICS This research only examines a small portion of all potential covert channels that can be found in Industrial Control Systems. As covert channel research in CPS is still in its infancies, for each class of the proposed classification of covert channels in CPS in Section 3.3, further research is required to get a full understanding of the threat imposed by covert channels in CPS. For instance, OPC UA is an open and vendor-independent automation protocol that has been gaining popularity in recent years. In [HLD+20], we introduced a covert channel in timestamps for OPC UA, and in the context of a Bachelor thesis [Han23] more covert channels were evaluated. Despite these efforts, this research covers only a fraction of OPC UA, as it is a complex protocol with many sub-protocols and different security configurations. Furthermore, preliminary results show that the difference between *specification* and *implementation* might be signifcant, requiring further research to cover the potential of covert channels in OPC UA and how to effectively mitigate them. Another major research focus is on CPS-specific covert channels, which take advantage of process control. For example, the works of Herzberg et al. [HK19a, HK19b] demonstrate the potential of covert channels in the behavior of controlled processes (representing *Behaviour- and Property based Covert Channels* in the classification proposed in Section 3.3). Another large research area are *Out-of-Band* covert channels which have not been covered in this work (see Section 2.1.6). With the analysis of time synchronization protocols and logging mechanisms like syslog in [LNH+22], the threat imposed by innoucous looking infrastructure protocols has been demonstrated. It can be expected that other infrastructure might be susceptible to covert channels in similar fashion. For example, preliminary results of student projects indicated that Layer 2 protocols of the OSI model, like ARP, DHCP,

LLDP and others might also be used for covert channels *within* a security zone or -level. As such protocols are typically not inspected by IDS systems, these might provide a stealthy way of communication. As the analysis of NTS demonstrated, the introduction of cryptographic features might open the door for more covert channels. Therefore, encrypted protocols are a potential topic for future investigations as well.

Further Investigations of Covert Timing- and Event-based Channels While not being the focus in this thesis, the investigations show a tendency for a high potential of adversarial use of covert timing channels not only in network protocols but also on the application layer of CPS in the form of event-based channels (named B_{2_2} *Event Timing* in the classification, cf. Section 3.4.2.2). Another example for these event-based channels are our Syslog covert channels published in [LNH+22]. Here, the idea is to trigger the creation of logfiles by certain actions, in this case port scans against a firewall. The preliminary results are promising, yet require further investigations.

Continuous Improving of Classification for IH-Methods in CPS The classification of methods of Information Hiding in Cyber-Physicals Systems, described in Section 3.3 is a first proposal for such a classification and comes with some limitations and potential inconsistencies, as discussed in Section 3.4.4. Classification provides a strong basis for further exploration and enhancement. Ongoing research and real-world events can also help to continually improve it over time.

Further investigations on Stego-Key Exchanges In Section 5.1.10, a first classification for stego key distribution schemes in CPS is proposed. As considerations on the stego key were not the focus of this thesis, the classification represents a first step towards better understanding of the variety of potential methods for the distribution of stego keys and motivates further investigations, especially in terms of their practicality, performance, and robustness.

Evaluation of Full-scale Threat Scenarios Threat scenarios were discussed at the example of selected case studies. Future work should discuss full-scale threat scenarios from initial compromise in the IT domain and the following lateral movement from IT to OT with the use of multiple covert methods in combination, their performance, robustness, and mitigation.

Detection The covert channels discussed in this work illustrate the imposed threat and required methods for detection and mitigation. Further evaluations of Machine Learning-based techniques as well as the enhancement of existing security infrastructure for detection of covert channels seem advisable. Moreover, the combination of threat detection on the application layer and network-based detection and their integration into Event Aggregation-based systems, such as SIEMs are also a topic for future investigations. In this work, specialized, custom-tailored detectors were used. While these showed promising results, they were explicitly crafted for the known characteristics of the analyzed covert channels and (computationally) expensive. Therefore, generalized detectors for multiple carriers, capable of detecting a broad range of covert channels

should be investigated as well as their optimziation in terms of performance and accuracy.

Mitigation In this work, a range of potential methods are touched on. For the future, these should be investigated in more depth, especially in terms of their performance and also verified against a variety of Information Hiding-based techniques.

Risk Analysis Currently sufficient data on adversarial use of Information Hiding in CPS is missing, therefore, from a standpoint of risk analysis, it is difficult to calculate which countermeasures should be taken. In general, passive measures, such as changes to architecture and network segmentation, the use of protocol-conversions and data-diodes is generally less expensive than targted methods, for example, the proposed ML-based detection methods. Furthermore, such detection methods are not yet ready for commercial use.

Generalization and Specialization for certain CPS In this work, the focus was set to Industrial Control Systems, as a representative sub-form of Cyber-Physical Systems. In a next step, the findings of this work have to be verified to be generalizable for CPS and investigated how they translate to certain other forms of CPS and how the might be adapted to fit these systems. It is important to consider both generalization and specialization.

Refinements of Plausibility regarding Protocol- and Warden Compliance The concepts of plausibility in terms of procotol- and warden compliance provided a foundation for description and comparison of covert channels in CPS. However, in the analysis of network protocols in terms of potential covert channels, the estimation of plausibility and protcol- and warden compliance shows the tendency of not being exact and generalizable, instead being partly ambiguous and dependent on the actual environment and warden capabilities. Therefore, further investigations should aim at improving these concepts.

Integration in Attack Modelling As pointed out in Section 1.2 as well as in Section 2.2.5.2, covert channels in Industrial Control Systems are currently not reflected in the threat modeling of MITRE ATT&CK® for ICS. A first proposal for the integration was made in Section 5.1.12. In the future, it should be further investigated and discussed how carrier, assets and the correlating covert channels can be fully integrated into the model.

Integration as potential Threat during Protocol Specification The results of the case studies indicate, that there are covert channels which could have been prevented or mitigated in the design and specification phase of the protocol. Therefore, it seems advisable to include Information Hiding and covert channels as potential threats during design and specification. This topic is also highlighted in a recent study on IETF protocols by Caviglione and Mazurczyk [CM24].

Plausible Deniability & Attribution This work has discussed the potential threat of using Information Hiding techniques to establish covert communication channels, making it harder to detect adversarial activity in target environments

and allowing adversaries to remain undetected for longer periods of time. As security measures become more advanced, adversaries must find creative ways to bypass them in order to be successful. This has led to an increase in the use of Information Hiding-based techniques to conceal their actions, as seen in common IT-malware [CM22, BKD23]. However, one may question if this is necessary when other forms of attack, such as ransom-attacks or double/triple-extortion attacks, are so profitable. An important factor in the rise of Information Hiding is the concept of *plausible deniability*, which allows attackers to deny any involvement in an attack. This is especially important for nation-state (sponsored) attacks, as they can be politically motivated and require the attacker to deny any involvement. With increasing tensions in global politics and digital warfare, *plausible deniability* is becoming increasingly important. Techniques of Information Hiding make it more difficult to attribute attacks to certain threat actors, so it is likely that they will use these techniques to cover their tracks. It is essential for future research to develop methods and means for attributing the use of Information Hiding-based techniques to threat actors and to provide the necessary threat intelligence and countermeasures that will be needed in the future.

As this listing demonstrates, the work presented in this thesis motivates a broad range of future topics and further investigations for Information Hiding and Covert Channels in Cyber-Physicals Systems as well as the detection and mitgation of Information Hiding-based threats.

A. Appendix

A.1 Research Data Management

To promote transparency, reproducibility, and further investigation of the research results, the datasets and related code samples used in the case studies have been made available as an open-access repository and can be accessed at [Lam23][1]. The repository contains following data:

Modbus/TCP Due to licensing, only data that has been captured within the custom testbed is included. The sources of the other datasets that were used are listed below. Datasets that are included:

DS_1 **Modbus-node-red-1h** The modbus-node-red-1h.pcapng in the overt folder contains one hour of unmodified recorded traffic from the custom testbed. The description of the testbed can be found in Section 4.1 in Case Study CS_1. The pcap files in the covert folder contain network traffic with covert channels using different parameters as indicated in the filename. The description of the covert channels can also be found in the documentation of the case study.

Datasets not that are not directly included in the repository due to licensing:

DS_2 **Cyberville** Network capture from the 2020 SANS ICS Virtual Conference Capture-the-Flag (CTF) [Rob20]. Contains Modbus/TCP as well as Siemens S7Comm.

DS_3 **DEFCON23** Packet captures from the ICS village at the DEFCON 23 conference, 2020 [DEF20]. Contains mainly Profinet and some samples of Modbus/TCP.

DS_4 **CRITIS18-1v2** Clean network capture (`eth2dump-clean-1h_1.pcap` from the 1v2-capture) by Frazão et al. [Cru18], presented at CRITIS 2018

[1] https://doi.org/10.24352/ub.ovgu-2023-112

[FAC+19]. Contains one hour of clean Modbus/TCP traffic (i.e., capture does not contain any conducted attacks).

DS_5 **Lemay-run8** Modbus network captures by A. Lemay [LFK16]. `run8.pcap` used for investigation; contains one hour of non-modified Modbus/TCP traffic [Lem16].

DS_6 **Chinese CTF** contains Modbus/TCP and Siemens S7Comm traffic from a Chinese Capture-the-Flag [New18].

NTS As part of Case Study CS_3, the code and resulting datasets implementing the NTS-UID covert channel (cf. Section 4.3.5.3) are included. This includes 10.000 unmodified, as well as 10.000 UIDs with embedded messages and corresponding code.

Process Data Details on the covert channel can be found in [LNK+21] and in Case Study CS_2 in Section 4.2. The jupyter notebook for generation and analysis can be found in the corresponding code folder. Due to licensing, the repository does not contain parts of the original dataset that was used for embedding. The original data source used for embedding is the A1 dataset of the iTrust Secure Water Treatment (SWaT) dataset [GAJM17][2].

[2] https://itrust.sutd.edu.sg/itrust-labs_datasets/dataset_info/

Bibliography

[ABP+19] Cristina Alcaraz, Giuseppe Bernieri, Federica Pascucci, Javier Lopez, and Roberto Setola. Covert Channels-Based Stealth Attacks in Industry 4.0. *IEEE Systems Journal*, 13(4):3980–3988, December 2019. Conference Name: IEEE Systems Journal. doi:10.1109/JSYST.2019.2912308. (cited on Page 3, 4, 20, 25, 45, 47, 50, and 54)

[ABS21] Otis Alexander, Misha Belisle, and Jacob Steele. MITRE ATTACK for Industrial Control Systems: Design and Philosophy, 2021. URL: https://collaborate.mitre.org/attackics/index.php/Main_Page. (cited on Page 2, 6, 7, 33, 38, and 62)

[AL15] Michael J. Assante and Robert M. Lee. The ICS Cyber Kill Chain | SANS Institute, May 2015. URL: https://www.sans.org/white-papers/36297/. (cited on Page 6, 33, and 38)

[Ala10] Mohammed M Alani. Testing randomness in ciphertext of block-ciphers using diehard tests. *Int. J. Comput. Sci. Netw. Secur*, 10(4):53–57, 2010. (cited on Page 142)

[ALD23] Robert Altschaffel, Kevin Lamshöft, and Jana Dittmann. Verbundprojekt: Evaluierung von verfahren zum testen der informationssicherheit in der nuklearen leittechnik durch smarte testfallgenerierung 2 – teilprojekt: Hierarchisches smart-testing mit basis-angriffen, 2023. Suchbegriff: SMARTEST-2 - 1501600B. URL: https://grs-fbw.de. (cited on Page iv and vi)

[Alt20] Robert Altschaffel. Computer forensics in cyber-physical systems : applying existing forensic knowledge and procedures from classical IT to automation and automotive. 2020. Accepted: 2021-01-15T10:30:19Z ISBN: 9781744743729. URL: https://opendata.uni-halle.de//handle/1981185920/35574, doi:10.25673/35364. (cited on Page 30 and 41)

[ALY21] Ahmed Abdelwahab, Walter Lucia, and Amr Youssef. Covert Channels in Cyber-Physical Systems. *IEEE Control Systems Letters*, 5(4):1273–1278, October 2021. Conference Name: IEEE Control Systems Letters. doi:10.1109/LCSYS.2020.3033059. (cited on Page 26)

[AMO20] Chuadhry Mujeeb Ahmed, Aditya P. Mathur, and Martín Ochoa. NoiSense Print: Detecting Data Integrity Attacks on Sensor Measurements Using Hardware-based Fingerprints. *ACM Transactions*

on Privacy and Security, 24(1):2:1–2:35, September 2020. URL: https://dl.acm.org/doi/10.1145/3410447, doi:10.1145/3410447. (cited on Page 113)

[AZL+21] Robert Altschaffel, Fan Zhang, Jianghai Li, Jonas Hielscher, Tamas Holczer, Wen Si, and Kevin Lamshöft. Enhancing Safety and Security of Digital Instrumentation and Control System by Event Aggregation. In *ANS 12th Nuclear Plant Instrumentation, Control and Human-Machine Interface Technologies*. American Nuclear Society, June 2021. URL: https://www.ans.org/pubs/proceedings/issue-3123/. (cited on Page 14 and 167)

[AZM18] Chuadhry Mujeeb Ahmed, Jianying Zhou, and Aditya P. Mathur. Noise Matters: Using Sensor and Process Noise Fingerprint to Detect Stealthy Cyber Attacks and Authenticate sensors in CPS. In *Proceedings of the 34th Annual Computer Security Applications Conference*, ACSAC '18, pages 566–581, New York, NY, USA, 2018. Association for Computing Machinery. URL: https://dl.acm.org/doi/10.1145/3274694.3274748, doi:10.1145/3274694.3274748. (cited on Page 113)

[BBC+19] E. Biham, S. Bitan, Aviad Carmel, Alon Dankner, Uriel Malin, and A. Wool. Rogue 7 : Rogue Engineering-Station attacks on S7 Simatic PLCs, 2019. (cited on Page 8)

[BBT22] Emmanuel Aboah Boateng, J. W. Bruce, and Douglas A. Talbert. Anomaly detection for a water treatment system based on one-class neural network. *IEEE Access*, 10:115179–115191, 2022. Publisher: IEEE. (cited on Page 111)

[BEB18] Robert G Brown, Dirk Eddelbuettel, and David Bauer. Dieharder. *Duke University Physics Department Durham, NC*, pages 27708–0305, 2018. (cited on Page 142)

[BeI19] BeInCrypto.com. Indian Nuclear Power Plant Confirmed to Be Infected with Malware, November 2019. URL: https://beincrypto.com/indian-nuclear-power-plant-confirmed-to-be-infected-with-malware/. (cited on Page 158)

[Bis02] Matt Bishop. Computer security: art and science, 2002. (cited on Page 20)

[BKD23] Bernhard Birnbaum, Christian Kraetzer, and Jana Dittmann. Stego-Malware Attribution: Simple Signature and Content-based Features Derived and Validated from Classical Image Steganalysis on Five Exemplary Chosen Algorithms. pages 33–42, September 2023. URL: https://www.thinkmind.org/index.php?view=article&articleid=securware_2023_1_70_30047. (cited on Page 156 and 178)

[CA15] Brent C. Carrara and Carlisle Adams. On Characterizing and Measuring Out-of-Band Covert Channels. In *Proceedings of the 3rd ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec '15, pages 43–54, New York, NY, USA, June 2015. Association for Computing Machinery. doi:10.1145/2756601.2756604. (cited on Page 20, 23, and 46)

[CA16] Brent Carrara and Carlisle Adams. Out-of-Band Covert Channels - A Survey. *ACM Computing Surveys*, 49(2):23:1–23:36, June 2016. doi:10.1145/2938370. (cited on Page xiii, 3, 19, 20, 23, and 46)

[Cac04] Christian Cachin. An Information-Theoretic Model for Steganography. *Inf. Comput.*, 192(1):41–56, July 2004. (cited on Page 127)

[Cav21] Luca Caviglione. Trends and Challenges in Network Covert Channels Countermeasures. *Applied Sciences*, 11(4):1641, 2021. Publisher: Multidisciplinary Digital Publishing Institute. (cited on Page 5, 7, 27, and 164)

[CCM+18] Krzysztof Cabaj, Luca Caviglione, Wojciech Mazurczyk, Steffen Wendzel, Alan Woodward, and Sebastian Zander. The New Threats of Information Hiding: The Road Ahead. *IT Professional*, 20(3):31–39, May 2018. Conference Name: IT Professional. doi:10.1109/MITP.2018.032501746. (cited on Page 1, 5, and 7)

[CDB+12] Marco Conti, Sajal K. Das, Chatschik Bisdikian, Mohan Kumar, Lionel M. Ni, Andrea Passarella, George Roussos, Gerhard Tröster, Gene Tsudik, and Franco Zambonelli. Looking ahead in pervasive computing: Challenges and opportunities in the era of cyber–physical convergence. *Pervasive and Mobile Computing*, 8(1):2–21, February 2012. URL: https://www.sciencedirect.com/science/article/pii/S1574119211001271, doi:10.1016/j.pmcj.2011.10.001. (cited on Page 28)

[Cho21] Dhiman Deb Chowdhury. *Packet Timing: Network Time Protocol*, pages 103–116. Springer International Publishing, Cham, 2021. doi:10.1007/978-3-030-71179-5_7. (cited on Page 138)

[Cis11] Cisco Rockwell Automation. Converged Plantwide Ethernet (CPwE) Design and Implementation Guide, 2011. (cited on Page 30 and 41)

[CJV21] Martin Cooney, Eric Järpe, and Alexey Vinel. "Robot Steganography"?: Opportunities and Challenges. *arXiv:2108.00998 [cs]*, August 2021. arXiv: 2108.00998. URL: http://arxiv.org/abs/2108.00998. (cited on Page 26)

[CM22] Luca Caviglione and Wojciech Mazurczyk. Never Mind the Malware, Here's the Stegomalware. *IEEE Security & Privacy*, 20(5):101–106, September 2022. Conference Name: IEEE Security & Privacy. doi:10.1109/MSEC.2022.3178205. (cited on Page 1, 5, 7, and 178)

[CM24] Luca Caviglione and Wojciech Mazurczyk. You can't do that on protocols anymore: analysis of covert channels in ietf standards. *IEEE Network*, 38(5):255–263, 2024. doi:10.1109/MNET.2024.3352411. (cited on Page 177)

[Cra98] Scott Craver. On public-key steganography in the presence of an active warden. In *International Workshop on Information Hiding*, pages 355–368. Springer, 1998. (cited on Page 22)

[Cru18] Tiago Cruz. ICS_pcaps, 2018. URL: https://github.com/tjcruz-dei/ICS_PCAPS. (cited on Page 68 and 179)

[DEF20] DEFCON. DEFCON 23 - ICS Village Packet Captures, 2020. URL: https://t.ly/0GGLW. (cited on Page 68 and 179)

[Del15] Dell SecureWorks Counter Threat Unit™ Threat Intelligence. Stegoloader: A Stealthy Information Stealer, June 2015. URL: https://www.secureworks.com/research/stegoloader-a-stealthy-information-stealer. (cited on Page 22)

[DHH05] Jana Dittmann, Danny Hesse, and Reyk Hillert. Steganography and steganalysis in voice-over IP scenarios: operational aspects and first experiences with a new steganalysis tool set. In *Security, Steganography, and Watermarking of Multimedia Contents VII*, volume 5681, pages 607–618. SPIE, March 2005. URL: https://www.spiedigitallibrary.org/conference-proceedings-of-spie/5681/0000/Steganography-and-steganalysis-in-voice-over-IP-scenarios--operational/10.1117/12.586579.full, doi:10.1117/12.586579. (cited on Page 23)

[Dra22] Dragos Inc. CHERNOVITE's PIPEDREAM Malware Targeting Industrial Control Systems (ICS) | Dragos, April 2022. URL: https://www.dragos.com/blog/industry-news/chernovite-pipedream-malware-targeting-industrial-control-systems/. (cited on Page 1 and 66)

[DS21] Joanna F. DeFranco and Dimitrios Serpanos. The 12 Flavors of Cyber-physical Systems. *Computer*, 54(12):104–108, December 2021. Conference Name: Computer. doi:10.1109/MC.2021.3112319. (cited on Page 2 and 28)

[dWSK+23] Christian Schroeder de Witt, Samuel Sokota, J. Zico Kolter, Jakob Foerster, and Martin Strohmeier. Perfectly Secure Steganography Using Minimum Entropy Coupling, April 2023. arXiv:2210.14889 [cs]. URL: http://arxiv.org/abs/2210.14889, doi:10.48550/arXiv.2210.14889. (cited on Page 5)

[Ebi21] Takuji Ebinuma. GPS-SDR-SIM, December 2021. https://github.com/osqzss/gps-sdr-sim. (cited on Page 132)

[EG22] Muawia A. Elsadig and Ahmed Gafar. Covert Channel Detection: Machine Learning Approaches. *IEEE Access*, 10:38391–38405, 2022. Conference Name: IEEE Access. doi:10.1109/ACCESS.2022.3164392. (cited on Page 27 and 163)

[End16] Endace Technology Limited. IEEE 1588 PTP clock synchronization over a WAN backbone. Technical report, 2016. URL: https://www.endace.com/ptp-timing-whitepaper. (cited on Page 133)

[ESE22] ESET. Industroyer2: Industroyer reloaded, April 2022. Section: Ukraine Crisis – Digital Security Resource Center. URL: https://www.welivesecurity.com/2022/04/12/industroyer2-industroyer-reloaded/. (cited on Page 1)

[FAC+19] Ivo Frazão, Pedro Henriques Abreu, Tiago Cruz, Hélder Araújo, and Paulo Simões. Denial of Service Attacks: Detecting the Frailties of Machine Learning Algorithms in the Classification Process. In Eric Luiijf, Inga Žutautaitė, and Bernhard M. Hämmerli, editors, *Critical Information Infrastructures Security*, Lecture Notes in Computer Science, pages 230–235, Cham, 2019. Springer International Publishing. doi:10.1007/978-3-030-05849-4_19. (cited on Page 68 and 180)

[Fal20a] Robert Falcone. OilRig Targets Middle Eastern Telecommunications Organization and Adds Novel C2 Channel with Steganography to Its Inventory, July 2020. URL: https://unit42.paloaltonetworks.com/oilrig-novel-c2-channel-steganography/. (cited on Page 1)

[Fal20b] Robert Falcone. xHunt Campaign: Newly Discovered Backdoors Using C2, November 2020. URL: https://unit42.paloaltonetworks.com/xhunt-campaign-backdoors/. (cited on Page 2)

[FGCK23] Florian Bartusek, Glenn Diebetz, Christian Heidecke, and Kilian Oswald. Technische Aspekte der IT-Sicherheit SoSe 2023 - Team 2 Hide in Time, Covert Channels in NTS, Aufgabenstellung von K. Lamshöft und Jana Dittmann, 2023. (cited on Page 135, 138, and 140)

[FM19] Daniel Fox Franke and Aanchal Malhotra. NTP Client Data Minimization. Internet Draft draft-ietf-ntp-data-minimization-04, Internet Engineering Task Force, March 2019. Num Pages: 6. URL: https://datatracker.ietf.org/doc/draft-ietf-ntp-data-minimization-04. (cited on Page 141)

[FMC11] Nicolas Falliere, Liam O. Murchu, and Eric Chien. W32. stuxnet dossier. *White paper, Symantec Corp., Security Response*, 5(6):29, 2011. (cited on Page 1)

[Fri99] J. Fridrich. Applications of data hiding in digital images. In *ISSPA '99. Proceedings of the Fifth International Symposium on Signal Processing and its Applications (IEEE Cat. No.99EX359)*, volume 1, pages 9 vol.1–, August 1999. doi:10.1109/ISSPA.1999.818099. (cited on Page xiii, 23, and 24)

[Fri09] Jessica Fridrich. *Steganography in digital media: principles, algorithms, and applications*. Cambridge University Press, 2009. (cited on Page 5)

[FST+20] Daniel Fox Franke, Dieter Sibold, Kristof Teichel, Marcus Dansarie, and Ragnar Sundblad. Network Time Security for the Network Time Protocol. RFC 8915, September 2020. URL: https://rfc-editor.org/rfc/rfc8915.txt, doi:10.17487/RFC8915. (cited on Page xvi, xx, 123, 126, 134, 135, 136, and 137)

[FW19] Barbara Filkins and Doug Wylie. SANS 2019 State of OT/ICS Cybersecurity Survey | SANS Institute, November 2019. URL: https://www.sans.org/white-papers/38995/. (cited on Page 7)

[GAJM17] Jonathan Goh, Sridhar Adepu, Khurum Nazir Junejo, and Aditya Mathur. A Dataset to Support Research in the Design of Secure Water Treatment Systems. In Grigore Havarneanu, Roberto Setola, Hypatia Nassopoulos, and Stephen Wolthusen, editors, *Critical Information Infrastructures Security*, Lecture Notes in Computer Science, pages 88–99, Cham, 2017. Springer International Publishing. doi:10.1007/978-3-319-71368-7_8. (cited on Page 111 and 180)

[GB19] Mordechai Guri and Dima Bykhovsky. aIR-Jumper: Covert air-gap exfiltration/infiltration via security cameras & infrared (IR). *Computers & Security*, 82:15–29, 2019. URL: https://www.sciencedirect.com/science/article/pii/S0167404818307193, doi:10.1016/j.cose.2018.11.004. (cited on Page 2)

[GBMF20] Marcus Geiger, Jochen Bauer, Michael Masuch, and Jörg Franke. An Analysis of Black Energy 3, Crashoverride, and Trisis, Three Malware Approaches Targeting Operational Technology Systems. In *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, volume 1, pages 1537–1543, September 2020. ISSN: 1946-0759. doi:10.1109/ETFA46521.2020.9212128. (cited on Page 1)

[GBWG19] Christopher Greer, Martin Burns, David Wollman, and Edward Griffor. Cyber-physical systems and internet of things. Technical Report NIST SP 1900-202, National Institute of Standards and Technology, Gaithersburg, MD, March 2019. URL: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.1900-202.pdf, doi:10.6028/NIST.SP.1900-202. (cited on Page 28)

[GE18] Mordechai Guri and Yuval Elovici. Bridgeware: the air-gap malware. *Communications of the ACM*, 61(4):74–82, 2018. doi:10.1145/3177230. (cited on Page 2)

[Gil08] Helen Gill. From Vision to Reality: Cyber-Physical Systems, 2008. URL: https://labs.ece.uw.edu/nsl/aar-cps/Gill_HCSS_Transportation_Cyber-Physical_Systems_2008.pdf. (cited on Page 28)

[Gil19] Martin Giles. Triton is the world's most murderous malware, and it's spreading, May 2019. URL: https://www.technologyreview.com/2019/03/05/103328/cybersecurity-critical-infrastructure-triton-malware/. (cited on Page 1)

[GMME15] Mordechai Guri, Matan Monitz, Yisroel Mirski, and Yuval Elovici. BitWhisper: Covert Signaling Channel between Air-Gapped Computers Using Thermal Manipulations. In *2015 IEEE 28th Computer Security Foundations Symposium*, pages 276–289, July 2015. ISSN: 2377-5459. doi:10.1109/CSF.2015.26. (cited on Page 23)

[GRSS12] Holger Giese, Bernhard Rumpe, Bernhard Schätz, and Janos Sztipanovits. Science and Engineering of Cyber-Physical Systems (Dagstuhl Seminar 11441). *Dagstuhl Reports*, 1(11):1–22, 2012. Place: Dagstuhl, Germany Publisher: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. URL: http://drops.dagstuhl.de/opus/volltexte/2012/3375, doi:10.4230/DagRep.1.11.1. (cited on Page 28)

[GSMZ14] Luis Garcia, Henry Senyondo, Stephen McLaughlin, and Saman Zonouz. Covert channel communication through physical interdependencies in cyber-physical infrastructures. In *2014 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 952–957, November 2014. doi:10.1109/SmartGridComm.2014.7007771. (cited on Page 25)

[Gun17] Hendra Gunadi. Comparison of IDS Suitability for Covert Channels Detection. 2017. URL: https://www.semanticscholar.org/paper/Comparison-of-IDS-Suitability-for-Covert-Channels-Gunadi/a86434ddfe87e713f6157b68b6ca6b71c1af1fd3. (cited on Page 27 and 164)

[Gur18a] Mordechai Guri. BeatCoin: Leaking Private Keys from Air-Gapped Cryptocurrency Wallets. In *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 1308–1316, July 2018. doi:10.1109/Cybermatics_2018.2018.00227. (cited on Page 2)

[Gur18b] Mordechai Guri. Optical Covert Channel from Air-Gapped Networks via Remote Orchestration of Router/Switch LEDs. In *2018 European Intelligence and Security Informatics Conference (EISIC)*, pages 54–60, October 2018. doi:10.1109/EISIC.2018.00016. (cited on Page 46)

[Gur20] Mordechai Guri. AIR-FI: Generating Covert Wi-Fi Signals from Air-Gapped Computers. *arXiv:2012.06884 [cs]*, December 2020. arXiv:

2012.06884. URL: http://arxiv.org/abs/2012.06884. (cited on Page 2)

[GZBE20] Mordechai Guri, Boris Zadov, Dima Bykhovsky, and Yuval Elovici. PowerHammer: Exfiltrating Data From Air-Gapped Computers Through Power Lines. *IEEE Transactions on Information Forensics and Security*, 15:1879–1890, 2020. Conference Name: IEEE Transactions on Information Forensics and Security. doi:10.1109/TIFS.2019.2952257. (cited on Page 23)

[GZE17] Mordechai Guri, Boris Zadov, and Yuval Elovici. LED-it-GO: Leaking (a lot of) Data from Air-Gapped Computers via the (small) Hard Drive LED. In *International conference on detection of intrusions and malware, and vulnerability assessment*, pages 161–184. Springer, 2017. (cited on Page 23 and 46)

[GZE20] Mordechai Guri, Boris Zadov, and Yuval Elovici. ODINI: Escaping Sensitive Data From Faraday-Caged, Air-Gapped Computers via Magnetic Fields. *IEEE Transactions on Information Forensics and Security*, 15:1190–1203, 2020. Conference Name: IEEE Transactions on Information Forensics and Security. doi:10.1109/TIFS.2019.2938404. (cited on Page 46)

[HAL+20] Mario Hildebrandt, Robert Altschaffel, Kevin Lamshöft, Mathias Lange, Martin Szemkus, Tom Neubert, Claus Vielhauer, Yongdian Ding, and Jana Dittmann. Threat analysis of steganographic and covert communication in nuclear I&C systems. In *International Conference on Nuclear Security: Sustaining and Strengthening Efforts*, volume 10, page 14, 2020. (cited on Page 14)

[Han23] Kai Hansmann. *Evaluation of the possibility to automate Covert Channel Analysis on the example of OPC UA. Supervisor: Kevin Lamshöft and Jana Dittmann.* Bachelor Thesis, Otto-von-Guericke University, Magdeburg, Germany, October 2023. (cited on Page 175)

[HCA11] Eric M. Hutchins, Michael J. Cloppert, and Rohan M. Amin. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Leading Issues in Information Warfare & Security Research*, 1(1):80, 2011. (cited on Page 6, 22, and 33)

[HG12] Imre Horváth and Bart H M Gerritsen. CYBER-PHYSICAL SYSTEMS: CONCEPTS, TECHNOLOGIES AND IMPLEMENTATION PRINCIPLES. page 19, 2012. (cited on Page 28)

[HK19a] Amir Herzberg and Yehonatan Kfir. The chatty-sensor: a provably-covert channel in cyber physical systems. In *Proceedings of the 35th Annual Computer Security Applications Conference*, ACSAC '19, pages 638–649, New York, NY, USA, 2019. Association for Computing Machinery. doi:10.1145/3359789.3359794. (cited on Page 25, 45, 47, 51, and 175)

[HK19b] Amir Herzberg and Yehonatan Kfir. The Leaky Actuator: A Provably-covert Channel in Cyber Physical Systems. In *Proceedings of the ACM Workshop on Cyber-Physical Systems Security & Privacy*, CPS-SPC'19, pages 87–98, New York, NY, USA, November 2019. Association for Computing Machinery. doi:10.1145/3338499.3357358. (cited on Page 45, 47, and 175)

[HLD+20] Mario Hildebrandt, Kevin Lamshöft, Jana Dittmann, Tom Neubert, and Claus Vielhauer. Information Hiding in Industrial Control Systems: An OPC UA based Supply Chain Attack and its Detection. In *Proceedings of the 2020 ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec '20, pages 115–120, New York, NY, USA, June 2020. Association for Computing Machinery. doi:10.1145/3369412.3395068. (cited on Page 13, 118, 123, 170, and 175)

[HLKD21] Jonas Hielscher, Kevin Lamshöft, Christian Krätzer, and Jana Dittmann. A Systematic Analysis of Covert Channels in the Network Time Protocol. In *The 16th International Conference on Availability, Reliability and Security*, ARES 2021, pages 1–11, New York, NY, USA, August 2021. Association for Computing Machinery. doi:10.1145/3465481.3470075. (cited on Page 13, 123, 124, 127, 141, and 151)

[Ho19] Jun-Won Ho. Covert Channel Establishment Through the Dynamic Adaptation of the Sequential Probability Ratio Test to Sensor Data in IoT. *IEEE Access*, 7:146093–146107, 2019. Conference Name: IEEE Access. doi:10.1109/ACCESS.2019.2945974. (cited on Page 26)

[HS16] United States Cyber Emergency Response Team Homeland Security. Recommended Practice: Improving Industrial Control System Cybersecurity with Defense-in-Depth Strategies. In *Homeland Security Digital Library*. United States. Department of Homeland Security; Industrial Control Systems Cyber Emergency Response Team, September 2016. URL: https://www.hsdl.org/?abstract&did=. (cited on Page xiii, xiv, 30, 32, 38, 40, and 41)

[HZW21] Laura Hartmann, Sebastian Zillien, and Steffen Wendzel. Reset- and Reconnection-based Covert Channels in CoAP. In *Proceedings of the 2021 European Interdisciplinary Cybersecurity Conference*, EICC '21, pages 66–71, New York, NY, USA, November 2021. Association for Computing Machinery. URL: https://dl.acm.org/doi/10.1145/3487405.3487660, doi:10.1145/3487405.3487660. (cited on Page 25)

[IAE21] IAEA International Atomic Energy Agency. *Computer security techniques for nuclear facilities*. Number 17-T (Rev. 1) in Technical guidance. INTERNATIONAL ATOMIC ENERGY AGENCY, Vienna, 2021. URL: https://www.iaea.org/publications/14729/

computer-security-techniques-for-nuclear-facilities. (cited on Page xiii, 32, 62, and 165)

[IEE08] IEEE Standards Association. IEEE 1588 - Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems. Standard, IEEE, 2008. URL: https://standards.ieee.org. (cited on Page 123 and 125)

[IYC+17] Jun Inoue, Yoriyuki Yamagata, Yuqi Chen, Christopher M. Poskitt, and Jun Sun. Anomaly detection for a water treatment system using unsupervised machine learning. In *2017 IEEE international conference on data mining workshops (ICDMW)*, pages 1058–1065. IEEE, 2017. (cited on Page 111)

[JLY10] Daryl Johnson, Peter Lutz, and Bo Yuan. Behavior-based covert channel in cyberspace. In *Intelligent decision making systems*, pages 311–318. World Scientific, 2010. (cited on Page 4 and 20)

[Ker83] Auguste Kerckhoffs. La cryptographie militaire. *J. des Sci. Militaires*, 9:161–191, 1883. (cited on Page 57)

[Ker07] Andrew D. Ker. Batch Steganography and Pooled Steganalysis. In Jan L. Camenisch, Christian S. Collberg, Neil F. Johnson, and Phil Sallee, editors, *Information Hiding*, Lecture Notes in Computer Science, pages 265–281, Berlin, Heidelberg, 2007. Springer. doi:10.1007/978-3-540-74124-4_18. (cited on Page 114)

[KKK+18] Prashanth Krishnamurthy, Farshad Khorrami, Ramesh Karri, David Paul-Pena, and Hossein Salehghaffari. Process-Aware Covert Channels Using Physical Instrumentation in Cyber-Physical Systems. *IEEE Transactions on Information Forensics and Security*, 13(11):2761–2771, November 2018. Conference Name: IEEE Transactions on Information Forensics and Security. doi:10.1109/TIFS.2018.2833063. (cited on Page 26)

[KLG15] Marina Krotofil, Jason Larsen, and Dieter Gollmann. The Process Matters: Ensuring Data Veracity in Cyber-Physical Systems. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security*, ASIA CCS '15, pages 133–144, New York, NY, USA, April 2015. Association for Computing Machinery. URL: https://dl.acm.org/doi/10.1145/2714576.2714599, doi:10.1145/2714576.2714599. (cited on Page 113)

[KM15] Siddhartha Kumar Khaitan and James D. McCalley. Design Techniques and Applications of Cyberphysical Systems: A Survey. *IEEE Systems Journal*, 9(2):350–365, June 2015. Conference Name: IEEE Systems Journal. doi:10.1109/JSYST.2014.2322503. (cited on Page 2, 28, and 29)

[Kop20] Dr Henning Kopp. Attacking a random number generator, October 2020. URL: https://www.schutzwerk.com/en/blog/attacking-a-rng/. (cited on Page 142)

[KP00] Stefan Katzenbeisser and Fabien A. Petitcolas. Information hiding techniques for steganography and digital watermarking, 2000. (cited on Page 57)

[KWJ21] Tomasz Koziak, Katarzyna Wasielewska, and Artur Janicki. How to Make an Intrusion Detection SystemAware of Steganographic Transmission. In *European Interdisciplinary Cybersecurity Conference*, pages 77–82. Association for Computing Machinery, New York, NY, USA, November 2021. URL: https://doi.org/10.1145/3487405.3487421. (cited on Page 27 and 164)

[LAD23] Kevin Lamshöft, Robert Altschaffel, and Jana Dittmann. Innovative sicherheitsleittechnik, bewertung und verbesserung der sicherheit gegenüber schadprogrammen mit verdeckten funktionen und wirkungsweisen, 2023. doi:10.2314/KXP:1878724320. (cited on Page iv and vi)

[Lam73] Butler W. Lampson. A note on the confinement problem. *Communications of the ACM*, 16(10):613–615, 1973. Publisher: ACM New York, NY, USA. (cited on Page 20)

[Lam23] Kevin Lamshöft. Information hiding in cyber-physical systems - covert channel dataset, December 2023. doi:10.24352/ub.ovgu-2023-112. (cited on Page 16 and 179)

[Lat86] Donald C. Latham. Department of defense trusted computer system evaluation criteria. *Department of Defense*, 198, 1986. (cited on Page 20)

[LD20] Kevin Lamshöft and Jana Dittmann. Assessment of Hidden Channel Attacks: Targetting Modbus/TCP. *IFAC-PapersOnLine*, 53(2):11100–11107, January 2020. URL: https://www.sciencedirect.com/science/article/pii/S240589632030536X, doi:10.1016/j.ifacol.2020.12.258. (cited on Page xiv, xv, xix, 13, 23, 38, 58, 59, 60, 61, 66, 67, 73, 79, 118, 151, and 156)

[LD22] Kevin Lamshöft and Jana Dittmann. Covert Channels in Network Time Security. In *Proceedings of the 2022 ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec '22)*, page 11, Santa Barbara, June 2022. ACM. doi:https://doi.org/10.1145/3531536.3532947. (cited on Page xvi, 13, 57, 123, 124, 134, 135, 136, 139, 141, 142, 151, 156, and 157)

[Lem16] Antoine Lemay. Modbus_dataset, 2016. original-date: 2016-06-26T22:50:26Z. URL: https://github.com/antoine-lemay/Modbus_dataset. (cited on Page 68 and 180)

[LFK16] Antoine Lemay, José M. Fernandez, and Scott Knight. A Modbus command and control channel. In *2016 Annual IEEE Systems Conference (SysCon)*, pages 1–6, April 2016. doi:10.1109/SYSCON.2016.7490631. (cited on Page 25, 68, and 180)

[LHA+22] Kevin Lamshöft, Mario Hildebrandt, Robert Altschaffel, Oliver Keil, Ivo Hempel, Jana Dittmann, Tom Neubert, and Claus Vielhauer. *Resilience Against and Detection of Information Hiding in Nuclear Instrumentation and Control Systems within the Scope of NSS 17-T*, volume KERNTECHNIK 2022. INFORUM Verlags- und Verwaltungsgesellschaft mbH, Berlin, July 2022. URL: https://www.kerntechnik.com/kerntechnik-wAssets/docs/2022/Proceedings.zip. (cited on Page 14)

[LHKD22] Kevin Lamshöft, Jonas Hielscher, Christian Krätzer, and Jana Dittmann. The Threat of Covert Channels in Network Time Synchronisation Protocols. *Journal of Cyber Security and Mobility*, pages 165–204, 2022. (cited on Page xvi, 13, 123, 124, 126, 127, 130, 132, and 141)

[LJ14] Carlos Leonardo and Daryl Johnson. MODBUS covert channel. In *Proceedings of the International Conference on Security and Management (SAM)*, page 1. The Steering Committee of The World Congress in Computer Science, Computer . . . , 2014. (cited on Page 25)

[LK17] Antoine Lemay and Scott Knight. A timing-based covert channel for SCADA networks. In *2017 International Conference on Cyber Conflict (CyCon U.S.)*, pages 8–15, November 2017. doi:10.1109/CYCONUS.2017.8167507. (cited on Page 25)

[LMS18] Robert M Lee, Ben Miller, and Mark Stacey. Collection Management Frameworks – Looking Beyond Asset Inventories in Preparation for and Response to Cyber Threats. page 13, 2018. (cited on Page 28)

[LNH+22] Kevin Lamshöft, Tom Neubert, Jonas Hielscher, Claus Vielhauer, and Jana Dittmann. Knock, knock, log: Threat analysis, detection & mitigation of covert channels in syslog using port scans as cover. *Forensic Science International: Digital Investigation*, 40:301335, April 2022. URL: https://www.sciencedirect.com/science/article/pii/S266628172200004X, doi:10.1016/j.fsidi.2022.301335. (cited on Page 13, 57, 155, 156, 157, 158, 166, 175, and 176)

[LNK+21] Kevin Lamshöft, Tom Neubert, Christian Krätzer, Claus Vielhauer, and Jana Dittmann. Information Hiding in Cyber Physical Systems: Challenges for Embedding, Retrieval and Detection using Sensor Data of the SWAT Dataset. In *Proceedings of the 2021 ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec '21, pages 113–124, New York, NY, USA, June 2021. Association for Computing Machinery. doi:10.1145/3437880.3460413. (cited on Page xvi,

13, 47, 48, 51, 52, 108, 110, 112, 114, 117, 119, 120, 122, 152, 155, 156, 157, 158, and 180)

[LNL+20] Kevin Lamshöft, Tom Neubert, Mathias Lange, Robert Altschaffel, Mario Hildebrandt, Yongjian Ding, claus Vielhauer, and Jana Dittmann. Novel Challenges for Anomaly Detection in I&C Networks: Strategic Preparation for the Advent of Information Hiding based Attacks. *atw - International Journal for Nuclear Power*, 65:504–508, October 2020. (cited on Page 14)

[LS07] Pierre L'ecuyer and Richard Simard. Testu01: Ac library for empirical testing of random number generators. *ACM Transactions on Mathematical Software (TOMS)*, 33(4):1–40, 2007. (cited on Page 142)

[LY20] Walter Lucia and Amr Youssef. Wyner wiretap-like encoding scheme for cyber-physical systems. *IET Cyber-Physical Systems: Theory & Applications*, 5(4):359–365, October 2020. Publisher: IET Digital Library. URL: https://digital-library.theiet.org/content/journals/10.1049/iet-cps.2020.0012, doi:10.1049/iet-cps.2020.0012. (cited on Page 26)

[LY21] Walter Lucia and Amr Youssef. Covert channels in stochastic cyber-physical systems. *IET Cyber-Physical Systems: Theory & Applications*, n/a(n/a), 2021. _eprint: https://ietresearch.onlinelibrary.wiley.com/doi/pdf/10.1049/cps2.12020. URL: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/cps2.12020, doi:10.1049/cps2.12020. (cited on Page 2 and 26)

[MBKM10] Jim Martin, Jack Burbank, William Kasch, and Professor David L. Mills. Network Time Protocol Version 4: Protocol and Algorithms Specification. Request for Comments RFC 5905, Internet Engineering Task Force, June 2010. Num Pages: 110. URL: https://datatracker.ietf.org/doc/rfc5905, doi:10.17487/RFC5905. (cited on Page 123 and 125)

[MC15] Wojciech Mazurczyk and Luca Caviglione. Information Hiding as a Challenge for Malware Detection. *IEEE Security Privacy*, 13(2):89–93, March 2015. Conference Name: IEEE Security Privacy. doi:10.1109/MSP.2015.33. (cited on Page 4 and 22)

[McG08] David McGrew. An Interface and Algorithms for Authenticated Encryption. RFC 5116, January 2008. URL: https://rfc-editor.org/rfc/rfc5116.txt, doi:10.17487/RFC5116. (cited on Page 126)

[MH10] Professor David L. Mills and Brian Haberman. Network Time Protocol Version 4: Autokey Specification. RFC 5906, June 2010. URL: https://rfc-editor.org/rfc/rfc5906.txt, doi:10.17487/RFC5906. (cited on Page 126)

[Mil92] David L. Mills. Network Time Protocol (Version 3) Specification, Implementation and Analysis. RFC 1305, March 1992. URL: https://rfc-editor.org/rfc/rfc1305.txt, doi:10.17487/RFC1305. (cited on Page 126)

[MM20] Morten Marstander and Matteo Malvica. SNIcat: Circumventing the guardians, 2020. URL: https://www.mnemonic.no/blog/introducing-snicat/. (cited on Page 7 and 167)

[Mod06] Modbus Organization. MODBUS Messaging on TCP/IP Implementation Guide V1.0b, 2006. URL: http://modbus.org/specs.php. (cited on Page xv, 66, and 67)

[Mod12] Modbus Organization. Modbus Protocol Specification V1.1b3, 2012. URL: http://modbus.org/specs.php. (cited on Page 66)

[Mod21] Modbus Organization. Modbus Specifications and Implementation Guides, 2021. URL: https://modbus.org/specs.php. (cited on Page 8 and 83)

[MSWC19] Wojciech Mazurczyk, Przemysław Szary, Steffen Wendzel, and Luca Caviglione. Towards Reversible Storage Network Covert Channels. In *Proceedings of the 14th International Conference on Availability, Reliability and Security*, ARES '19, pages 1–8, New York, NY, USA, August 2019. Association for Computing Machinery. doi:10.1145/3339252.3341493. (cited on Page 59)

[MVH+21] Aleksandra Mileva, Aleksandar Velinov, Laura Hartmann, Steffen Wendzel, and Wojciech Mazurczyk. Comprehensive analysis of MQTT 5.0 susceptibility to network covert channels. *Computers & Security*, 104:102207, 2021. URL: https://www.sciencedirect.com/science/article/pii/S0167404821000316, doi:10.1016/j.cose.2021.102207. (cited on Page 25)

[MVS18] Aleksandra Mileva, Aleksandar Velinov, and Done Stojanov. New Covert Channels in Internet of Things. pages 30–36, Venice, Italy, 2018. URL: https://www.thinkmind.org/index.php?view=instance&instance=SECURWARE+2018. (cited on Page 25)

[MW19] Mayra Macas and Chunming Wu. An unsupervised framework for anomaly detection in a water treatment system. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 1298–1305. IEEE, 2019. (cited on Page 111)

[MWAVS16] Wojciech Mazurczyk, Steffen Wendzel, Ignacio Azagra Villares, and Krzysztof Szczypiorski. On importance of steganographic cost for network steganography. *Security and Communication Networks*, 9(8):781–790, 2016. Publisher: Wiley Online Library. (cited on Page 24)

[MWC18] Wojciech Mazurczyk, Steffen Wendzel, and Krzysztof Cabaj. Towards Deriving Insights into Data Hiding Methods Using Pattern-based Approach. In *Proceedings of the 13th International Conference on Availability, Reliability and Security*, ARES 2018, pages 1–10, New York, NY, USA, August 2018. Association for Computing Machinery. doi:10.1145/3230833.3233261. (cited on Page xv, xix, 8, 24, 38, 45, 46, 54, 73, 74, 76, 77, 79, 106, 124, 150, 152, and 169)

[MWZ+16] Wojciech Mazurczyk, Steffen Wendzel, Sebastian Zander, Amir Houmansadr, and Krzysztof Szczypiorski. *Information Hiding in Communication Networks: Fundamentals, Mechanisms, Applications, and Countermeasures*. Wiley-IEEE Press, 1st edition, 2016. (cited on Page xiii, xvii, 19, 20, 21, 22, 23, 24, 27, 38, 55, 152, 161, 162, and 163)

[Nat21] National Science Foundation. Cyber-Physical Systems (CPS) (nsf21551) | NSF - National Science Foundation, 2021. URL: https://www.nsf.gov/pubs/2021/nsf21551/nsf21551.htm. (cited on Page 28)

[Nel16] Nell Nelson. The Impact of Dragonfly Malware on Industrial Control Systems | SANS Institute, January 2016. URL: https://www.sans.org/white-papers/36672/. (cited on Page 1)

[Net20] Netnod. How does nts wok and why is it important. 2020. URL: https://www.netnod.se/time-and-frequency/white-paper-how-does-nts-work-and-why-is-it-important. (cited on Page xvi, 134, and 136)

[New18] NewBee119. ctf_ics_traffic, 2018. [Github] https://github.com/NewBee119/ctf_ics_traffic. URL: https://github.com/NewBee119/ctf_ics_traffic. (cited on Page 68 and 180)

[NKK+22] Nathan Brubaker, Keith Lunden, Ken Proska, Muhammad Umair, Daniel Kapellmann Zafra, Corey Hildebrandt, and Rob Caldwell. INCONTROLLER: New State-Sponsored Cyber Attack Tools Target Multiple Industrial Control Systems. Technical report, Mandiant, April 2022. URL: https://www.mandiant.com/resources/blog/incontroller-state-sponsored-ics-tool. (cited on Page 1 and 66)

[NPG+21] Ben Nassi, Yaron Pirutin, Tomer Galor, Yuval Elovici, and Boris Zadov. Glowworm Attack: Optical TEMPEST Sound Recovery via a Device's Power Indicator LED. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, pages 1900–1914, New York, NY, USA, November 2021. Association for Computing Machinery. doi:10.1145/3460120.3484775. (cited on Page 23)

[OAS21] OASIS MQTT Technical Committee. MQTT Specification, 2021. URL: https://mqtt.org/mqtt-specification/. (cited on Page 8)

[Obr15] Luciana Obregon. Secure Architecture for Industrial Control Systems | SANS Institute, October 2015. URL: https://www.sans.org/white-papers/36327/. (cited on Page xiv, 31, 32, 38, 40, 41, and 42)

[OPC08] OPC Foundation. Unified Architecture, 2008. URL: https://opcfoundation.org/about/opc-technologies/opc-ua/. (cited on Page 8)

[PAK99] F.A.P. Petitcolas, R.J. Anderson, and M.G. Kuhn. Information hiding-a survey. *Proceedings of the IEEE*, 87(7):1062–1078, July 1999. Conference Name: Proceedings of the IEEE. doi:10.1109/5.771065. (cited on Page xiii, 3, and 19)

[Pla18] Platforms4CPS. D4.3 Collaboration on the foundations of CPS Engineering. Technical report, 2018. URL: https://ec.europa.eu/research/participants/documents/downloadPublic?documentIds=080166e5bec10e68&appId=PPGMS. (cited on Page 28)

[RDMMR19] A. Robles-Durazno, N. Moradpoor, J. McWhinnie, and G. Russell. WaterLeakage: A Stealthy Malware for Data Exfiltration on Industrial Control Systems Using Visual Channels*. In *2019 IEEE 15th International Conference on Control and Automation (ICCA)*, pages 724–731, July 2019. ISSN: 1948-3457. doi:10.1109/ICCA.2019.8899564. (cited on Page 26)

[Res10] Eric Rescorla. Keying Material Exporters for Transport Layer Security (TLS). RFC 5705, March 2010. URL: https://rfc-editor.org/rfc/rfc5705.txt, doi:10.17487/RFC5705. (cited on Page 135 and 146)

[Res18] Eric Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446, August 2018. URL: https://rfc-editor.org/rfc/rfc8446.txt, doi:10.17487/RFC8446. (cited on Page 126)

[Rha23] Rhadamanthys. Malware-Traffic-Analysis.net - 2023-01-03 - Google ad –> fake Notepad++ page –> Rhadamanthys Stealer, March 2023. URL: https://www.malware-traffic-analysis.net/2023/01/03/index.html. (cited on Page 22)

[RMT16] Robert M. Lee, Michael J. Assante, and Tim Conway. ICS Defense Use Case (DUC) #4: Analysis of the recent reports of attacks on US infrastructure by Iranian Actors, May 2016. (cited on Page 6)

[Rob20] Robert M. Lee. Cyberville - 2020 SANS ICS Virtual Conference CTF, 2020. https://malcolm.fyi/examples/Cyberville.pcap, https://www.sans.org/media/DISC-event-details.pdf, https://www.sans.org/blog/disc-sans-ics-virtual-conference-and-ics-ctf-event-details/. URL: https://www.sans.org/blog/disc-sans-ics-virtual-conference-and-ics-ctf-event-details/. (cited on Page 68 and 179)

[RSS19] Denis Reilly, Harlan Stenn, and Dieter Sibold. Network Time Protocol Best Current Practices. RFC 8633, July 2019. URL: https://rfc-editor.org/rfc/rfc8633.txt, doi:10.17487/RFC8633. (cited on Page 126)

[RTM16] Robert M. Lee, Tim Conway, and Michael J. Assante. Analysis of the cyber attack on the Ukrainian power grid. *Electricity Information Sharing and Analysis Center (E-ISAC)*, 388, 2016. (cited on Page 1)

[SAM+18] Blake E. Strom, Andy Applebaum, Douglas P. Miller, Kathryn C. Nickels, Adam G. Pennington, and Cody B. Thomas. MITRE ATT&CK™ : Design and Philosophy. July 2018. URL: https://www.mitre.org/publications/technical-papers/mitre-attack-design-and-philosophy. (cited on Page 6, 7, and 33)

[SB13] Pascal Schöttle and Rainer Böhme. A Game-Theoretic Approach to Content-Adaptive Steganography. In Matthias Kirchner and Dipak Ghosal, editors, *Information Hiding*, Lecture Notes in Computer Science, pages 125–141, Berlin, Heidelberg, 2013. Springer. doi:10.1007/978-3-642-36373-3_9. (cited on Page 114)

[SFL18] Dmitry Shalyga, Pavel Filonov, and Andrey Lavrentyev. Anomaly detection for water treatment system based on neural network with automatic architecture optimization. *arXiv preprint arXiv:1807.07282*, 2018. (cited on Page 111)

[SGLW08] Lui Sha, Sathish Gopalakrishnan, Xue Liu, and Qixin Wang. Cyber-Physical Systems: A New Frontier. In *2008 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (sutc 2008)*, pages 1–9, June 2008. doi:10.1109/SUTC.2008.85. (cited on Page 28)

[Sha48] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, July 1948. Conference Name: The Bell System Technical Journal. doi:10.1002/j.1538-7305.1948.tb01338.x. (cited on Page 20)

[Sie19] Siemens AG. What properties, advantages and special features does the S7 protocol offer? - ID: 26483647 - Industry Support Siemens, November 2019. URL: https://support.industry.siemens.com/cs/ww/en/view/26483647. (cited on Page 8)

[Sim84] Gustavus J. Simmons. The prisoners' problem and the subliminal channel. In *Advances in Cryptology*, pages 51–67. Springer, 1984. (cited on Page xiii, xiv, 21, 55, and 58)

[SPL+15] Keith Stouffer, Victoria Pillitteri, Suzanne Lightman, Marshall Abrams, and Adam Hahn. Guide to Industrial Control Systems (ICS) Security. Technical Report NIST SP 800-82r2, National Institute of Standards and Technology, June 2015. URL: https://nvlpubs.

nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r2.pdf, doi:10.6028/NIST.SP.800-82r2. (cited on Page xiii, xiv, 2, 29, 30, 31, 32, 38, 40, 41, 42, and 43)

[VMWM19] Aleksandar Velinov, Aleksandra Mileva, Steffen Wendzel, and Wojciech Mazurczyk. Covert Channels in the MQTT-Based Internet of Things. *IEEE Access*, 7:161899–161915, 2019. Conference Name: IEEE Access. doi:10.1109/ACCESS.2019.2951425. (cited on Page 8 and 25)

[WCM+21] Steffen Wendzel, Luca Caviglione, Wojciech Mazurczyk, Aleksandra Mileva, Jana Dittmann, Christian Krätzer, Kevin Lamshöft, Claus Vielhauer, Laura Hartmann, Jörg Keller, and Tom Neubert. A Revised Taxonomy of Steganography Embedding Patterns. In *The 16th International Conference on Availability, Reliability and Security*, ARES 2021, pages 1–12, New York, NY, USA, August 2021. Association for Computing Machinery. doi:10.1145/3465481.3470069. (cited on Page xx, 2, 7, 45, 137, and 150)

[WCM+22] Steffen Wendzel, Luca Caviglione, Wojciech Mazurczyk, Aleksandra Mileva, Jana Dittmann, Christian Krätzer, Kevin Lamshöft, Claus Vielhauer, Laura Hartmann, Jörg Keller, Tom Neubert, and Sebastian Zillien. A Generic Taxonomy for Steganography Methods, July 2022. URL: https://www.techrxiv.org/articles/preprint/A_Generic_Taxonomy_for_Steganography_Methods/20215373/2, doi:10.36227/techrxiv.20215373.v2. (cited on Page xiii, 4, 24, and 54)

[Web20] Don C. Weber. Responding to Incidents in Industrial Control Systems: Identifying Threats/Reactions and Developing the IR Process | SANS Institute, May 2020. URL: https://www.sans.org/white-papers/39595/. (cited on Page 6)

[Wen12a] Steffen Wendzel. Covert and side channels in buildings and the prototype of a building-aware active warden. In *2012 IEEE International Conference on Communications (ICC)*, pages 6753–6758, June 2012. ISSN: 1938-1883. doi:10.1109/ICC.2012.6364876. (cited on Page 26 and 51)

[Wen12b] Steffen Wendzel. The Problem of Traffic Normalization Within a Covert Channel's Network Environment Learning Phase. May 2012. (cited on Page 27, 164, and 166)

[Wil92] Theodore Joseph Williams. *The Purdue enterprise reference architecture: a technical guide for CIM planning and implementation*. Instrument Society of America, 1992. (cited on Page xiii, 2, 29, 30, 31, 41, and 62)

[Wir21] Wireshark. Protocols/ptp - The Wireshark Wiki, 2021. URL: https://wiki.wireshark.org/Protocols/ptp. (cited on Page 133)

[WKR12] Steffen Wendzel, Benjamin Kahler, and Thomas Rist. Covert Channels and Their Prevention in Building Automation Protocols: A Prototype Exemplified Using BACnet. In *Proceedings of the 2012 IEEE International Conference on Green Computing and Communications*, GREENCOM '12, pages 731–736, USA, November 2012. IEEE Computer Society. doi:10.1109/GreenCom.2012.120. (cited on Page 8 and 25)

[WMH17a] Steffen Wendzel, Wojciech Mazurczyk, and Georg Haas. Don't You Touch My Nuts: Information Hiding in Cyber Physical Systems. In *2017 IEEE Security and Privacy Workshops (SPW)*, pages 29–34, May 2017. doi:10.1109/SPW.2017.40. (cited on Page 8, 26, 47, 51, 52, and 114)

[WMH17b] Steffen Wendzel, Wojciech Mazurczyk, and Georg Haas. Steganography for Cyber-physical Systems. *Journal of Cyber Security and Mobility*, 6(2):105–126, April 2017. Publisher: River Publishers. URL: https://riverpublishers.com/journal_read_html_article.php?j=JCSM/6/2/1, doi:10.13052/jcsm2245-1439.621. (cited on Page 26, 51, and 52)

[WZFH15] Steffen Wendzel, Sebastian Zander, Bernhard Fechner, and Christian Herdin. Pattern-Based Survey and Categorization of Network Covert Channel Techniques. *ACM Computing Surveys*, 47(3):50:1–50:26, April 2015. doi:10.1145/2684195. (cited on Page xiii, 3, 4, 8, 24, 38, 45, and 46)

[YBCP19] Xuhang Ying, Giuseppe Bernieri, Mauro Conti, and Radha Poovendran. TACAN: transmitter authentication through covert channels in controller area networks. In *Proceedings of the 10th ACM/IEEE International Conference on Cyber-Physical Systems*, ICCPS '19, pages 23–34, New York, NY, USA, April 2019. Association for Computing Machinery. doi:10.1145/3302509.3313783. (cited on Page 8)

[Zan17] S. Zander. Bro Covert Channel Detection (BroCCaDe) Framework: Scope and Background. 2017. URL: http://www.it.murdoch.edu.au/nsrg/cc_detection_ids/reports/Murdoch_University_IT_NSRG_TR20171117A.pdf. (cited on Page 27 and 164)