Assessment and enforcement of wireless sensor network-based SCADA systems security

Lyes Bayou

UNIVERSITE BRETAGNE LOIRE / MATHSTIC

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

THÈSE DE DOCTORAT DE

L'**É**COLE **N**ATIONALE **S**UPERIEURE **M**INES-**T**ELECOM
ATLANTIQUE **B**RETAGNE **P**AYS DE LA **L**OIRE - **I**MT **A**TLANTIQUE
COMUE **U**NIVERSITE **B**RETAGNE **L**OIRE

Ecole Doctorale N°601
Mathèmatique et Sciences et Technologies de l'Information et de la Communication
Spécialité : Informatique

Par

Lyes BAYOU

Assessment and enforcement of Wireless Sensor Networks-based SCADA systems security

Thèse présentée et soutenue à RENNES, le 19 juin 2018
Unité de recherche : LabSticc
Thèse N° : 2018IMTA0083

Rapporteurs avant soutenance :

Mme. Isabelle Chrisment	Professeur, TELECOM Nancy
M. Damien Sauveron	Maître de conférences, Université de Limoges

Composition du jury :

Président :	Mme. Ana Cavali	Professeur, TELECOM SudParis
Examinateurs :	M. Abdelmalek Benzekri	Professeur, IRIT, Toulouse
	Mme. Isabelle Chrisment	Professeur, TELECOM Nancy
	M. Damien Sauveron	Maître de conférences, Université de Limoges
Dir. de thèse :	Mme. Nora Cuppens	Directeur de Recherche, IMT-Atlantique, Rennes
Co-dir. de thèse :	M. Frédéric Cuppens	Professeur, IMT-Atlantique, Rennes
	M. David Espès	Maître de conférences, UBO, Brest

"Dedicated to
the memory of my parents,
my best friend and wife Meriem,
and my two lovely children Nihel and Achraf"

Abstract

Industrial Control Systems (ICS) are computer-based systems used for monitoring and managing industrial installations and facilities. We can find such systems in airports, power plants, gas refineries, etc. The architecture of these systems relies on several sensors and actuators deployed throughout the industrial installation. Sensors are responsible for gathering different kinds of information about the industrial process such as temperature, pressure, flow, etc. This information is sent to a controller that processes them and sends back commands to actuators. As results, an actuator can for example open a valve to increase the flow of a chemical component or stop a pump when the oil tank is filled.

The security in Industrial Control Systems is a major concern. Indeed, these systems manage installations that play an important economical role. Furthermore, targeting these systems can lead not only to economical losses but can also threaten human lives.

Therefore and as these systems depend on sensing data, it becomes obvious that additionally to real-time requirement, it is important to secure communication channels between these sensors and the main controllers. These issues are more challenging in Wireless Sensor Networks (WSN) as the use of wireless communications brings its own security weaknesses.

This thesis aims to address WSN-based security issues. Firstly, we conduct an in-deep security study of the WirelessHART protocol. This latter is the leading protocol for Wireless Industrial Sensor Networks (WISN) and is the first international approved standard. We assess its strengths and emphasize its weaknesses and limitations. In particular, we describe two harmful security vulnerabilities in the communication scheme of WirelessHART and propose improvement in order to mitigate them.

Secondly, we present wIDS, a multilayer specification-based Intrusion Detection System (IDS) specially tailored for Wireless Industrial Sensor Networks. The proposed IDS checks the compliance of each action performed by a wireless node based on a formal model of the expected normal behavior.

Résumé

Les systèmes de contrôle industriel (SCI) sont des systèmes informatisés utilisés pour la surveillance et la gestion d'installations industrielles. Nous pouvons trouver de tels systèmes dans les aéroports, les centrales électriques, les raffineries de gaz, etc. L'architecture de ces systèmes repose sur plusieurs capteurs et actionneurs déployés sur l'ensemble de l'installation industrielle. Les capteurs sont responsables de la collecte de différents types d'informations sur le processus industriel, telles que la température, la pression, le débit, etc. Ces informations sont envoyées à un contrôleur qui les traite et renvoie des commandes aux actionneurs. Ainsi, un actionneur peut par exemple ouvrir une vanne pour augmenter le débit d'un composant chimique ou arrêter une pompe lorsque le réservoir est rempli.

La sécurité dans les systèmes de contrôle industriel est une préoccupation majeure. En effet, ces systèmes gèrent des installations qui jouent un rôle économique important. En outre, attaquer ces systèmes peut non seulement entraîner des pertes économiques, mais aussi menacer des vies humaines.

Par conséquent, et comme ces systèmes dépendent des données collectées, il devient évident qu'en plus des exigences de temps réel, il est important de sécuriser les canaux de communication entre ces capteurs et les contrôleurs principaux. Ces problèmes sont plus difficiles à résoudre dans les réseaux de capteurs sans fil (WSN), car l'utilisation des communications sans fil entraîne ses propres faiblesses en matière de sécurité.

Cette thèse a pour but d'aborder les questions de sécurité des WSN. Tout d'abord, nous effectuons une étude de sécurité approfondie du protocole WirelessHART. Ce dernier est le protocole leader pour les réseaux de capteurs sans fil industriels (WISN). Nous évaluons ses forces et soulignons ses faiblesses et ses limites. En particulier, nous décrivons deux vulnérabilités de sécurité dangereuses dans son schéma de communication et proposons des améliorations afin d'y remédier.

Ensuite, nous présentons wIDS, un système de détection d'intrusion (IDS) multi-couches qui se fonde sur les spécifications, spécialement développé pour les réseaux

de capteurs sans fil industriels. L'IDS proposé vérifie la conformité de chaque action effectuée par un nœud sans fil sur la base d'un modèle formel du comportement normal attendu.

Remerciements

Je tiens tout d'abord à remercier ma directrice de thèse Nora Cuppens-Boulahia et mes co-encadrants Frédéric Cuppens et David Espès pour leur dévouement, leurs encouragements et leurs soutiens durant ces quatre années. J'ai énormément appris à leurs côtés, leurs conseils et orientations m'ont permis de progresser, de m'améliorer et de me dépasser. Cette aventure fût une expérience enrichissante tant sur le plan scientifique que sur le plan personnel.

Je remercie également les membres du Jury qui m'ont fait l'honneur d'évaluer et de valider mes travaux. Notamment, Isabelle Chrisment et Damien Sauveron pour avoir accepté d'être les rapporteurs de ce travail, dont les conseils et les remarques ont permis d'améliorer la version finale de ce manuscrit. Ainsi qu'Abdelmalek Benzekri et Ana Cavali, pour leur disponibilité et la pertinence de leurs remarques.

Je remercie aussi tous les membres de notre équipe: permanents, post-doctorants, doctorants et stagiaires pour la bonne ambiance de travail. J'ai passé d'agréables moments en votre compagnie notamment lors de nos discussions et nos pauses café, thé et autres.

Je remercie chaleureusement les membres de ma famille pour leurs encouragements. En particulier, je remercie *Meriem*, ma chère épouse pour son soutien indéfectible, ses conseils pertinents et surtout sa patience infinie.

Je tiens aussi à remercier tout le personnel de l'IMT-Atlantique, en particulier ceux du campus de Rennes, pour avoir mis à ma disposition tous les moyens nécessaires pour mener à bien ce travail.

Merci à vous tous, ce travail n'aurait pas pû être mené à terme sans votre participation.

Contents

CHAPTER 1

Introduction

In this chapter we introduce the context of this thesis followed by the motivation and background of this studies. Then we present our main contributions and we conclude by the structure of this manuscript.

1.1 Context and Motivation

In industrial environments, Supervisory Control and Data Acquisition systems (SCADA) are used for monitoring and managing complex installations such as power plants, refineries, railways, etc. These systems rely on sensors deployed over large area to gather in real time information about the industrial process. These informations are sent to a controller that processes them and sent back commands to field devices such as actuators or valves.

Security is an important issue in SCADA systems. Indeed, the disruption of these systems can cause significant damages to critical infrastructures such as electric power distribution, oil and natural gas distribution, water and waste-water treatment, and transportation systems. This can have a serious impact on public health, safety and can lead to large economical losses [Huang et al. 2009].

However, for a long time ICS systems have been considered as secured systems until several incidents and cyberattacks come to illustrate their vulnerability, especially towards highly motivated with deep knowledges attackers. Thus, in 2000, a former contractor hacked the communication network of a sewage plant in Queensland/Australia[Slay and Miller 2007]. As result, nearby areas including a hotel, a park and a parking, were flooded by one million liters of untreated water. In 2008, Stuxnet [Falliere et al. 2011], the first cyber-weapon, infected the Iranian nuclear facilities of Natanz in probably a state-leaded cyberattack. In 2015, a cyberattack hit the Ukrainian electric system. Up to 225,000 customers were affected by a blackout that lasted about 7 hours [ICS CERT 2016].

Managing security threats targeting these systems is a problem of vital importance for the company's long-range strategy. An attack can have either an outside or an inside origin. An outside attacker does not have any knowledge about secrets (passwords, keys, etc.) used to protect the network. The outside attacker can have several profiles. He can be a State targeting a strategic facilities, a terrorist group, a hacktivist with political or ideological motivations like Anonymous or Greenpeace, or cybercriminals wanting to make profits using ransomwares or botnets.

Therefore and as these systems depend on sensing data, it becomes obvious that additionally to real-time requirement, it is important to secure communication channels between these sensors and the main controllers. These issues are more challenging in Wireless Sensor Networks (WSN) as the use of wireless communications brings its own security weaknesses.

Indeed, WSN are subject to the same attacks as other wireless networks. Mainly, attackers use wireless communication as a vector to launch their attacks. Furthermore, sensor's limited capabilities in terms of processing power, memory space and energy make it hard the implementation of strong security mechanisms.

1.2 Objectives and Contributions

This thesis aims to address WSN-based security issues in an industrial context. To do that, our First contribution is a review of the general security context of Industrial Control Systems. We describe within it the main characteristics of these systems and emphases their important role in managing vital economical and national facilities. We detailed the different attackers profiles and their motivations. We also list some significant cyberattacks involving ICS in order to illustrate their evolution and the continuous improvement of attack vectors.

In our Second contribution we analysis the security of Wireless Sensor Network protocol used in industrial systems. Therefore, we conduct and in-deep assessment of the WirelessHART protocol, the leading communication protocol for WISN. We give a detailed description of its security mechanisms. We show how these mechanisms are used along with other non-security mechanisms to ensure security requirements. Then, we assess their strengths and emphasize their weaknesses and limitations.

Our Third contribution is the implementation of WirelessHART NetSim, a simulator dedicated to the security study of WirelessHART networks. The proposed simulator fully implements the WirelessHART protocol stack and its different kind of devices. It includes all important services including routing and communication scheduling al-

gorithms. Our simulator includes scenarios for testing several kinds of attacks and can be easily extended in order to test additional ones. This work is published in [Bayou et al. 2015b].

Our Forth contribution is the description of two security issues targeting the WirelessHART protocol. These two issues result mainly from the use of shared cryptographic keys for securing communications and allows an insider attacker using only its own credential, to bypass security implemented mechanisms. The first one allows an insider attacker to conduct a sybil attack that can lead to isolate partially or totally parts of the wireless network. The second one allows an insider attacker to inject false commands into the network. These works are published in [Bayou et al. 2015a] [Bayou et al. 2016b].

The Fifth contribution is wIDS a multilayer specification-based intrusion detection system for Wireless-based SCADA Systems. We describe its architecture and deployment scheme. The proposed IDS checks the compliance of each action performed by a wireless node towards the formal model of the expected normal behavior. For this purpose, we propose wirelessOrBAC a formalisms that uses a control access model to express in a comprehensive and easy way the security requirements of WSN. It also permits an accurate description of WSN inherent constraints and limitations and emphases their roles in security enforcement. Access control rules are used to build the expected behavior of wireless nodes. Then, this model is used to monitor wireless nodes actions in order to perform intrusion detection tasks. Also, in addition to alerts that are raised by actions deviating from the normal model, we define additional intrusion rules that aim to detect basic attacker actions such as injecting, deleting, modifying and delaying packets. These works are published in [Bayou et al. 2017b][Bayou et al. 2017a]

The Sixth contribution is a deployment scheme for the placement of the IDS-agent of a decentralized IDS in a Wireless Industrial Sensor Network. It presents the best trade-off between the number of used IDS-agents and the detection efficiency. We use the graph theory concept of *Dominating Set* to select nodes that will be substituted by super-nodes. Super-nodes have enhanced storage and processing capacities that allow them to act in the same way as normal sensors and also as detection agents. By this way, a virtual wireless backbone network providing intrusion detection capabilities will be created upon the WSN. This work is published in [Bayou et al. 2016a]

1.3 Organization of the dissertation

This dissertation is organized as described below.

Chapter 2 – A Landscape of vulnerabilities and threats targeting Industrial Control Systems – we provide in this Chapter, a large overview of the security context of ICS systems. Thus, we detailed the characteristics of this systems and provide a description of their vulnerabilities and the threats targeting them.

Chapter 3 – Security analysis of WSN-based SCADA systems: Case of the WirelessHART protocol– proposes an in-deep analysis of the WirelessHART protocol, the leading communication protocol for Wireless Industrial Sensor Networks (WISN). Then, we assess the strengths of its security mechanisms and emphasize their weaknesses and limitations.

Chapter 4 – WirelessHART security issues – describes two vulnerabilities targeting WirelessHART networks. The sybil attack and the broadcast attack. It demonstrate their potential harmful impact on a WSN. We also provides some improvements and countermeasures to mitigate them.

Chapter 5 – wIDS: a multilayer IDS for Wireless-based SCADA Systems – describes a specification-based IDS for WISN. The proposed IDS checks the compliance of each action performed by a wireless node towards the formal model of the expected normal behavior.

Chapter 6 – Towards a CDS-Based Intrusion Detection Deployment Scheme for Securing Industrial Wireless Sensor Networks – defines a deployment scheme for the placement of the IDS-agent of a decentralized IDS in a Wireless Industrial Sensor Network.

Chapter 7 – Conclusions and Perspectives – this Chapter concludes the dissertation by summarizing the contributions and presenting the perspectives for future work.

CHAPTER 2

A Landscape of vulnerabilities and threats targeting Industrial Control Systems

2.1 Introduction

During past decades, several experts repeatedly had warn that computer-based attacks can cause physical damages to ICS systems [Pietre-Cambacedes et al. 2011][Lemay and Fernandez 2013]. Thus, the US Department of Energy (DoE) Idaho National Laboratory (INL) conducted an experimental research, named Aurora, on the replica of a process control system for electrical power generator [Lemay and Fernandez 2013]. As a result, a computer network attack launched against this system, caused a violent physical destruction of the electrical power generator.

However, ICS systems have continued to be considered as secured systems until several incidents and cyberattacks come to illustrate their vulnerabilities, especially towards highly motivated with deep knowledges attackers.

These cyberattacks convinced experts that [Lemay and Fernandez 2013]:

- Attacking an industrial infrastructure does not require a physical access to it;
- An attack can be launched from third-party networks such as the corporate or the maintenance networks;

- Damages resulting from a cyberattack against an industrial infrastructure are not circumscribed to its communication network, but can also be as harmful than an incident or a physical attack.

Indeed, cyberattacks targeting ICS systems can have various consequences. They can lead to information system compromising, infrastructure control takeover, component physical destruction, and sensitive information disclosure.

The aim of our study is to give on one sight a large landscape of vulnerabilities and threats targeting Industrial Control Systems. We present why SCADA systems are important, who is targeting these systems and what are their motivations. We also present an overview of methods used in SCADA attacks and show the consequences of such attacks. We also describe some relevant attacks that targeted SCADA systems.

The rest of the Chapter is organized as follows. In Section 2.2, we introduce ICS and detailed their characteristics. We detail in Section 2.3 ICS security vulnerabilities and threats. In Section 2.4 we describe some significant attacks that have targeted industrial control systems. A taxonomy of these attacks are presented in Section 2.5. In Section 2.6, we present main research axes on the security of ICS. Finally, Section 2.7 concludes this Chapter.

2.2 Background on SCADA systems

2.2.1 Definitions

Industrial control system (ICS) is a general term that encompasses several types of control systems, including supervisory control and data acquisition (SCADA) systems, distributed control systems (DCS), and other control system configurations such as Programmable Logic Controllers (PLC) often found in the industrial sectors and critical infrastructures [Stouffer et al. 2015] [National Communications System 2004].

We give below, the description of each type of ICS:

Supervisory Control and Sata Acquisition (SCADA) systems

they are used to control dispersed assets where centralized data acquisition is as important as control. These systems are used in distribution systems such as water distribution and wastewater collection systems, oil and natural gas pipelines, electrical

utility transmission and distribution systems, and rail and other public transportation systems.

Distributed Control Systems (DCS)

They are used to control production systems within the same geographic location for industries such as oil refineries, water and wastewater treatment, electric power generation plants, chemical manufacturing plants, automotive production, and pharmaceutical processing facsilities. These systems are usually process control or discrete part control systems.

Programmable Logic Controllers (PLC)

They are used in both SCADA and DCS systems as the control components of an overall hierarchical system to provide local management of processes through feedback control as described in the sections above. In the case of SCADA systems, they may provide the same functionality of Remote Terminal Unites (RTUs). When used in DCS, PLCs are implemented as local controllers within a supervisory control scheme.

In addition to PLC usage in SCADA and DCS, PLCs are also implemented as the primary controller in smaller control system configurations to provide operational control of discrete processes such as automobile assembly lines and power plant soot blower controls

2.2.2 Architecture

Industrial Control Systems (ICS) are used for gathering data in real-time from remote locations in order to control and monitor industrial processes, including data aggregation and presentation to operators. A typical SCADA system shown in Fig. 2.1, consists of Field Station, Control Station and Communication Network. A description of these components are given hereafter [Stouffer et al. 2015] [National Communications System 2004]:

- **Supervisory computer or Master Terminal Unit (MTU)**: It refers to the device that acts as the master in a SCADA system. It is responsible for communicating with the field connection controllers, which are RTUs and PLCs.
- **Remote Terminal Units (RTU)**: They are connected both to sensors and actuators in the process and to the supervisory computer. They collect data from

the field sensors, make the necessary adjustments and transmit the data to the monitoring and control system.

- **Programmable Logic Controllers (PLCs)**: They are used as an alternative to RTUs. PLCs have advanced capabilities allowing them controlling complex processes.
- **Field devices**: They are sensors and actuators deployed throughout the industrial installation. They are used for acquiring or controlling the process in real-time.
- **Human machine interface (HMI)**: It is a piece of software and hardware that allows human operators to monitor the state of a process under control, modify control settings to change the control objective, and manually override automatic control operations in the event of an emergency.
- **Data historian**: It is a centralized database for logging all process information that could be used for various analyses, from statistical process control to enterprise level planning.
- **Communication infrastructure**: It is used for connecting the supervisory control level to lower-level control modules

We should notice that the architecture of SCADA systems evolved throughout the time according to industrials needs and the technological developments [National Communications System 2004]:

- **Monolithic SCADA Systems**: They use 'mainframe' systems. Networks were generally non-existent, and each centralized system stood alone and did not include connectivity features. They use proprietary communication protocols.
- **Distributed SCADA Systems**: The next generation of SCADA systems took advantage of developments and improvement in system miniaturization and Local Area Networking (LAN) technology. It used mini-computer stations which were smaller and less expensive than their first generation mainframe. The control system were distributed across multiple stations connected through a LAN. Each station was dedicated for the monitoring of a particular task. The used communication protocols were still proprietary and not standard. The communication's security was not enforced.
- **Networked SCADA Systems**: The major improvement in the third generation is that of opening the system architecture, utilizing open standards and protocols

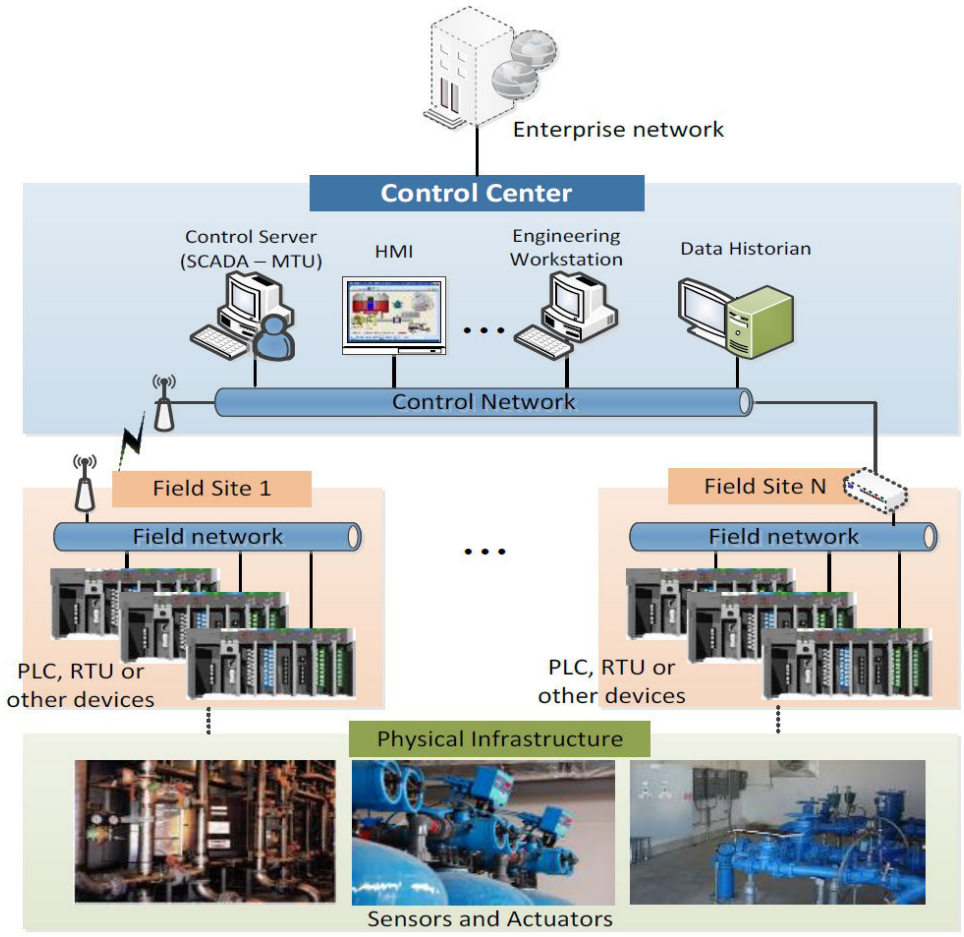


Figure 2.1: ICS Architecture

and making it possible to distribute SCADA functionality across a Wide Area Network (WAN) and not just a LAN. Consequently, these systems are composed of several distributed SCADA spread over large geographical areas.

Furthermore, the utilization of off-the-shelf systems makes it easier for the user to connect third party peripheral devices (such as monitors, printers, disk drives, tape drives, etc.) to the system.

- **Fourth Generation Internet of Things**: Industry 4.0 also called the 'Smart factory' refers to the increase use and the integration of the Internet of Things (IoT) technology in the industrial control systems [Waidner and Kasper 2016]. It results from the convergence of the Information Technology (IT) and the Operational Technology (OT) across the whole manufacturing supply chain. Thus, through the Internet of Things, cyber-physical systems communicate and cooperate with each other and with humans in real time, and via cloud computing.

The ultimate aim is to achieve a more flexible architecture in order to increase the sensing efficiency and productivity [Lee et al. 2014]

2.2.3 ICS vs IT systems

Historically, ICS had widely differed from IT systems. Thus, ICS were isolated systems running proprietary protocols using dedicated hardware and software. However, as these systems have started adopting IT solutions in order to enhance corporate connectivity and remote access capabilities, they are starting to resemble more and more to IT systems.

Nevertheless, ICS still have many characteristics that differ from traditional IT systems, including different risks and priorities [Stouffer et al. 2015] [Zhu et al. 2011]. These differences are summarized in Table 2.1.

The most significant ones are that ICS operate continuously with little down time, they are designed to meet high performances in terms of reliability and safety, and they are expected to work for 10 or 15 years long.

Furthermore, ICS have different performance and reliability requirements, and also use operating systems and applications that may be considered unconventional in a typical IT network environment [Lemay and Fernandez 2013].

For these reasons traditional security mechanisms used in IT must be adapted before deploying them in SCADA systems.

2.2.4 Industrial Communication Protocols

Typical communications in an industrial network are composed of control messages exchanged between master and slave devices. A master device such as a PLC, is in charge of the operation control of another device. A slave device is usually a sensor or actuator which executes the master commands and can periodically send report messages.

Formerly, there were more than 200 different industrial communication protocols [Igure et al. 2006]. Most of them were proprietary standards developed by individual companies. Nowadays, these latter have moved to use open standard protocols mainly IP-based. Among these protocols, the most used ones are: ModbusTCP, EtherNetIP, IEC 61850, ICCP, OPCUA and DNP3. For wireless-based networks we can also mention: ZigBee Pro, WirelessHART and ISA 100.11a.

Categories	IT systems	ICS
Performances	Non-real time	Real-time
Availability	Availability deficiencies generally tolerated	High availability
Risk	Manage Data, Confidentiality and integrity are paramount. Delay of business operation	Control physical world. Safety is paramount. Loss of life, equipment or production is at risk
System operation	Generic OS	Proprietary OS often without security capabilities
Resources	Have enough resources	Have constrained resources memory space and computing
Communication	Standard communication protocols	Proprietary protocols
Updates & Changes	Applied periodically, generally it is automated	Hard to apply patches or updates, tested carefully before applied, planned and scheduled in advance
Component life time	On the order of 3 to 5 years	On the order of 10 to 15 years
Components location	Local and easy to access	Isolated, remote and require efforts to access them

Table 2.1: IT vs ICS differences summary

Unfortunately, many of these protocols, in particular those used in wired networks, do not implement any security mechanisms. Thus, generally, they do not include any authentication to remotely execute commands on a control device.

2.2.5 Critical Infrastructures

As indicated in Section 2.2, SCADA systems are used for monitoring industrial facilities. Critical infrastructures (CI) are a part of these installations and facilities that provide the essential goods and services forming the backbone of the society and its way of life [Auerswald et al. 2005] [France 2013]. These infrastructures have to be protected in order to ensure their safety, availability and continuity both from physical and cyberattacks.

Several definitions have been issued to define which facilities are considered as critical infrastructures. Thus, according to the European Union (EU) Directive, issued in 2008 [The Council of the European Union 2008]: "*Critical infrastructures* means an asset, system or part thereof located in Member States which is essential for the maintenance of vital societal functions, health, safety, security, economic or social well-being of people, and the disruption or destruction of which would have a significant impact in a Member State as a result of the failure to maintain those functions".

The US Presidential Executive Order 13636, issued in 2013, defines Critical infrastructures as "systems and assets, whether physical or virtual, so vital to the United States that the incapacity or destruction of such systems and assets would have a debilitating impact on security, national economic security, national public health or safety, or any combination of those matters" [Marsh 1997] [The National Institute of Standards and Technology 2013].

Although these definitions slightly differ, all critical infrastructures definitions mainly includes the following areas:

- Energy: Electrical power, oil, gas;
- Sanitation: Water supply, waste water collection and processing;
- Transportation: Roads, railway, traffic organisation, civil/military aviation;
- Communications: Information technology infrastructure, telecommunications, Internet access;
- Security and Safety: Military, police, emergency services;
- Medicine: Health-care, hospitals;
- Research: Industrial and scientific developments;
- Finances: State treasury, banks, money wire transfers;
- Politics: National secrets, foreign policy and affairs.

Furthermore, as indicated in Figure 2.2, Critical infrastructures are highly interconnected and mutually dependent throughout physical and virtual (shared information and communications technologies) links [Rinaldi et al. 2001]. These interdependencies increase the impact of an infrastructure failure. Thus, an incident in one part of the infrastructure may spread out through the system and have cascading effects on other sectors and impact large geographic regions [Rinaldi et al. 2001].

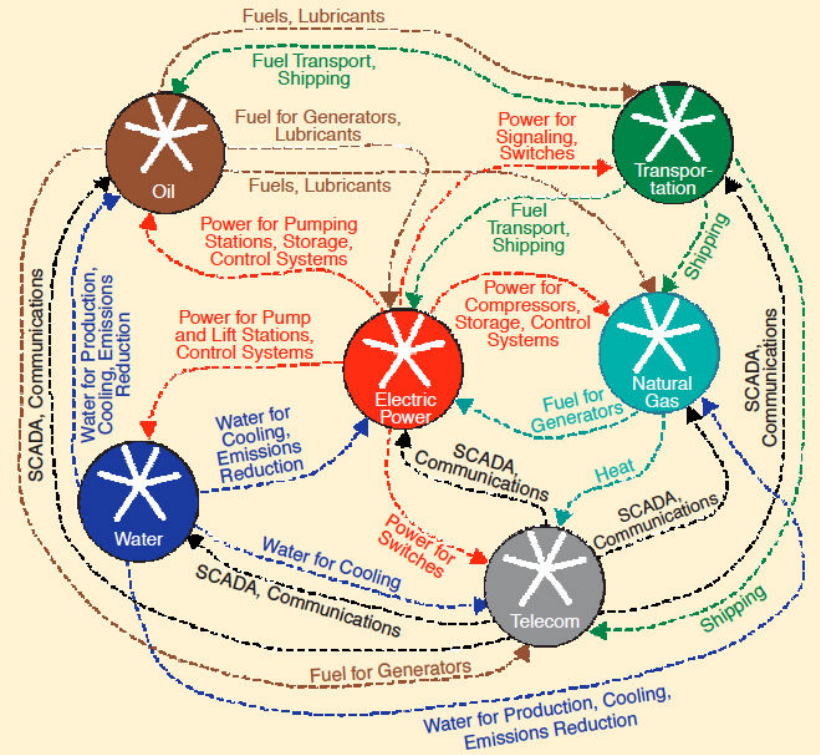


Figure 2.2: Critical infrastructures interdependencies [Rinaldi et al. 2001]

2.3 SCADA systems security issues

For a long time, attacks against SCADA systems seemed to be part of science-fiction. Indeed, SCADA systems were considered as secured networks. Thus, widely shared beliefs were [Pietre-Cambacedes et al. 2011]: that nobody wants to attack these systems; they are isolated from external networks; they use obscure protocols only known by experts; and the embedded security mechanisms such as cryptography ensure a high security level.

However, these last decades, SCADA systems have been facing security challenges they were not initially designed to deal with [Anton et al. 2017]. This situation is mainly due to the following technological and architectural evolutions [Stouffer et al. 2015][Igure et al. 2006]:

- *The increase networks interconnectivity*: SCADA networks were originally located on separated and stand-alone networks. Nowadays, there are great needs to increase interconnections between the factory floor and the corporate network in order to improve efficiency and productivity. However, this interconnectivity adds multiple access points to the SCADA networks that are not anymore isolated. This could be exploited by an attacker to gain access to the factory floor devices.

- *The move from proprietary standards for SCADA communication protocols towards open international standards*. However, most legacy SCADA protocols lack security mechanisms as they were never intended for the use on publicly accessible networks and in some cases not even on IP networks. Furthermore, the open standards make it very easy for attackers to gain in-depth knowledge about the working of these SCADA networks.

- *The use of commercial off-the-shelf (COTS) devices and technologies*: due to their cost saving and design time reducing, COTS hardware and software are heavily used to develop devices for operating in the SCADA network. However these devices and software are generic components that are not designed to meet specific industrial security requirements.

2.3.1 ICS vulnerabilities

A vulnerability is defined as a weakness of a hardware or a software, that can be exploited by one or more threats. Since 1997 and the disclosure of the first vulnerabilities in ICS, the number of discovered vulnerabilities in ICS components has significantly increased [Anton et al. 2017][Byres et al. 2004]. Thus, as shown in Figure 2.3, it has passed from 19 vulnerabilities in 2010 to 69 in 2011 (probably due to the Stuxnet attack), to 189 in 2015 [Kaspersky Lab ICS CERT Threat 2016] and 322 in 2017 [Kaspersky Lab ICS CERT Threat 2017] [Kaspersky Lab ICS CERT Threat 2018].

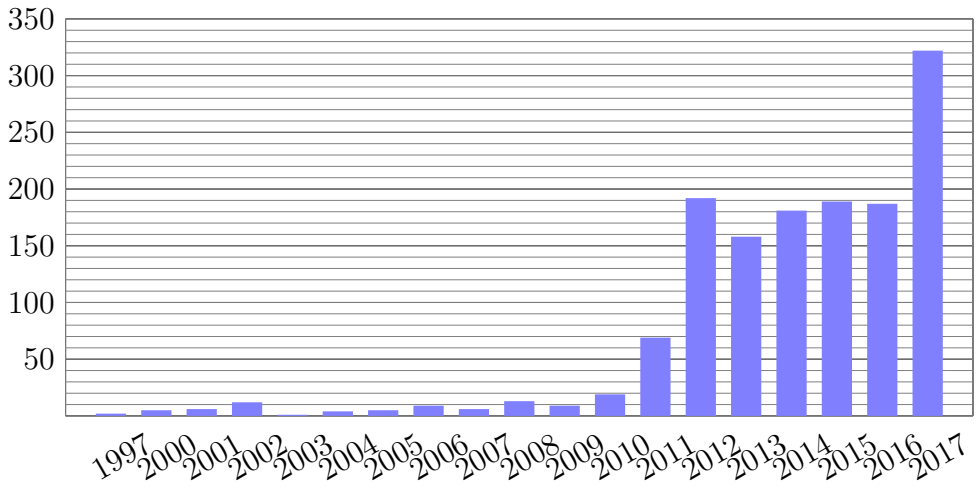


Figure 2.3: ICS discovered vulnerabilities by year

Furthermore, not all of the discovered vulnerabilities are fixed. Thus for the 189 vulnerabilities discovered during 2015 [Kaspersky Lab ICS CERT Threat 2016] :

- Exploits are available for 26 of these vulnerabilities.
- Most of them are critical (49%) or have a medium severity (42%).
- Vulnerabilities were discovered in different components of different manufacturers.
- 15% of these vulnerabilities stay not fixed or only partially fixed (in 2016).
- 14 vulnerabilities among the 19 unpatched ones are of high level risk.

For SCADA systems the most widespread issues are [Kaspersky Lab ICS CERT Threat 2016] [Kaspersky Lab ICS CERT Threat 2018]:

1. Cross-site scripting (7 vulnerabilities);
2. Buffer overflows (5 vulnerabilities);
3. Cross-site request forgery (4 vulnerabilities);
4. Unrestricted file upload (3 vulnerabilities);
5. SQL injection (3 vulnerabilities);
6. Hard-coded credentials.

In [Anton et al. 2017], authors analyzed available exploits for SCADA systems. They found about 100 metasploit modules and Proofs of Concepts (PoC) exploits specially tailored for targeting PLC. These exploits are published in several specialized databases and are publicly available. This makes it easy for anybody to exploit them without much difficulties.

However, we must notice that compared to the 100.000 IT-based attacks, the number of SCADA exploits are relatively small, with only 373 entries [Anton et al. 2017].

2.3.2 Cyberattack's Risks and Impact

Since last decades, attacks targeting ICS are not only keeping increasing but also are changing and evolving [Anton et al. 2017]. Indeed, until 2000, almost 70% of the reported incidents were either due to accidents or due to disgruntled employees [Byres et al. 2004]. Since 2001, in addition to the continuous increase of attacks number, reports also indicate that almost 70% of the incidents were due to attacks originating from outside the SCADA network [Igure et al. 2006][Anton et al. 2017].

Because of their cyberphysical aspect, the impact of a cyber incident in an ICS may include both digital and physical effects. Thus, attacks on industrial control systems can lead to several consequences as follows [Stouffer et al. 2015] :

- Personal injury and loss of life;
- Installation damage;
- Loss of integrity or reliability of process data and production information;
- Installation unavailability;
- Process upset leading to compromised process functionality, inferior product quality, lost production capacity, compromised process safety;
- Environmental releases or damages;
- Unauthorized access, theft, or misuse of credential information;
- Disclosure of sensitive information to unauthorized destinations;
- Violation of legal or regulatory requirements;
- Risk to public health and confidence;
- Threat to a nation's security.

On the other hand, in the case of companies, a cyberattack can have specific consequences such as:

- Brand and reputation damage;
- Financial implications;
- Share value losses;
- Customers or employees loss of life or injury.

2.4 Significant attacks against SCADA systems

While many ICS attacks have been publicized, many more have not been disclosed and even those that are made public are not clearly understood.

In this Section, we describe some relevant attacks against SCADA systems either in term of consequences or in term of method [Miller and Rowe 2012].

2.4.1 The Maroochy breach

In 2000, an attack targeted the communication network of a sewage plant at the Maroochy Water Services in Queensland/Australia [Slay and Miller 2007]. Consequently, the plant's staff noticed that communications sent by radio links to wastewater pumping stations were being lost, pumps were not working properly, and alarms put in place to alert the staff about troubles were not going off. As a result, over a three-month period, nearby areas including a hotel, a parc and a parking, were flooded by one million liters of untreated water.

At the begining of the incident, the plant's staff thought that they were facing technical problems with the new installed system. However, after deep investigation and monitoring every signal passing through the system, they discovered that someone had hacked the system and deliberately caused the problems.

The attacker was identified as Vitek Boden, a former contractor. He was motivated by revenge after he failed to secure a job with the Maroochy Water Services.

2.4.2 Conficker

Conficker (also known as Downadup) [F-Secure Labs 2008] is a computer worm targeting the Microsoft Windows operating system that was first detected in November 2008. It was mainly designed for gathering login information and financial data. It propagated through the Internet by exploiting a vulnerability in a network service (MS08-067) [Microsoft 2008]. It includes capabilities to spread through networks and removable data drives and was able to perform dictionary attacks on administrator passwords.

The Conficker worm infected millions of computers around the world. And although it was not specifically designed for targeting industrial facilities, several cases of ICS infections were reported. Thus, it infected computers in a German nuclear plant without posing a serious threat to the installation [Dine et al. 2016]. In the same way, the French Navy computer network, was also infected with Conficker on January 15, 2009. Consequently, the network was quarantined, forcing aircrafts at several airbases to be grounded as their flight plans could not be downloaded [The Telegraph 2009].

2.4.3 Stuxnet

Stuxnet is a worm specially tailored to infect and damage industrial installations. It was discovered in July 2010 and supposed to have existed since one or two years earlier [Falliere et al. 2011].

Technically, it is a large (500kB) complex piece of malware with many different components and functionalities [Falliere et al. 2011]. It capabilities includes the ability to spread in LAN using a Windows vulnerabilities (CVE-2008-4250(MS-08-067)) that was already used before by Conficker or a zero-day vulnerability in the Windows Print Spooler Service Vulnerability (CVE-2010-2729(MS-10-061)). It can also spread by infecting USB Flash memory using another zero-day vulnerability (CVE-2010-2568(MS-10-046)). Moreover, it includes capabilities to attack industrial control systems (ICS), in particular to infect Siemens Step 7 files (Siemens' products configuration language), modify programmable logic controllers (PLCs), and hide its presence on an ICS network [Falliere et al. 2011, Langner 2011].

In addition, Stuxnet uses a driver digitally signed with a compromised Realtek certificate. A different version of the driver was also found signed by a different compromised digital certificate from JMicron [Falliere et al. 2011].

For several experts [Falliere et al. 2011, Langner 2011], Stuxnet is considered as the first used cyber-weapon. It was probably part of a state-leaded cyber attack targeting the Iranian nuclear installations as more than 60% of infected hosts were in this country [Falliere et al. 2011]. Furthermore, the Natanz uranium enrichment facility is considered as the probably primary target of this cyberattack [Falliere et al. 2011, Langner 2011].

Indeed, Stuxnet was designed to destroy uranium enrichment centrifuges by abruptly speeding them up and slowing them down. Its ultimate goal was to sabotage that facility by reprogramming programmable logic controllers (PLCs) to operate as the attackers intend them to, most likely out of their specified boundaries.

2.4.4 Ukrainian blackout

In December 2015, a cyberattack hit the Ukrainian electric system two days before Christmas. The incident had cut electricity to nearly a quarter-million customers and lasted about 7 hours [ICS CERT 2016, Lee et al. 2016]. This attack is one of multiple victims of the BlackEnergy campaign [Cherepanov 2016].

In this attack, intruders get access to the business networks of a regional electricity distribution company through spearphising emails with a malicious MS Office documents (i.e., Word and Excel) attachments. Opening the document and enabling the macros installs the BlackEnergy 3 malware on the victim system [GReAT 2016]. Once installed, the malware connects to the Command & Control (C&C) server and starts to install several malicious tools and moves into the network to infect other devices. The attackers appear to have executed these steps more than six months prior to the power outage [ICS CERT 2016].

Then using keystroke loggers, they perform credential theft that allows them to exploit a VPN connection to enter the ICS network and to access the SCADA dispatch workstations and servers. At this stage, the intruders used the HMIs in the SCADA environment to issue legitimate commands that caused the power outage [GReAT 2016, ICS CERT 2016]. Moreover, they uploaded a malicious firmware to the serial-to-ethernet gateway devices to ensure that operators will be unable to issue remote commands to bring the installations back online. The attackers aimed by conducting this operation two days before Christmas to provoke a fear sentiment and chaos among the targeted population.

The malware used in these attacks belongs to the BlackEnergy malware family [ICS CERT 2016]. Originally designed for Distributed Denial of Service (DDoS) attacks, BlackEnergy evolved into a plug-in based architecture easing the development of new attack-specific modules for espionage, DDoS, spam and fraud. It has been involved in several major cyberattacks including coordinated DDoS attack on Georgia's finance, military and government agencies, fraudulent bank transactions and the Ukrainian power grid [Tarakanov 2010, Khan et al. 2016].

2.4.5 Mirai against IoT

On October 21, 2016, a cyberattack hit the US Domain Name System (DNS) provider Dyn [Hilton 2016]. The attack involved multiple Denial of Service attacks (DDoS attacks) which caused major Internet platforms and services such as Amazon, Facebook, Twitter, etc., to be partially unreachable from Europe and North America [Hilton 2016, Nixon et al. 2016]. Dyn estimated that the attack had involved 100,000 malicious endpoints [Hilton 2016].

This attack was part of a series of previous ones including an attack on September 20, 2016 [Schneier 2016] on a computer security web site and an attack on the French web host OVH [OVH 2016]. These attacks were the largest known DDoS attacks to date [Hallman et al. 2017] reaching 1.2 Tbps and 1 Tbps respectfully.

These attacks were perpetrated using the Mirai malware that has been active since at least August 2016 [Hilton 2016, Nixon et al. 2016]. Mirai, that means "future" in Japanese [Hallman et al. 2017], is designed to brute-force the security of IoT devices in order to infect and remotely control them. It first scans the Internet looking for unsecured IoT devices. Then, it identifies vulnerable IoT devices including networked cameras, digital video recorders (DVR), and home routers. It gains access on these devices and infect them through the use of a table of more than 60 common factory default usernames and passwords. Infected devices will continue to function normally and could be used further as part of on-demand DDoS attacks. It is estimated that Mirai malware has infected more than 500,000 devices in 164 countries [Hallman et al. 2017].

Moreover, since the Mirai's source code was published in hacker forums as open-source, it has been adapted to other malware projects [Nixon et al. 2016] [SecurityWeek.Com 2016].

2.5 Attacks taxonomy

2.5.1 Attacks classification

We extend the attacks classification defined for power systems [Amin 2002], to all kind of Critical Infrastructures (CI). It includes the three following different kinds of threats:

- *Attacks upon the CI*: in this kind of attacks, the installation is the main target. The attacker aims to stop the production or at least significantly perturbs it. Stuxnet is the best example of this kind of attack. Indeed, the attackers' objective was to destroy Iranian uranium enrichment centrifuges.
- *Attacks by the CI*: here the attacker uses the installation as a *weapon* to target civilians. This can be for example the case of a nuclear plant that enters in an unsafe state due to a cyberattack. It can release radioactive elements in the air or pollute a nearby river that crosses a town. In the case of the Ukrainian blackout, the aim of the attacker was to widespread fear among the population. This was also the case with the Mirai malware where several thousand of cameras were hacked and used to launch a DDoS attack.
- *Attacks through the CI*: in this case, disturbing the industrial installation is part of a large plan against a third target. Thus, the attacker can before attacking an airport, an hospital or a bank, start by targeting the power system in order to cause a power outage.

2.5.2 Attackers profiles classification

Industrial systems can be targeted by attackers with several kinds of profiles. Each of them has its own characteristics and motivations. Hereafter, we describe the most relevant ones [Nicholson et al. 2012, Rocchetto and Tippenhauer 2016]:

State and governmental agencies

This class includes attackers that are belonging or are sponsored by a state generally through dedicated agencies. They are in charge of carrying out both defensive and offensive cyber operations. Their main targets are public infrastructure systems, power or water systems, banks, and governmental institutions. This kind of attackers is viewed as the most powerful profile characterized by high offensive skills and tools, high resources and determination. We should also note that generally they give a great importance to the stealthiness of their attacks.

The Stuxnet worm is considered as the first ICS cyberattack probably conducted by a nation [Falliere et al. 2011]. Attackers involved in the Ukrainian blackout also belong to this profile. Indeed, conducted investigations have shown that during this attack, the attackers have demonstrated their ability to conduct highly synchronized, multistage, multisite cyberattacks [ICS CERT 2016].

Terrorists groups or cyberterrorist

They are attackers with political motivations. Attacks are parts of their propaganda and are launched in order to cause severe disruption or widespread fear. The attacks mainly target the physical availability of the system. By their actions, they aim to gain a large coverage from the media. Currently, they are considered as having low offensive skills and average resources and not possessing sophisticated cyber capabilities [Dine et al. 2016].

Hacktivists or Activist hackers

They are attackers that aim to promote their political or social cause such as freedom of information, net neutrality, animals protection or denuclearization. Groups like "Greenpeace" or "Anonymous" can move from conducting physical protest to virtual ones by launching cyberattacks against nuclear facilities. Although the fact that these

groups possess a lower financial support, they have a higher manpower support which can allow them for example to launch large DDoS attacks.

Criminal organization

This category includes attackers that use their knowledge of known vulnerabilities to launch attacks against valuable systems. They can also use their high skills to figure out new zero-day vulnerabilities in order to sell them on the blackmarket. On the other hand, criminal organizations can hire skilled hackers or purchase malicious infrastructures such as botnets to launch cyberattacks.

This kind of attackers aims to earn money from blackmailing, ransomware or sabotage campaigns. They have advanced knowledge of IT network attacks but have limited knowledge of industrial standards. They use advanced tools and have average financial resources.

Disgruntled employees/inside attackers

This kind of attacker is the most common profile of inside attacker. Other profiles are composed of social engineering victims. It is the only profile which has an advance knowledge of the system because it has physical access to it. Generally, an inside attacker acts alone, with low budgets but with dedicated tools. His aim is to target the system availability. Insiders pose a serious threat considering their knowledge of the system and the implemented security procedures. In the majority of the case, the inside attack is only detected once the system becomes unavailable [Keeney et al. 2005].

According to a study on insider threat profiles, insiders' actions are generally triggered by a negative work-related event [Keeney et al. 2005], These events included: employment termination (47%), dispute with a current or former employer (20%), and employment related demotion or transfer (13%). As indicated in the case of the Maroochy breach, revenge is the most frequently reported motive of an inside attack [Keeney et al. 2005].

Hobbyists and Script kiddies

This category is composed mainly of passionate people looking for challenges, fun and discovery. They mainly rely on ready to use and automated tools to attack a system. They have average access to hardware, software, and Internet connectivity. Although,

attacks from this type of profile are believed to be very frequent in IT, it may not be the case for ICS as these systems require advanced skills.

However, several tools that intend to automate attacks against ICS are now freely available. Thus, AutoSploit [AutoSploit 2018] for example, attempts to automate the exploitation of remote hosts. AutoSploit combines together several different tools and workflows for hackers into one package [Joseph 2018]. It uses Shodan [Shodan 2018], an internet-connected devices search engine, and Metasploit [Metasploit 2018], a well-known penetration testing tool for executing exploits. This kind of tools makes it easy for people with low skills to scan the Internet in order to find available industrial systems, to check whether the target is vulnerable to implemented exploits and then to successfully launch attacks against the target system.

2.5.3 Attackers motivation

Attacks targeting ICS can have several motivations such as:

- **Money**: Attackers can aim to earn money either by threating or conducting attacks against industrial installations. This can be done by threating companies to launch DDoS attacks, or by using Blackmailing or ransomware campaigns.
- **Spying/Economical intelligence**: In this kind of attacks, perpetrators intended to obtain sensitive documents, designs and schemas for manufacturing, or extract confidential information about customers.
- **Defending Political/Ideological causes**: Attackers can launch actions against an industrial installation to put on the spotlight their cause. It could also aim to bring awareness of the public towards particular dangers.
- **Cyber-warfare**: It includes attacks issued by nation states against networks and communication infrastructures belonging to another state or political entity. It aims to jeopardize the confidentiality, integrity and availability of computer systems. These kinds of attack can range from companies' website defacements, Denial of Service (DoS) attacks, gathering sensible data to the physical destruction of the targeted installation [Nicholson et al. 2012].

2.5.4 Attacks techniques classification

Attacks against industrial installations can be launched through several vectors such as:

Advanced Persistent Threat (APT)

Considered as the most serious threat targeting industrial systems. The APT is a sophisticated adversary with significant resource engaged in information warfare in support of long-term strategic goals [NIST 2011, Blumbergs 2014]. It creates opportunities to achieve its objectives by using multiple attack vectors. These objectives typically include establishing and extending footholds within the targeted installation networks in order to exfiltrate sensitive information or disturb production process. It can also position itself to carry out these objectives in the future [Ussath et al. 2016, Lemay et al. 2018]. Industrial installations have already been targeted by APT groups such as in Stuxnet and Ukrainian attacks.

Spearfishing

In this kind of attacks, a fake email is sent to the victim in order to push her to perform an action such as opening a link to a website that is infected with malware or downloading an attached malicious content. The email content is generally personalized in order to increase the probability that the victim will open it [Anton et al. 2017]. Several attacks targeting ICS involved spearfishing such as in the Ukrainian blackout [Lee et al. 2016]. Indeed, the attackers send a malicious Office document via emails to administrative employees of the electricity company. When opened, the malicious document installs a malware on the victim system.

Malware

They are malicious computer programs specially developed to infect systems and to damage them. Industrial systems are threatened both by IT malwares and also by those specifically created for targeting them [Nai Fovino et al. 2009]. Thus, the infection of some installations by Conficker or Slammer [Moore et al. 2003] is a collateral damage and not an intended effects from their creators [Dine et al. 2016].

Ransomware

It is a specific type of malware that encrypts files and data of an information system and renders them unexploitable. Then, they present a message asking the victim for a payment in return for a decryption key to get the documents unlocked again. The payment is usually done through the use of virtual money [Mansfield-Devine 2016]. Several industrial installations have been hit by this kind of malware. As an example,

in October 2015, hackers have targeted a French wind turbine company and encrypted the server managing communication [Protais 2016]. They asked for 4.000$ to be paid through paypal or bitcoins. The staff of the company did not pay the ransom and succeeded to make the installation working again after an interruption of 15 days.

Botnet

It is a network composed of compromised hosts controlled remotely by hackers. It is based on a self-propagating malware that infects vulnerable hosts and is designed to perform specific malicious tasks after being triggered. It is mainly used to carry out several kinds of malicious activities such as Distributed Denial of Service (DDoS) attacks, steal personal user information, and spam [Hallman et al. 2017, Sagala et al. 2017]. The Mirai attack is one of the best example of botnet network involving industrial IoT devices.

Social engineering

It is a kind of psychological manipulation. It relies on human vulnerability in order to bypass the security of a system. Social engineering techniques include persuasion, coercion, urgency, authority, impersonation or request for help [Xiangyu et al. 2017][Bakhshi 2017]. The ultimate aim of the attackers is making the victim complies with the attacker request. Spearfishing is a kind of a modern version of social engineering technique.

2.5.5 Targeted vs Untargeted attacks

On the base of the attackers intention, attacks involving ICS can be classified as targeted or untargeted attacks:

- Targeted attacks: They are attacks specially tailored to hit industrial installations. In this kind of attacks, perpetrator requires deep knowledges of the target installation specification. Stuxnet is the best example of targeted attacks.
- Untargeted attacks: Since ICS have adopted IT standards and protocols, they could be hit by the same threats that target classical IT systems. In these cases, ICS could be seen as collateral damages and not the main target. Thus, the infection of several industrial installations by the worm Conficker was an accidental side-effect and not a desired one.

2.5.6 Discussion

Attacks detailed in Section 2.4 do not aim to be exhaustive. Indeed, its purpose was mainly to describe significant cyberattacks that have hit industrial installations. In Table 2.2, we apply the previous taxonomy to these attacks.

	Maroochy Breach	Conficker	Stuxnet	Ukrainian Blackout	Mirai
Classification	Upon	Upon	Upon	Through	By
Attackers	Disgruntle employee	-	State	State	Cybercriminal/ Scriptkiddies
Motivations	Revenge	-	Cyberwarfare (Sabotage)	Cyberwarfare	Revenge (?)
Techniques	Internal programs	Malware (Worm)	APT	APT	Botnet
Targeted/ Untargeted	Targeted	Untargeted	Targeted	Targeted	Targeted

Table 2.2: Significant ICS attacks classification

One should also note that attacks techniques and vectors are quickly evolving and getting more and more sophisticated. Thus, what was in the past accidental infections or collateral damages became now deliberated and targeted attacks [Langner 2011].

Thus, unlike Stuxnet that encapsulates a precompiled payload, authors in [McLaughlin and McDaniel 2012] propose a tool to dynamically generate PLC payload. This tool reduces the attacker necessary prerequisite knowledge for targeting any installation.

On the other hand, these evolutions make useless implemented security mechanisms. Thus, the Mirai attack bypassed DDoS detection and remediation technologies implemented by DNS service providers [Schneier 2016]. Indeed, they are able to mitigate attacks involving one thousand devices sending 500 DNS requests per minute, but not tailored to face an attack involving 400,000 devices sending 25 DNS requests per minute.

2.6 ICS security Research axes

There are 3 research axes on the security of ICS [Kieseberg and Weippl 2018, Nazir et al. 2017, Khaitan and McCalley 2015]:

- Cyber-protection: It aims to apply protection security mechanisms to Industrial Control Systems. These mechanisms aim to ensure authentication and communications confidentiality. It also includes proposing novel methods for verifying and validating both hardware and software components and more secure communication protocols.
- Cyber-defense: It aims mainly to apply intrusion detection (IDS) and intrusion protection systems (IPS) to detect and react against a cyberattack.
- Cyber-resilience: It aims to build more resilient systems that are able to continue functioning even under an attack. This continuity can be ensured under a degraded mode that ensures the availability of essential services. Indeed, ICS must be adaptive, resilient to failures of individual components, and able to maintain an overall situation awareness.

2.7 Conclusion

We have presented in this Chapter a wide overview of industrial Control Systems vulnerabilities and threats. We have shown that these systems are targeted as they are used to manage critical infrastructures playing important economical and social roles. We have also described the profiles and motivations of the attackers. We have then given a list of significant cyberattacks that have targeted industrial installations and provided a taxonomy of several kinds of these attacks.

CHAPTER 3

Security analysis of WSN-based SCADA systems: Case of the WirelessHART protocol

3.1 Introduction

The security in Industrial Control Systems is a major concern. Indeed, these systems manage installations that play an important economical role. Even more, targeting these systems can lead not only to economical losses but can also threaten human lives [Huang et al. 2009].

Therefore and as these systems depend on sensing data, it is important to secure communication channels between these sensors and the main controllers. This issue is more challenging in Wireless Sensor Networks as the use of wireless communications brings its own security weaknesses.

Indeed, the increasing use of wireless connections due to their flexibility and easy deployment, management and maintenance also brings new security challenges. Therefore, it becomes obvious that additionally to real-time requirement, any communication protocol used in these systems must ensure the availability and the integrity of data collected from these sensors. Several communication protocols were specially developed to meet this requirement in terms of time, availability, and security. The most important ones are ZigBee Pro [ZigBee Alliance], WirelessHART [HART Communication Foundation], and ISA 100.11a [Wireless System for Automation].

In this Chapter, we propose an in-deep security study of the WirelessHART protocol. This latter is the leading protocol for Wireless Industrial Sensor Networks (WISN)

and is the first international approved standard. We give a detailed description of its security mechanisms. We show how these mechanisms are used along with other non-security mechanisms to ensure security requirements. Then, we assess their strengths and emphasize their weaknesses and limitations.

The rest of the Chapter is organized as follows. We discuss in Section 3.2, previous work on the security of WirelessHART. In Section 3.3, we describe the functioning of a WirelessHART network. We detail in Section 3.4 its security and non-security mechanisms and show how they are involved in ensuring security requirements. In Section 3.5, we discuss the ability of WirelessHART to mitigate WISN threats. We present in Section 3.6, some countermeasures that can strengthen its security. Finally, Section 3.7 presents the conclusion of this work.

3.2 Related Work

SCADA systems are facing security challenges they were not initially designed to deal with. This is mainly due to the increasing interconnections between the factory floor and the corporate networks and the move from the use of proprietary owned communication standards to open international standards [Igure et al. 2006]. But there are differences between SCADA and traditional IT networks. Stouffer et al. summarize some of them in [Stouffer et al. 2011]. The most significant ones are that SCADA systems operate continuously with little down time, they are designed to meet high performances in terms of reliability and safety, and they are expected to work for 10 or 15 years long. For these reasons traditional security mechanisms used in IT must be adapted before deploying them in SCADA systems. In the literature we find that researchers focus on applying intrusions detection systems and firewalls in SCADA environment [Larkin et al. 2014, Tabrizi and Pattabiraman 2014], and only treat the wired part of the network. For example, most proposed IDS focus on wired communication protocols such as Modbus [Huitsing et al. 2008] or DNP3 [Fovino et al. 2010].

We find only few studies on the security of wireless sensor networks used in industrial environment. Coppolino et al. propose in [Coppolino et al. 2010] an architecture for an intrusion detection system for critical information infrastructures using wireless sensor networks. Their solution is a hybrid approach combining misuse and anomaly based techniques. In the same way, there are few number of studies dedicated to WirelessHART and even less to its security. Mostly these studies show the performances of this protocol by evaluating its capabilities to operate in an industrial environment and its capacity to meet real-time requirement [Han et al. 2011, Kim et al. 2008,

Song et al. 2008]. Other studies make comparison between WirelessHART and its principal competitors [Alcaraz and Lopez 2010, Petersen and Carlsen 2011] such as ZigBee Pro and ISA 100.11a.

In [Raza et al. 2009] Raza et al. discuss the strengths and the weaknesses of security mechanisms and analyze them against the well known threats in the wireless sensor networks. They conclude that WirelessHART is strong enough to be used in the industrial process control environment and specifically they state that sybil attacks are almost impossible in this kind of networks. Alcazar and Lopez identify in [Alcaraz and Lopez 2010] vulnerabilities and threats in ZigBee PRO, WirelessHART and ISA 100.11.a. They analyze in detail the security features of each of these protocols. For them, WirelessHART offers strong authentication capabilities before and after deployment. However, they recommend to add a rekeying process to WirelessHART to enforce its resilience to sniffing attacks and thereby key disclosure.

We must note that these studies are based on the specifications of the standard without conducting any tests.

For our part, we provide in this Chapter a detailed description of WirelessHART mechanisms that are involved in ensuring reliable and secure communication. We also assess in-deep their efficiency to mitigate attacks and emphasis their weakness and limitations. Furthermore, we describe for each weakness a practical scenario that exploit it. Finally, we provide some solution proposals to mitigate these weaknesses.

3.3 WirlessHART overview

WirelessHART [HART Communication Foundation] is a major wireless protocol developed by HART Communication Foundation for industrial process automation. It is included in version 7 of the HART standard, a widely-used wired protocol for industry. It was released in 2007 and was approved as a IEC 62591 international standard in 2010. It uses a time-synchronized, self-organized and self-healing mesh architecture to provide a reliable, secured and real-time communication.

In this Section, we present an overview of the functioning of WirelessHART networks and describe the different mechanisms implemented for ensuring communications reliability and security. The efficiency and limitations of theses mechanisms are then analyzed and discussed in Section 3.4.

3.3.1 Topology of a WirelessHART network

A typical WirelessHART network, illustrated in Figure 3.1, is composed of the following devices:

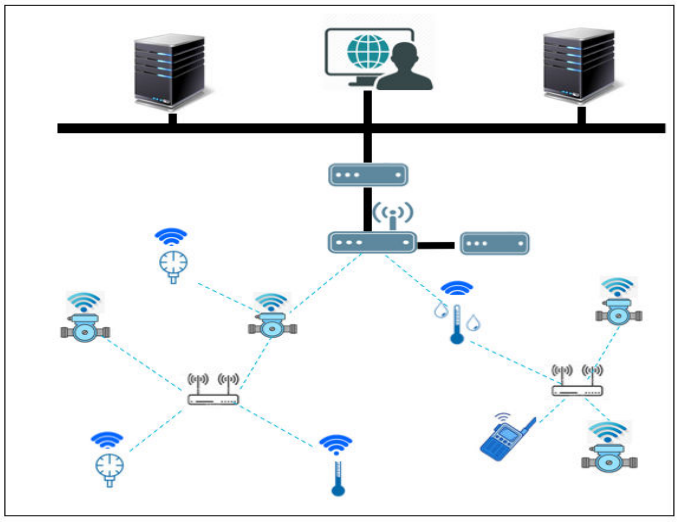


Figure 3.1: Example of a WirelessHART network

- a Gateway which connects the wireless network to the plant automation network, allowing data to flow between the two networks. It can also be used to convert data and commands from one protocol to another one;
- a Network Manager responsible for the overall management, scheduling, and optimization of the wireless network. It generates and maintains all of the routing information and also allocates communication resources;
- a Security Manager responsible for the generation, storage, and management of cryptographic keys;
- Access Points which connect through a wired connection, the Gateway to the wireless network;
- Field devices deployed in the plant field and which can be sensors or actuators;
- Routers which are used to forward packets from one network device to another;
- Handheld devices which are portable equipments operated by the plant personnel used in the installation and maintenance of a network device.

The Network Manager is one of the most important device in a WirelessHART network. It is responsible for the overall management, scheduling, and optimization of the wireless network. It generates and maintains all of the routing information and also allocates communication resources. Along with it there is also the Security Manager which is responsible for the generation, storage, and management of cryptographic keys. These two devices can be implemented in one entity.

3.3.2 WirelessHART stack

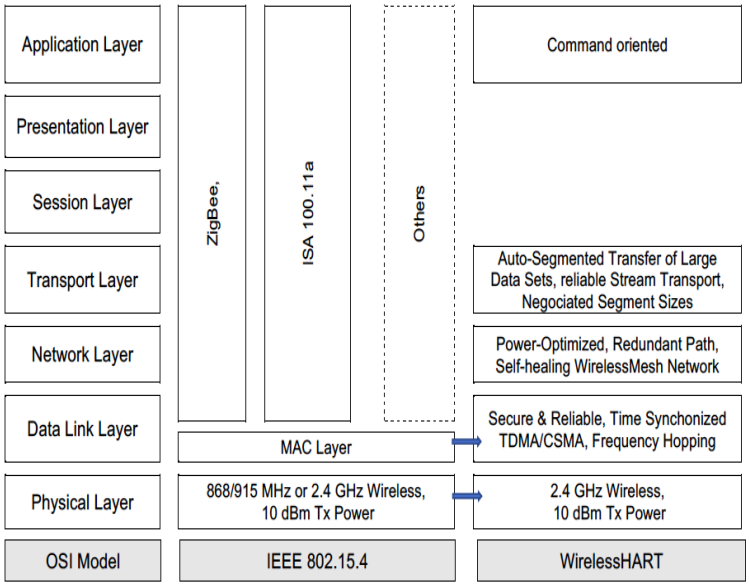


Figure 3.2: WirelessHART protocol stack [Deji et al. 2010]

The WirelessHART protocol is based upon the IEEE 802.15.4 standard [IEEE 802.15.4-2006] that specifies the physical layer and media access control (MAC) for low-rate wireless personal area networks (LR-WPANs). However, unlike other WISN protocols like ZigBee Pro or ISA 100.11a that use the physical and the MAC layers of IEEE 802.15.4 and develop on the top of them their own upper layers (i.e., Network, Transport and Application layers), WirelessHART implements partially the IEEE 802.15.4 physical layer and extends its MAC layer with the add of new functionalities. On the top of them, it implements its own data link, network and transport layers. Finally, it shares the same application layer as the wired HART protocol (with the add of wireless commands).

Thus, the same network can include indifferently devices implementing WirelessHART and devices implementing HART. On the other hand, WirelessHART packets are legitimate IEEE 802.15.4 packets of type data without encryption. The physical

and the MAC headers of a WirelessHART packet are respectfully the same as those of an IEEE 802.15.4 packet.

A brief description of the WirelessHART protocol stack is given below:

- Physical Layer (PhL): It is based on IEEE 802.15.4-2006 standard and operates in the 2.4 GHz. It is responsible of wireless transmission and reception.
- Data Link Layer (DLL): It is composed of a higher sublayer logical link control (LLC), and a lower sublayer medium access control (MAC). It is responsible of preparing packets for transmission, managing time slots and updating different tables. It provides hop-by-hop authentication.
- Network Layer (NL): It ensures end-to-end integrity and confidentiality. It provides routing features. It receives the Network Protocol Data Unit (NPDU) from the DLL and checks if it has to be transmitted to the AL or has to be resent to the DLL to be forwarded to next device.
- Transport Layer (TL): It provides mechanisms to ensure data delivery without loss, duplication or misordering to its final destination. It supports acknowledged and unacknowledged transactions.
- Application Layer (AL): It is a command based layer. It is used to send sensing data from field devices to the Network Manager, and to send commands from the Network Manager to the field devices. Additionally to WirelessHART commands, it supports common HART commands (inherited from wired version).

3.3.3 WirelessHART packets

In WirelessHART there are five packets types:

1. **Data packet**: It encapsulates packet from the NL in transit to their final destination device. They are generated and processed in the Network layer;
2. **Ack packet**: It represents the immediate response sent to acknowledge the good reception of a packet;
3. **Keep-alive packet**: It is used for maintaining connection between neighboring devices;
4. **Advertise packet**: It is used for providing information to neighboring devices trying to join the network;

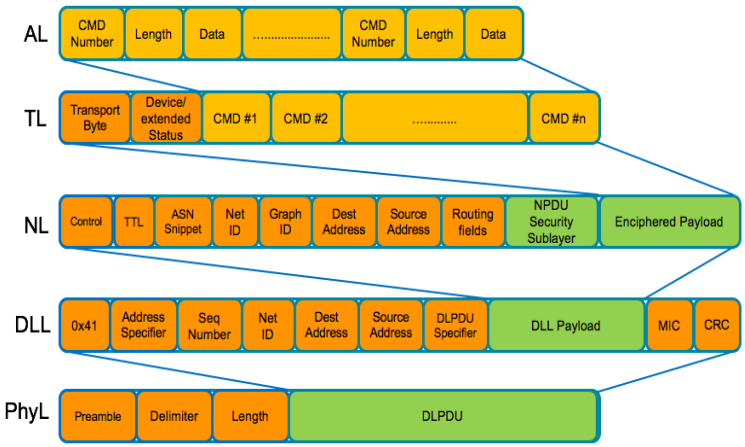


Figure 3.3: WirelessHART packets structure

5. **Disconnect packet**: It is used to inform neighboring devices that the device is leaving the network.

Ack, Advertise, Keep-Alive and Disconnect packets are generated and processed in the Data Link Layer and are not propagated to the network layer or forwarded through the network. This means that these packets are only used in local communication between neighbors. The Data packet is the only kind of packets that is transmitted in an end-to-end communication. During the transmission, the data packet payload is enciphered. We show in Section 3.4 that the choice to only encipher the Data packet payload and not those of the other types of DLPDU introduces a potentially harmful breach in the WirelessHART communication security scheme.

Data Link Protocol Data Unit (DLPDU) Structure

The Data Link Layer is responsible of preparing packets for their transmission.

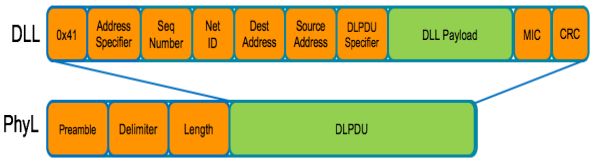


Figure 3.4: WirelessHART DLPDU structure

The structure of a WirelessHART DLPDU, illustrated in Figure 3.4, is:

- A header: indicating the source and destination addresses (which can be 2 or 8 bytes long), a Sequence Number, the type of the DLPDU, its priority and the type of the network key used for the generation of the MIC (Message Integrity Code).
- The DLPDU payload which depends on the type of the packet (in the case of a Data DLPDU, it encapsulates a NPDU).
- A footer composed of a Keyed Message Integrity Code (MIC) calculated on the header and the payload, and a Cyclic Redundancy Check (CRC16) used for error detection.

Network Protocol Data Unit (NPDU) Structure

The common structure of a WirelessHART NPDU, illustrated in Figure 3.5, is as follows:

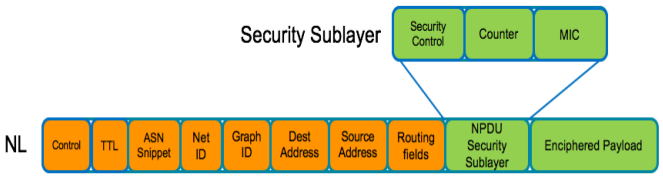


Figure 3.5: WirelessHART NPDU structure

- A header: indicating the source and the final destination address, packet time creation, packet maximum number of allowed hops before it is discarded, and the Graph ID to be used for routing.
- A security sublayer: it is part of the NPDU header and indicates the key and the nonce (counter) used for enciphering the NPDU payload and calculating the MIC on the header and the payload.
- A payload: containing Application Layer commands. It is enciphered by the session key (or the join key during the joining process).

3.3.4 WirelessHART network functioning

Time Division Medium Access (TDMA)

WirelessHART uses Time Division Medium Access (TDMA) to control access to the medium. It provides collision free and deterministic communication between two wireless devices.

Thus, time is divided into fixed-size intervals of 10 ms called *slot* and each communication between two devices occurs in one *slot*. *Superframes* are collection of slots repeated continuously with a fixed repetition rate. Typically, two devices are assigned to one time slot (i.e., one as the sender and the second as the receiver). Only one packet is transmitted in one slot from the sender to the receiver which has to reply with an acknowledgment packet in the same slot. In the case of a broadcast message, there is one sender and multiple receivers assigned to the same slot. In this case the message is not acknowledged.

Channel hopping and blacklisting

To enhance reliability, channel hopping is combined with TDMA to provide frequency diversity and avoid interferences. Each slot is used on multiple channels at the same time by different nodes. The 2.4 GHz band is divided into 16 channels numbered from 11 to 26 which provides up to 15 communications in the same slot (channel 26 is not used). So, each slot is identified by a number called Absolute Slot Number (ASN) and a channel offset. The ASN represents the count of elapsed slots since the start of the network.

Thus, a *link* designates a full communication specification between adjacent devices in a network. It indicates the source and destination address pairing, slot and channel offset assignment, direction of communication (sending or receiving), dedicated or shared communication, and type (unicast or broadcast). Links are assigned to *superframes* as part of the scheduling process.

WirelessHART also allows channel blacklisting. The administrator can restrict the channel hopping by the devices to selected channels. This can be done for example in order to avoid interferences on some channels caused by another wireless network. Thus, the *ActiveChannelArray* represents the list of active channels used in the network.

Time synchronization

As each communication occurs in a slot, clocks of all devices in the network must be synchronized. Indeed, it is imperative that each device knows exactly when a slot starts. Therefore, WirelessHART includes synchronization and time keeping.

The transmission of a packet, as illustrated in Figure 3.6, starts at a specified offset from the start of the time slot. This offset allows the source and destination to set their frequency channel and allows the receiver to begin listening on the specified channel. Since there is a tolerance on clocks (see Table 3.1), the receiver starts to listen before the ideal transmission start time and continues to listen after that ideal time.

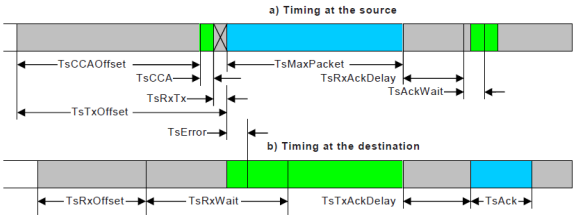


Figure 3.6: Slot timing

Also, when a device receives a packet, its time of arrival is used to calculate the difference between the actual time of arrival and the expected time of arrival. This difference (δt) is communicated in every acknowledgment reply packet sent to the source device.

The Network Manager specifies for each device one of its neighbor to be used as time synchronization source. When a packet from a time synchronization neighbor is received, the network time of the receiving device is adjusted.

Communication Scheduling

The Network Manager executes the scheduling algorithm to allocate slots to wireless devices. To do that, it needs to have a good knowledge about the network topology and connections quality between devices. The scheduling algorithm is executed each time a new device joins the network and when significant changes are reported.

WirelessHART do not provide a scheduling algorithm but proposes the strategy summarized below:

1. **Data superframes:**

   (a) Slots are allocated starting with the fastest to the slowest scan rate.

Symbol	Definition	Required value
TsTxOffset	Time from the start of the slot to the start of the preamble transmission	2 120 μs ± 100 μs
TsRxOffset	Start of the slot to when transceiver shall be listening	1 120 μs ±100 μs
TsRxWait	The minimum time to wait before the transmission beginning; this correlates to the amount of drift between the neighbors that can be tolerated to maintain communications.	2 200 μs ±100 μs
TsMaxPacket	The amount of time it takes to transmit the longest possible packet that includes PhL preamble, delimiter, length and DLPDU	4 256 μs
TsTxAckDelay	End of message to start of ACK; the destination device shall validate the packet, and generate an ACK, if required, during this interval.	1 000 μs ±100 μs
TsRxAckDelay	End of the PhPDU transmission to when the transceiver shall be listening for ACK	800 μs ±100 μs
TsAckWait	The minimum time to wait for the start of an ACK	400 μs ±100 μs
TsAck	Time to transmit an ACK (26 bytes)	832 μs
TsCCAOffset	Start of slot to the beginning of the *Clear Channel Assessment* (CCA)	1 800 μs ±100 μs
TsCCA	Time to perform CCA (8 symbols)	128 μs
TsRxTx	The maximum time it takes to switch from reception to transmission or vice versa (12 symbols)	192 μs
TsError	This is the difference between the actual start of message and the ideal start of message time as perceived by the receiving device.	

Table 3.1: Slot timing definitions and values [HART Communication Foundation]

(b) Starting from the device furthest from the gateway, one link for each en-route network device to the gateway is allocated. A 2nd dedicated slot for retry is also allocated.

(c) Each transmission is also scheduled with a retry on another path, if one is available.

(d) One network device can only be scheduled to receive once in a slot.

(e) Event notification shares data slots.

2. **Management superframe:**

(a) Management superframe has priority over data superframes.

(b) The graph should be traversed by breathfirst search, starting from the gateway.

(c) It includes Advertisement slots and command request/response slots.

Routing

WirelessHART implements in the Network Layer, two methods of routing packets throughout the network, i.e., graph routing and source routing.

- Graph routing: a graph is a collection of directed paths that connect network devices. It is built by the Network Manager based on its knowledge of the network topology and connectivity. Every graph has a unique graph identifier that is inserted in the network packet header. Each device, receiving this packet, must forward it to the next hop belonging to that graph. This routing method is used for normal communications, in both upstream (from a device to the network manager) and downstream (from the Network Manager to a specific device) directions.
- Source routing: it is a single directed route between a source and a destination device. The complete route is completely inserted in the network packet header by the sender device. Each intermediate device propagates the packet to the next device indicated in the source route field. This method of routing is used only for testing routes, troubleshooting network paths or for ad-hoc communications.

The construction of routing table is an important feature of the Network Manager. It is based on information transmitted periodically by wireless devices to the Network Manager called *health reports*.

A basic routing strategy is summarized below.

1. If there is a one hop path from a device to the gateway it should be used.
2. The maximum number of hops to be considered when constructing the initial graph is 4.
3. Minimize the ratio of signal strength on number of hops.

3.3.5 Security mechanisms

WirelessHART [HART Communication Foundation] was developed to provide reliable and secure communications for industrial process automation requirements. In particular, security is one of its important features. Therefore, it implements several mechanisms to ensure data confidentiality, authenticity and integrity in hop-by-hop and end-to-end transmissions.

The hop-by-hop transmission security is provided by the Data Link Layer (DLL) using a cryptographic key called "Network Key" shared by all devices part of the wireless network. It defends against attackers who are outside the network and do not share its secret (Outside attacker). The end-to-end security is provided by the Network Layer (NL) using a cryptographic key called "Session Key" known only by the two communicating devices. It defends against attackers who may be on the network path between the source and the destination (Inside attacker).

Security at Data Link Layer

To ensure hop-by-hop security a keyed Message Integrity Code (MIC) is implemented in Data Link Layer. In WirelessHART each Data Link Protocol Data Unit (DLPDU) is authenticated by the sending device using a cryptographic key shared by all devices that belong to the same network. Therefore, before processing any received DLPDU, a device must check the MIC to verify the identity of the sending device. We must note that the DLPDU itself is not enciphered but authenticated by a four-byte MIC generated with CCM* mode (Combined Counter with CBC-MAC (corrected)) using the AES-128 block cipher. Each device is configured with two kinds of cryptographic keys: the well-known key and the network key:

- The well-known key which is used in the Advertisement and joining process. It is identical for all devices and has a built in value set to *7777 772E 6861 7274 636F 6D6D 2E6F 7267* hexadecimal,
- and the network key which is used for all other DLPDUs. It is supplied by the Network Manager to a device when it joins the network.

Security at Network Layer

The end-to-end security is provided by the Network Layer (NL) using a cryptographic key called "Session Key" known only by the two communicant devices. It defends against

attackers who may be on the network path between the source and the destination (Inside attacker). The network layer uses also a keyed Message Integrity Code (MIC) for the authentication of the Network Protocol Data Unit (NPDU). Additionally, it uses the same key to encrypt and decrypt the NPDU payload. The end-to-end security is session oriented, i.e., it provides a private and secure communication between a pair of network devices. Each session is defined by the following two elements:

- a session key: it is a dedicated 128 bits cryptographic key. It is used to encipher the NPDU payload and to authenticate the whole NPDU.
- a session counter: it is a 32 bits value that defends against replay attacks and used as the nonce for generating the NPDU MIC. Each device keeps a history of received nonce counter.

Four sessions are set up as soon as any device joins the network. They allow the transmission of sensing data from a device to the Network Manager, and the transmission of commands from the Network Manager to a field device. Each communication can be done in a unicast or a broadcast mode. In addition, each device has a join session which cannot be deleted.

All used cryptographic keys are generated and distributed to devices by the Network Manager. The *Join Key*, used as session key during the joining of a new device, is the only key that can also be written directly to a device through its maintenance port.

CCM* mode (Combined Counter with CBC-MAC (corrected))

The WirelessHART standard applies encryption at two places, at the data link layer for authentication and at the network layer for authentication and encryption. The WirelessHART standard adopts the CCM* encryption algorithm defined in the IEEE 802.15.4 standard [IEEE 802.15.4-2006]. CCM* is an extension of CCM [Dworkin 2004] i.e., a combination of *Cipher Block Chaining-Message Authentication Code* (CBC-MAC) and *Counter* modes. CCM is not designed to support partial processing or stream processing. Indeed, CCM is intended for use in a packet environment i.e., when all of the data is available in storage before CCM is applied.

In the WirelessHART standard a two way communication is completed within a 10ms timeslot. At the data link layer, a receiver of a maximum-length message must authenticate the message and prepare the Ack message, which must be applied CCM*, in 1ms [Deji et al. 2010]. It will be extremely challenging for the receiver to execute this in time.

As WirelessHART defines the formats of all packet types, we do not need to wait until the whole packet is receive to start applying CCM*. Once the MAC message header is received, we could start applying CCM* as we could stream process the incoming message in 16-bytes blocks. This is called incremental encryption [Deji et al. 2010].

In order to guaranty the strength of the algorithm, it is recommended [IEEE 802.15.4-2006] that its implementation shall limit the total amount of data that is encrypted with the same key. The CCM* encryption and authentication transformation shall not apply the same key to more than 2^{61} times in total [IEEE 802.15.4-2006].

On the other hand, in order to avoid timing error attacks, the US NIST recommends [Dworkin 2004] that the implementation of CCM ensures that when an error message is returned, an unauthorized party cannot distinguish whether the error message results from the use of invalid inputs (i.e. key, nonce, etc) or from the comparison fail between the transmitted and the calculated MIC.

Nonce

In the DLL, the nonce is 13 bytes long and is the concatenation of *the Absolute Sequence Number* (ASN) (5 bytes) and the source address (2 or 8 bytes) indicated in the DLPDU. In the case of short addresses (2 bytes) the nonce is padded with 6 bytes set to 0x00. It is used for MIC calculation to authenticate the DLPDU.

In the NL, the nonce is formed by the concatenation of one byte indicating the session type (join or normal), the *session counter* (4 bytes) and the source address with padding in the case of short addresses. It is used for MIC calculation to authenticate the NPDU and for enciphering its payload.

3.4 WirelessHART Security analysis

The WirelessHART protocol implements several mechanisms that are involved directly or indirectly in ensuring secure communication. Hereafter we describe how these mechanisms are involved in ensuring security services. We also discuss the efficiency of these mechanisms and how far they are able to protect against security threats targeting WSN.

Security requirements in WSNs include [Wang et al. 2006, Roosta et al. 2006]: authentication, confidentiality, integrity, availability authorization, non-repudiation,

freshness, graceful degradation and forward and backward secrecy. Hereafter, we discuss how WirelessHART fulfills these requirements.

Authentication

It ensures that only legitimate nodes have access to network services.

In WirelessHART, node's authentication is ensured both in hop-by-hop and end-to-end communications using the appropriate cryptographic key. In *the hop-by-hop communication* the address of the sender is inserted in the header of the DLPDU. This latter is then authenticated with the MIC (keyed message integrity code) generated using *the network key*. The destination node authenticate the source node by calculating, using the network key, the DLPDU MIC and comparing it with the one included in the DLPDU header. If the verification succeeds, the node is authenticated, an Ack DLPDU is sent to the source node if required (i.e., unicast communication), and the DLPDU is processed. Otherwise, the DLPDU is discarded. Thus, only legitimate node can send and receive packets.

In *the end-to-end communication* the address of the source node is also inserted in the NPDU header that is authenticated with a MIC generated using *the session key*. This key is specific to each node's pairwise and known only by the sender and the destination.

Discussion: Authentication in DLL level is efficient to identify and discard packets sent by illegitimate nodes as they are not provisioned with *the network key*. But it is inefficient to protect against nodes that knows the network key. As an example, a compromised legitimate node can forge a fake packet and insert the address of another node as source address. The destination node will be misleaded and will authenticate the packet as a legitimate one. An attack described in [Bayou et al. 2015a] that uses this weakness, allows an attacker to disconnect one or several nodes from the network. In the NL level, the authentication is more stronger as it is based on *session keys* known only by the two communicant nodes. Nevertheless, in broadcast communication, the session key is shared by all nodes. Consequently, a fake packet can be forged by a compromised node and broadcasted to other nodes that will authenticate and process it as a legitimate packet. This kind of attacks is implemented in [Bayou et al. 2016b] that shows its harmful impact on the network.

Confidentiality

It ensures that a given message cannot be understood by anyone other than the desired recipients.

In the DLL level, all DLPDUs are sent in plain text. Data DLPDU is specific as its payload is an NPDU. Indeed, in the NL level, the NPDU payload is encrypted using the session key. Thus, even if a Data DLPDU is sent in plain text, its sensitive part is previously secured by the NL.

Discussion: WirelessHART ensures the confidentiality of sensitive data (i.e., AL commands) by encrypting the NPDU payload. Nevertheless, exchanged data in DLL level can be eavesdropped by an attacker that can retrieve useful information inserted in both the header (i.e., source and destination addresses, current ASN, etc.) and the payload (i.e., joining slots).

Integrity

It ensures that a message sent from one node to another is not modified during its transmission toward its final destination.

Two mechanisms are implemented in WirelessHART to avoid packets modification during their transmission: *a*) The *Cyclic Redundancy Check* (CRC) used to detect bit errors during the transmission. It is calculated over the entire packet and is implemented in the DLL. *b)* The *keyed message integrity code* (MIC) is used to ensure that the DLPDU is originated from an approved, authenticated device. The MIC is generated and checked using CCM* mode (combined counter with CBC-MAC (corrected)) in conjunction with the AES-128 block cipher to provide authentication. To generate the MIC, the network key is used in the DLL and the session key (unicast or broadcast) is used in the NL.

Discussion: In the DLL, the integrity is based on the MIC calculation using the network key. Even if this key is shared by all legitimate nodes, it is hard for an attacker to modify a packet at the DLL level. Indeed, each packet is created, transmitted and processed during one timeslot that is used as a nonce for the MIC and is also inserted in the DLPDU header. A node will discard a packet if the timeslot indicated in its header is different from the timeslot transmitted within. At the NL level, unicast transmission are secured as packet integrity is ensured by the MIC using a dedicated secret key known only by two communicant nodes. This is not the case of broadcast communications that are secured by a shared session key known by all legitimate nodes. This weakness is used in *the broadcast attack* [Bayou et al. 2016b], where a compromised node is used to

modify packets broadcasted by the Network Manager and secured using *the broadcast session key*.

Availability

It ensures that the desired network services are available.

The availability is the most important requirements. WirelessHART implements several techniques to ensure it.

- Channel hopping: 15 channels are used for communication and the used channel is changed at each slot. Also, if the network administrator notices interferences on a specific channel, he can blacklist it and remove it from the channel hopping pattern.
- Retry slots: For each slot assigned for the transmission of a packet, the next two slots are assigned for retries. The first one to the same destination using another channel and the second one using another path.
- Path redundancy: As indicated in previous section, WirelessHART implements graphs as routing techniques. For each graph, nodes are conFigured with two nodes as next hop (the first node is the default path and the second node is used for transmission retry).
- Routing table: They are periodically updated by the Network Manager. To do that this latter uses information on the quality of links sent periodically by all nodes. Thus, a path is reconFigured if it presents a high communication loss rate.

Discussion: The techniques implemented by WirelessHART to ensure the availability of the network are able to deal with intentional or unintentional perturbations that can lead to lose of communication. Nevertheless, these mechanisms cannot mitigate all attacks targeting the availability. As an example, *channel hopping* and *blacklisting* allow the network to deal with an attack that creates interferences on some communication channels. However, if the attack targets all the fifteen channels used by WirelessHART, nodes will be unable to send or receive any packets.

Authorization

It ensures that only authorized sensors can use network services.

In WirelessHART, authorization (i.e., transmit and receive) is ensured throughout the communication schedule. Indeed, as TDMA is used for managing access to the medium, the Network Manager builds the communication schedule and then transmit it to each node. Thus, each node knows exactly when it is supposed to transmit or receive a packet and to whom or from whom.

Discussion: An attacker should identify slots where a node is configured as receiver in order to send it any packets. Otherwise, the target node will not receive the packet. The attacker can eavesdrop the target node communications to Figure out when it is receiving packets (by sending an Ack DLPDU, the target node indicates that it receives a DLPDU). He can also use joining slots (i.e., slots used to receive joining request) that are periodically broadcasted by the target node through advertising DLPDU.

In the DLL, only the destination address is checked and no additional verifications are performed on the source address or the packet type to verify that they match those indicated in the communication schedule. When a packet is received, only the MIC validation in both DLL and NL, indicates if the action (reception of the packet) is authorized or not.

On the other hand, some sensitive AL commands (such as "write network key") are only executed if they are received from the Network Manager. This verification is performed in the NL using the session key with Network Manager (either the unicast or the broadcast key).

Non-repudiation

It denotes that a node cannot deny sending a message it has previously sent.

The packet sender identity (address) is indicated in both DLL and NL headers. Each of them is authenticated by the appropriate MIC.

Discussion: This mechanism can be bypassed by an attacker as the cryptographic keys used in the MIC calculation (the network key in the DLL and the broadcast session key in the NL) are shared by all nodes part of the network. In both cases a node can deny to be the source of a packet as this packet can be sent by any other node having knowledge of the used key.

Freshness

It implies that data are recent and ensures that it is not a replay of previous messages.

To ensure this, WirelessHART uses at the DLL the least significant byte (LSB) of the ASN (Absolute Slot Number coded with 5 bytes) as the sequence number. The ASN has the same value for all nodes and represents the time elapsed since the beginning of the network. The sequence number is included in the header of each transmitted DLPDU. Thus, at the reception of each DLPDU, the destination node checks that the sequence number in the header matches the timeslot the DLPDU is transmitted in. If it does not match, this indicates that something wrong happens and the DLPDU is discarded. On the other hand, the Ack DLPDU has the same sequence number that the received DLPDU as they are both transmitted in the same timeslot.

In the WirelessHART standard, as the sequence number is a kind of time stamp, it is not incremental with messages. We must notice that as it is coded using 1 byte, the sequence number rolls over after 256 slots (each 2.56 seconds) [Deji et al. 2010]. The ASN is also used to form the nonce for the MIC calculation in the DLL.

In the NL, a 4 byte nonce counter is used. Each node has one nonce counter per session. Each new message within a session is associated with the current nonce counter which is incremented by one after each new message is built. Each node keeps track in a sliding window, of all nonce counters of received messages. A message is discarded if it has the same nonce counter that a previous received one. The nonce counter is also used to construct the nonce, used to run CCM* for NPDU encryption and authentication.

Discussion: In order to replay a previous captured message, an attacker must be able to update message headers at both DLL and NL level. To do that, the attacker must be aware of either the network key or the session key. If it has only knowledge of the network key, the replayed message will be forwarded by nodes till its final destination where it will be finally discarded. However, even if the destination node will not authenticate the replayed message, the fact that this message where forwarded by relay nodes as legitimate can be used by the attacker to flood the network.

Graceful Degradation

It ensures that the designed mechanisms are resilient to node compromision, and the performance of the network degrades gracefully when a small portion of the nodes are compromised.

This requirement is partially covered by availability mechanisms. Indeed, channel hopping, retry slots and routing table updating allow the network to deal with any accidental or intentional failures. Thus, a node which experiences a high number of communication failure, which does not perform any transmission for a long time or

which the signal quality is deteriorating, will be removed by the Network Manager from the network and should perform a new joining request.

In addition and in order to avoid lost packets and network congestion, WirelessHART implements the following flow control mechanisms:

- A priority is assigned to each DLPDU according to its type. It can take one of the following values (from the highest to the lowest):

  1. *Command*: used to send network-related diagnostics, configuration, or control information.
  2. *Data* and *Process*: used to send measurements from process transmitters or setpoints to control devices or any other process related data or network statistics data.
  3. *Normal*: used to send any other data not meeting the criteria for "Command", "Process-Data", or "Alarm" priority.
  4. *Alarm*: used to report about only alarm or event.

  Each device is conFigured with a *priority_threshold* that specifies the lowest priority DLPDU to be accepted from another device.

- Packet buffers: WirelessHART recommends that each node has several storage buffer. At least one buffer should be reserved to receive *Alarm* priority DLPDU and at least one buffer reserved to receive *Command* priority DLPDU. These reserved buffers shall not be used to store the DLPDU of any other priority.

Discussion: The DLPDU priority allows the Network Manager to mitigate any network congestion. Thus, if the Network Manager notices any communication disturbance, it can for example, raise the *Priority_threshold* to reduce packet flow through devices.

We must also notice that the network management DLPDUs (Ack, Advertise, Keepalive, Disconnect and some specific Data) have a Command priority (highest priority). Consequently, they are always propagated through the network allowing the Network Manager to keep the network operational.

In the same time, the propagation of *alarms* packet through the network is restricted ensuring that alarm floods do not disrupt network operation. Since alarms are always time-stamped, no information regarding failure sequences is lost.

Finally, all other kinds of network packet are propagated through the network as buffer space and network capacity allows. Among this packets, *process data* has the highest priority. Operation and control of the process has lower priority to prevent network communication disruption

Forward and Backward secrecy

In addition of previous requirements, in WSN new sensors are deployed and old sensors fail. Therefore, authors in [Wang et al. 2006] suggest that *forward and backward secrecy* should also be considered. *Forward secrecy* ensures that a sensor is not able to read any future messages after it leaves the network; *Backward secrecy* ensures that a new joining sensor should not be able to read any previously transmitted messages.

Discussion: These two requirements are not ensured in WirelessHART. Indeed, as each new joining node is provisioned by the Network Manager with the current network key and session keys (dedicated unicast and shared broadcast keys), the new node can read previous exchanged messages secured using shared keys (i.e., the network and broadcast keys). However, the new node are not able to read messages secured with other nodes unicast keys. In the same manner, if a node leaves the network, it still have the knowledge of the network and the broadcast session keys. So it still can read exchanged messages that use these two keys till they are changed. On the other hand, messages exchanged in hop-by-hop communications are in plain text without any encryption.

Table 3.2 summarizes the efficiency of WirelessHART security mechanisms in ensuring above security requirements.

3.5 Security issues in WISN

We described in previous section, security mechanisms implemented by WirelessHART in order to ensure secure and reliable communication. In this Section we discuss threat mitigation capabilities of these mechanisms.

Indeed, WISN in the same manner than general Wireless Sensor Networks, can be subject to several kinds of attacks [Wang et al. 2006]. These attacks can target important mechanisms such as [Karlof and Wagner 2003]: routing protocol, data aggregation, voting, fair resource allocation, and misbehavior detection algorithms. We give below the description of some of well-known attacks on WSN [Wang et al. 2006,

	Hop-by-Hop communication	End-to-End communication	
		Unicast	Broadcast
Authentication	MIC (Network Key)	MIC(Unicast session key)	MIC (Broadcast session key)
Authorization	TDMA + communication scheduling	Source address + MIC (Unicast session key)	Source address + MIC (Broadcast session key)
Confidentiality	Plain text	CCM* (Unicast session key)	CCM* (Broadcast session key)
Integrity	CRC + MIC (Network Key)	MIC (Unicast session key)	MIC (Broadcast session key)
Availability	Retry slots, channel hopping	re-routing, routing table update	re-routing, routing table update
Non-Repudiation	Source address + MIC (Network Key)	Source address + MIC (Unicast session key)	Source address + MIC (Broadcast session key)
Freshness	ASN + MIC (Network Key)	Nonce counter + MIC (Unicast session key	Nonce counter + MIC (Broadcast session key)
Graceful Degradation	Packet priority, storage buffer, channel hopping, blacklisting, retries slots, path redundancy, priority management	Packet priority	Packet priority
Forward secrecy	Plain text+ MIC (Network Key)	CCM* (Unicast session key)	MIC (Broadcast session key)
Backward secrecy	Plain text + MIC (Network Key)	CCM* (Unicast session key)	MIC (Broadcast session key)

Table 3.2: WirelessHART security services coverage.
(Green): Efficient, (Orange): Partially efficient and (Red): not efficient or not implemented

Karlof and Wagner 2003] and discuss the ability of WirelessHART security mechanisms to mitigate them:

- Jamming attack: A malicious node disturbs transmissions of nearby nodes by emitting packets periodically or continuously.
- Eavesdropping: A malicious node listens illegally to the traffic exchanged between nodes.
- Denial of Service (DoS) attack: A malicious node overwhelms the targeted node by sending a great amount of packets that will not be able to receive legitimate packets.
- Sinkhole and blackhole attacks: A malicious node misleads routing algorithm by transmitting false information to the base station. Consequently, a part of the traffic will be redirected to the malicious node which can drop packets partially (sinkhole) or totally (blackhole).
- Hello Flooding attack: A malicious node with a large transmission range can flood a large part of the network with this kind of packets. Nodes receiving these packets, will assume that the malicious node is in their transmission range and exhaust their battery life by trying to communicate with it.
- Selective forwarding attack: A malicious node chooses selectively to drop some packets and to not forward them to their final destination.
- Wormhole attacks: In this kind of attacks a malicious node creates a virtual tunnel by capturing packets in one location and retransmits them in another location of the network. To do that, the malicious node must have a transmission range longer than other nodes or can require the help of another malicious node. As results, the malicious node can circumvent the routing protocol and lies on its location (number of hops from the base station).
- Forced delay attack: A malicious node delays the forwarding of some packets which can have harmful consequences in WISN where processes are time sensitive.
- Node compromising: An attacker by capturing a legitimate node could have access to its secret. It can inject into the network false data that can disturb its functioning.

Furthermore, WirelessHART network can be subject to the two specific following attacks that we will describe in details in Chapter 4:

- Sybil attack [Bayou et al. 2015a]: This kind of attacks was first described by Douceur in [Douceur 2002]. He shows that in the absence of a central identification authority that checks correspondence between entity and identity, a malicious

Attacks	Targeted security requirement	Mitigation techniques
Jamming	Availability	Channel hopping + blacklisting
Eavesdropping	Confidentiality	NPDU payload encipher-ing
Denial of Service (DoS)	Availability	Packet priority, storage buffer
Sinkhole and blackhole	Graceful degradation, Avail-ability	Re-routing and routing table update
Selective for-warding	Availability, Freshness	Retries and acknowledg-ment
Hello Flood	Graceful, Availability	Re-routing, packet prior-ity
Forced delay	Freshness	Retries and acknowledg-ment
Node compro-mising	Confidentiality, integrity, availability	NPDU payload encipher-ing, retries
Sybil	Authentication, authorization, availability	-
Broadcast	Authentication, integrity	-

Table 3.3: WirelessHART attacks mitigation capabilities

entity can present multiple identities. In the case of a WirelessHART network, a malicious insider node forges a fake *Disconnect* packet (used by nodes to inform their neighbors that they are leaving the network), puts the target node as the packet source address and then authenticates it using the *Network Key* (shared by all legitimate nodes). As results, receiving nodes erase the target node from their communication planning. This attack can lead to the partial or total disconnection of all nodes.

- Broadcast attacks [Bayou et al. 2016b]: In this attack, a malicious insider node uses its knowledge of the *Broadcast Session* (a key used to encipher end-to-end packets broadcasted by the Network Manager to all nodes) for injecting false commands into the networks. This attack is more harmful than the previous one as the attacker pretends to be the Network Manager and can change nodes configuration parameters.

As indicated in Table 3.3, WirelessHART security mechanism are able to mitigate a large number of security attacks. However, these mechanisms are not designed to deal with massive attacks such as a jamming attack on all transmission channels or a heavy DoS attack.

On the other hand, these security mechanisms rely mainly on cryptographic operations that use the same key. Consequently, bypassing this mechanism will allow an attacker to circumvent the others. This breaks the in-depth security principal.

Finally, we can notice that Sybil and Broadcast attacks, two attacks specially tailored to target WirelessHART networks, are able to bypass its security mechanism. These attacks can have harmful consequences on the network functioning.

3.6 Security improvement proposals

From the study of WirelessHART security mechanisms, we can emphases the following weaknesses in its implementation:

- use of a shared network key in the hop-by-hop communication.
- implementation of sensitive features (i.e., Disconnect DLPDU) in the DLL.
- use of a weak synchronization mechanism. This mechanism is implemented in the DLL through the Ack DLPDU. This latter includes in its payload the difference between the packet arrival expected arrival time. The sender node updates then its clock according the received value. As the Ack DLPDU is secured by the network key, an inside attacker can forge false Ack DLPDU in order to cause a cascading desynchronization of the network.
- use of a shared broadcast session key in broadcast end-to-end communication.
- implementation of a weak authentication mechanism based in the DLL on the network key and in the NL on the broadcast session key.
- lack of indication on cryptographic keys change. Indeed, although WirelessHART implements all necessary commands to renew cryptographic keys, both in the DLL and NL levels, it does not provide any recommendation on the key renew periodicity.
- lack of accountability mechanism that checks and reports abnormal nodes actions.

- lack of routing information correlation in the Network Manager. Indeed, these information are sent periodically by nodes to the Network Manager that uses them to build scheduling and routing table. These information are not checked before their use and an attacker can fool the Network Manager by sending false information about its neighborhood and location in the network.

In summary, WirelessHART weaknesses mainly result from the implementation of shared cryptographic keys in both the DLL and the NL. As shown in previous Sections, these features create dangerous breaches in the communication scheme security. However, these features are essentials to ensure the well functioning of WirelessHART.

The use of a shared network key in the DLL, makes it easy for each node to discover other nodes located in its neighborhood and also to assess the quality of each communication link. These information are then used to establish routing tables and also to quickly reconFigure them in the case of nodes failure or communication perturbations.

On the other hand, the broadcast communication allows the Network Manager to conFigure all devices composing the wireless network by only sending a single packet. It avoids a costing time and resources process of sending a single packet to each device.

As it is complicated to remove these two features, we propose hereafter, some ideas to reduce the exposition of resulting vulnerabilities.

- Use of a dedicated pairwise cryptographic key to authenticate nodes in the DLL. This will provide a strong authentication mechanisms and mitigate large number of attacks such as sybil and broadcast attacks. However this solution have a high cost in terms of storage and processing capabilities as it requires to store a key for each neighbor. It will also complicate the use of broadcast packet such as *keep-alive* and *advertisement*.
- Validation of a broadcast packet after the reception of 2 identical packets: As WirelessHART builds a meshed network, best practices in industrial sensor networks recommends that each node has at least 2 or 3 parents. Consequently, each sensor will receive the broadcast packet more than once. Thus, according to this rule, each node must wait till the reception of the same packet from another of its parents before it executes and forwards it. Nodes located at one hop do not have to apply this rule as they receive the broadcast packet directly from the Network Manager. This countermeasure adds a latency in the transmission of broadcast packets and can, in some cases, block their forwarding.
- Cross validation of DLL and NL addresses: in the case of the bounced command injection attack (a broadcast attack in which the malicious node misleads

its parent node by making this latter authenticates it as its own parent node [Bayou et al. 2016b]), DLL and NL headers of the injected packet indicate contradictory information. Indeed, the source address in the DLL header indicates that the packet has been sent by a children node i.e., the malicious node, while the source address in the NL header indicates that the packet has been sent by a parent node i.e., the network manager. Therefore, implementing in the NL a security mechanism that rejects packets indicating such contradictory information can mitigate this kind of attacks. We must note that even if this solution do not complain with the layer separation principle, in practice WirelessHART layers already use information provided by other layers such as addresses.

- Use of an IDS for monitoring node's behavior: indeed, except rethinking deeply the communication scheme of WirelessHART, as implementing asymmetric cryptography for packet's authentication, that is a costly process, the use of an IDS will increase significantly the security of such networks. Indeed, this kind of system by monitoring exchanged packets, are able to detect the injection of a false packet or the modification of a packet during its transmission.

In conclusion, the use of an IDS is the more efficient solution as other solutions either are partials and do not prevent all attacks, or are complex and add a big network overhead. Furthermore, although an IDS solution requires the installation of dedicated equipments for traffic monitoring, it is the only solution that detects all possible scenarios.

For this purpose, several studies have been conducted to propose IDS for WSN [Mitchell and Chen 2013, Mitchell and Chen 2014, Abduvaliyev et al. 2013] or more specifically for WirelessHART networks [Roosta et al. 2008, Bayou et al. 2017a]. Furthermore, given that WSNs are distributed systems, we must pay attention to the scheme used to deploy the IDS as it directly impacts its information gathering capabilities [Bayou et al. 2016a].

3.7 Conclusion

In this Chapter, we have analyzed security mechanisms implemented by WirelessHART, the most widely used wireless protocol in SCADA systems. We have given a description of each of them and have emphasized their strengths and weaknesses. We have shown that although, it implements several mechanisms to ensure security requirements in terms of authentication, availability, confidentiality, and non-repudiation, it

remains vulnerable to a large kind of attacks. This results mainly from the use of shared cryptographic keys known by all nodes that belong to the network.

On the other hand, proposed solutions do not totally prevent all possible attacks. Thus, except modifying deeply the communication scheme implemented by WirelessHART, the use of an Intrusion Detection System (IDS) is the best operational manner to detect and prevent attacks.

CHAPTER 4

WirelessHART security issues

4.1 Introduction

We have shown in Chapter 3 that although WirelessHART protocol implements several mechanisms to ensure the integrity and confidentiality of exchanged data, it remains vulnerable to a large kind of attacks. This results mainly from the use of shared cryptographic keys for securing communications.

On the base of this weakness, we present in this chapter two attacks against WirelessHART: a sybil attack which can isolate a large number of sensors from the network and the broadcast attack which allows an insider attacker to inject false commands into the network.

In order to prove the feasibility of these attacks and to assess their potential impact on the functioning of the industrial process, we first implement a simulator dedicated for the security studies of the WirelessHART protocol.

Thus, in Section 4.2 we present a literature review of related work. The implemented simulator is presented in Section 4.3. Then, we detail the Sybil attack in Section 4.4 and the Broadcast attack in Section 4.5.

4.2 Related Work

We find in the literature that there are few numbers of studies dedicated to WirelessHART and even less to its security.

In [Raza et al. 2009] Raza et al. state that Sybil attacks are almost impossible in this kind of networks. For Alcazar and Lopez [Alcaraz and Lopez 2010], Sybil attacks are hardly ever launched in WirelessHART since it offers strong authentication capabilities

before and after deployment. However, they recommend to add a rekeying process to WirelessHART to enforce its resilience to sniffing attacks and thereby key disclosure. We must also note that these studies are based on the specifications of the standard without conducting any tests.

Roosta and al. describe in [Roosta et al. 2008] a model-based intrusion detection system for WirelessHART sensor networks. This IDS models normal behavior of the different wireless devices and detects attacks when there is a deviation from the model. However, this kind of IDS can be bypassed with attacks based on the use of features deviated from their initial use.

On the other hand, there are in the literature, two categories of available WirelessHART simulators, partial [Biasi et al. 2008, De Dominicis et al. 2009, Nobre et al. 2010] and full [Zand et al. 2014] protocol stack implementation.

Partial implementations are mostly developed to study WirelessHART performances and its ability to be used in industrial environment. Therefore, only the Data Link Layer and basic Network layer are generally implemented.

De Biasi and al. developed in [Biasi et al. 2008] a WirelessHART simulator to address the problem of clock drift in a WirelessHART network. This problem occurs when no synchronization exists between two devices which causes packet losses. The proposed simulator uses TrueTime (a Matlab/Simulink-based environment) and implements only some features particularly in the MAC layer.

In [De Dominicis et al. 2009], De Diminicis et al. investigate coexistence issues when a WirelessHART network and another Wireless Network (WirelessHART, Wi-Fi or IEEE 802.15.4) are in the same radio coverage area. They implement their simulator on OMNet++, an open source simulation environment. As this study focused on interferences and aim to determine optimal network setup parameters for each kind of network, the simulator only implements the physical and the MAC layer.

Nobre and al. in [Nobre et al. 2010] develop a module for the NS-3 simulator. They focus on the implementation of the Physical layer in order to use it as the basis for the development of the other layers such as MAC and Application.

We find that there is only one full implementation. Indeed, Zand and al. propose in [Zand et al. 2014] an implementation of the Network Manager as well as the whole WirelessHART stack. The simulator is developed on NS-2 and validated using sniffed traffic from a real test-bed. However, this implementation does not focus on security aspects of WirelessHART as it is developed to assess performances and to test several routing and scheduling algorithms.

We choose to develop our own simulator, primarily for getting a complete implementation (including network devices and the Network Manager) and also to have an adapted environment to conduct studies on WirelessHART security mechanisms. It also permits to built basic and complex attack scenarios. Therefore, we use OMNet++ which is more flexible and easy handling than other simulation frameworks.

4.3 WirelessHART NetSIM: a WirelessHART SCADA-Based simulator

4.3.1 Introduction

Although, the security of SCADA systems is a major industrials concern, it is difficult to conduct any security analysis on a working SCADA system. Indeed, these systems are expected to work without any interruption for several years. Thus, conducting studies in real facilities is practically impossible. Consequently and in the absence of real test-bed specially deployed for research needs, using simulation is the best way to analyze SCADA systems.

This is more true for security analysis. Indeed, a simulator allows conducting deep tests and having an accurate assessment of existing security mechanisms. It also allows testing several attack scenarios and counter-measures which can be evaluated and validated easily. Another advantage is that we can evaluate the impact of the proposed security mechanisms on the system's operations, for instance in terms of availability and real-time requirement. Such simulator must be enough flexible to permit the elaboration of different scenarios in an easy way.

In this section, we present WirelessHART NetSim a simulator for WirelessHART SCADA-based systems. The proposed simulator fully implements the WirelessHART stack and both field devices and the Network Manager including routing and scheduling algorithms. It is based on OMNet++ [OMNeT++], a discrete event simulator based on C++ language. Our simulator includes scenarios for testing several kinds of attacks such as sybil and denial of service (DoS) attacks. It can be easily extended in order to test other kinds of attacks.

4.3.2 WirelessHART NetSim implementation overview

We use OMNeT++ [OMNeT++] for the implementation of the WirelessHART NetSim simulator. OMNeT++ is a discrete event, extensible, modular, C++ based simulation library and framework, for building network simulators. It includes extensions for real-time simulation, network emulation, database integration, and several other functions. We also use INETMANET [InetManet] a fork of the INET Framework, which adds a number of experimental features and protocols, mainly for mobile ad-hoc networks.

In the following, we present the implementation of the WirelessHART NetSim simulator:

Physical Layer implementation

The OMNet++ extension InetManet [InetManet] provides the implementation of several wireless protocols, including the full stack of the IEEE 802.15.4 protocol [IEEE 802.15.4-2006]. Therefore and as WirelessHART physical layer is based on the one of IEEE 802.15.4, we use the implementation provided by InetManet as an implementation of the Physical Layer of our simulator.

Data Link Layer implementation

We modify the implementation of the MAC layer of the IEEE 802.15.4 protocol provided in InetManet to support TDMA, channel hopping, slot communication, and modify the Data frame format. We also add communication tables used in DLL and their relationship as neighbor table, link table, graph table, and superframe table.

We use a *timer* to simulate the start of a slot. Thus, each 10ms the node wakes up and identifies the current slot. On the base of information in Link Table, an indication is sent to the Physical layer to put the transceiver in reception or transmission mode. In transmission mode, the appropriate DLPDU is selected from the buffer based on the destination address.

Network Layer implementation

In the Network Layer, we implement graph routing support and session mechanism to ensure end-to-end reliability as retry. We merge it with the Transport Layer to ensure

end-to-end acknowledgment and assembly/fragmentation. We also add routing table and correspondant table (i.e. table indicating which route to use to reach a node).

The Network Layer is responsible for routing the packet sent by the Application Layer. The address of the next hop is recovered from the routing table using the final destination address.

When a packet is received from the DLL, the Network Layer checks the destination address field. The packet is either passed to the Application Layer or sent again to the Data link Layer to be forwarded to the next hop.

Transport Layer implementation

In our implementation, we choose to merge the Transport Layer into the Network Layer. We implement only acknowledgment and retry mechanisms.

Application Layer implementation

The Application Layer of WirelessHART is a command based layer. They are used by the Network Manager to configure nodes with routing and scheduling information. We implement only necessary commands which can be classified into several categories: managing routing and graphs, managing superframes and links, and network health report commands.

We choose to implement the Network Manager, the Gateway and the Security Manager as the same entity. The Network Manager is based on the implementation of WirelessHART device. We add at its Application Layer several management algorithms for routing and communication scheduling. Each time a new device joins the network, these algorithms are executed to:

- provision the joining device with necessary credential (nickname and keys),
- create a downlink graph from the Network Manager to the joining device,
- allocate communication scheduling,
- and to update parent's tables with routing and scheduling information.

In the Network Manager's Application layer, we implement routing and scheduling algorithms. The routing algorithm creates for each node two graphs (an uplink graph from the node to the Network Manager and a downlink graph from the Network

Manager to the nodes). Based on these graphs, the Network Manager builds routing tables using the Dijkstra algorithm. The link weight on these graphs is function of the Received Signal Strength Indication (RSSI) between both vertices of the link.

The scheduling algorithm allocates a sending slot (and another one for retry) from each node to the Network Manager following the uplink graph in order to transmit the sensing data. It also allocates a sending slot from the Network Manager to each node following the downlink graph for command request and another one following the uplink for command response.

4.3.3 WirelessHART NetSim procedures implementation

After implementing the WirelessHART stack, the Network Manager and the field devices, we implemented some WirelessHART procedures. These procedures include joining process, advertisement, neighbor discovery and disconnect. A procedure is a set of actions leading to execute an exchange sequence. Some of them such as joining are executed at different layers.

An advertisement is an invitation sent by a device which is already part of the wireless network to new devices wanting to join the network. It contains needed information as current ASN, Join links which are slots in which a new device can send a join request.

The joining process, illustrated in Figure 4.1, is an exchange sequence between a device wanting to join the wireless network and the Network Manager. It also includes a proxy device which is a device acting as the parent of the new device during this procedure. During this process, the new device, previously configured with a Join Key (used as a session key), sends a join request to the Network Manager through the proxy device. The Network Manager after checking the request responds to the device by sending a nickname and session keys. The Network Manager will also create a downstream to the new device and configure all devices belonging to it. Finally, communication schedule will be allocated and transmitted to the new device and its parents. At the end of this procedure, the new device is entirely integrated in the network and starts to send sensing data to the Network Manager.

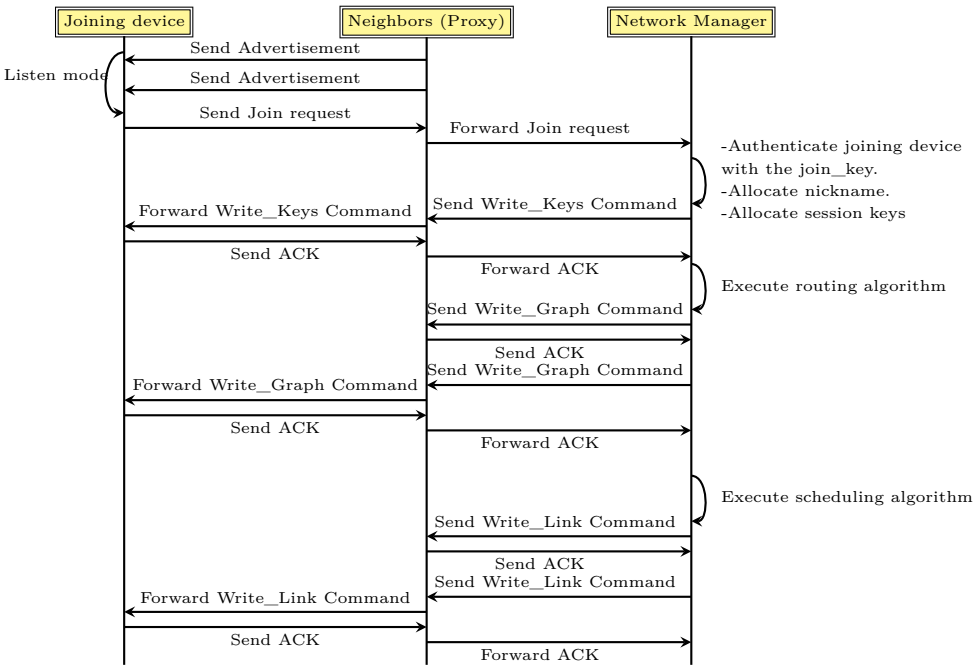


Figure 4.1: Joining message exchange sequence

4.4 Sybil attack in WirelessHART Network

4.4.1 Introduction

In this Section, we give the first description of a Sybil attack specially tailored to target a WirelessHART network. This attack can cause harmful damages to the facility by disconnecting partially or entirely the wireless sensors from the SCADA system. Conducted against a real facilities, such attack can disturb deeply its functioning and can lead to stop it or more again induce its destruction.

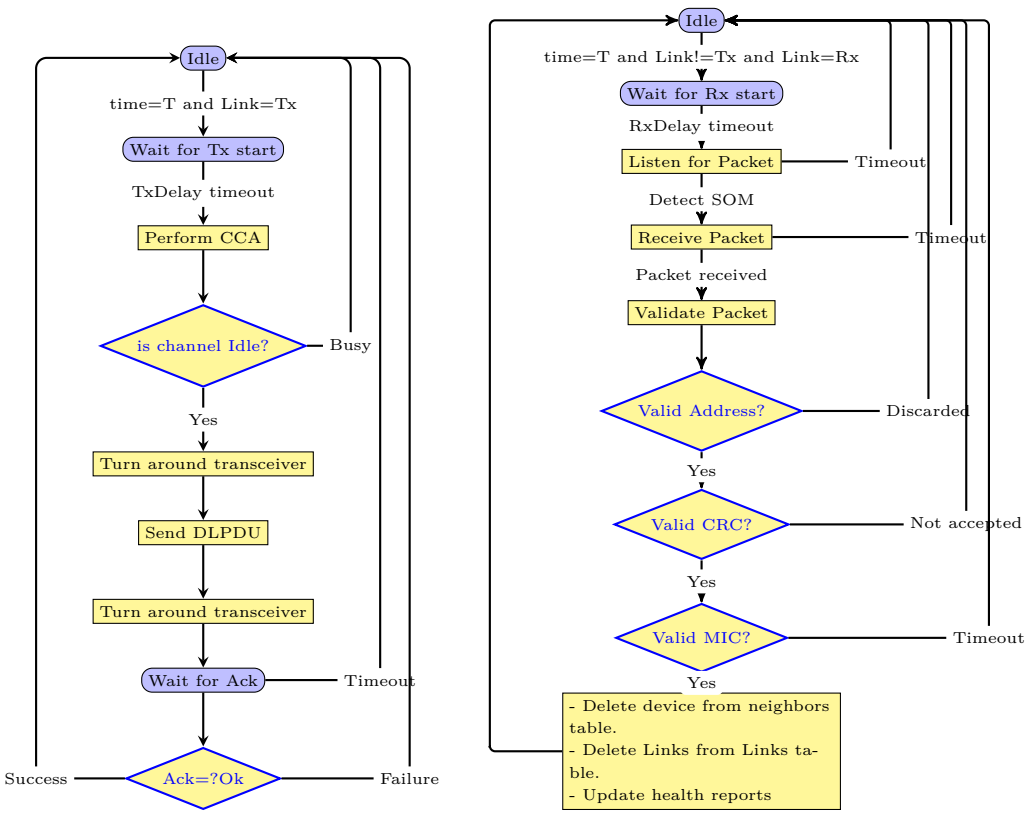
Sybil attack was first described by Douceur in [Douceur 2002]. He shows that in the absence of a central identification authority that checks correspondence between entity and identity, a malicious entity can present multiple identities. Sybil attack was initially described for peer-to-peer networks, however it can be applied to any network's type. Karlof and Wagner point out in [Karlof and Wagner 2003] that sybil attacks can be used against routing algorithms in sensor networks to reduce significantly the effectiveness of redundancy schemes. In [Newsome et al. 2004] Newsome and al. analyze sybil attacks in sensor networks and establish a classification of different forms of attacks (Direct vs Indirect communications, Fabricated vs Stolen identities, Simultaneous vs Non-simultaneous). They also examine how it can be used to attack several types of

protocols in WSN such as distributed storage, routing, data aggregation, voting, fair resource allocation and misbehavior detection algorithms.

In the following, we describe the Sybil Disconnect Attack. Then, we analyze its threats to the Wireless Sensor Networks. Finally, we present and evaluate a solution to mitigate this kind of attacks.

4.4.2 Disconnect DLPDU

According to WirelessHART standard [HART Communication Foundation] a device can either be disconnected by the Network Manager or disconnect itself or simply die. In the first case, the Network Manager sends a disconnect command (960) to the device, whereas in the second case the device sends a Disconnect DLPDU to inform its neighbors that it is leaving the network. This DLPDU is originated in the data link layer and secured by the Network Key. It is transmitted in the first available link as shown in Figure 4.2(a).



(a) Disconnect DLPDU transmission by device

(b) Disconnect DLPDU reception by neighbors

Figure 4.2: Disconnect DLPDU processing

When a Disconnect DLPDU is received by a device, it removes the sending device from its neighbor list, and deletes all links connecting to the sending device (see Figure 4.2(b)). Also, the neighbors indirectly inform the network manager with health reports (i.e, periodic statistics transmitted to the Network Manager by each device about its neighbors) about the device disconnection. The Network Manager updates device's routing tables to forward packets through other routes and reallocates disconnected device's resources (ex.: slots). By that, the disconnected device has not anymore any allocated resources and shall go through a complete rejoin sequence.The overall message exchange is summarized in Figure 4.3.

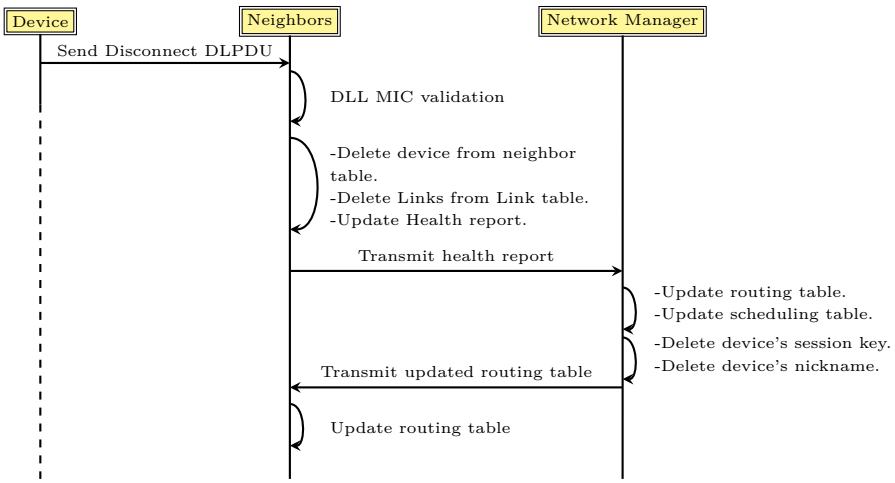


Figure 4.3: Disconnect message exchange sequence

4.4.3 Disconnect Sybil attack

As described in Section 3.3.5, in WirelessHART communication security is ensured by two cryptographic keys. The Network Key which defends against outsider attacks and the Session Key which defends against insider attacks. The use of these keys, aim to provide an in-deep defense against wireless security threats. We describe here a harmful attack requiring only the known of the Network Key and using disconnect DLPDU.

A disconnect attack is a sybil attack in which an attacker spoof the identity of a legitimate device by forging fake Disconnect DLPDU and setting the source address to the target device's address. As a result the target device will be disconnected from the network since its device neighbors will remove it from their tables. This attack is based on the fact that the disconnect DLPDU is originated in the data link layer and all devices in the network share the same key (Network Key) for generating and validating the Message Integrity Code (MIC) in the DLL.

To perform a Disconnect Sybil Attack, an attacker needs to collect some information about the targeted device: the short and long address of the device; and mainly find a slot to send out the disconnect packet. We assume that the attacker knows the Network Key but not the session key of the targeted device.

4.4.4 Collecting information about the target device

In order to gather needed information about the targeted device, the attacker should listen awhile to the packets forwarded throughout the network. By this, it will obtain needed information such as the short and the long address of the targeted device and also synchronize itself with the network current time. Indeed, these information are not enciphered in exchanged messages.

WirelessHART uses Time Division Multiple Access (TDMA) and Channel hopping to control access to the medium. Each communication between two devices occurs in one slot of 10 ms. Only one packet is transmitted in one slot from the sender to the receiver which has to reply with an acknowledgment packet in the same slot. The 2.4 GHz band is divided into 16 channels numbered from 11 to 26 (channel 26 is not used). So, each slot is identified by a number called Absolute Slot Number (ASN) indicating the count of slots elapsed since the start of the network and a channel offset. As all communication occur in predefined slots established by Network Manager, attacker need to find the right slot where it can send out the Disconnect packet to the target neighbors. This can be done in several way, such as:

- if the attacker is a legitimate device, it will receive its own schedule from the Network Manager and by that it will know if the targeted device will perform a broadcast transmission and at which frequency.
- the attacker can use the retry slot to send out the disconnect packet to target's neighbors one by one. Indeed, the Network Manager when allocating normal slot for data transmission, also allocates a retry slot. This slot is used only when the transmission in the normal one failed. Otherwise, it will not be used.
- in our scenario we send the disconnect DLPDU to the parent of the target device using the join link dedicated to the reception of the join request from new devices. Information about this link are periodically transmitted in the Advertisement DLPDU of the target device parent.

4.4.5 Sybil attack implementation

For validating our attack, we use the WirelessHART NetSim simulator. We implement an entirely automated sybil attack in which a legitimate device usurps the identity of another device by forging a fake Disconnect DLPDU and setting the nickname (short address) of the target device as the source address of the forged DLPDU.

The implementation of the malicious device is based on the implementation of a WirelessHART device. Initially, the malicious device acts as a normal device. When triggered, it enters a search mode in which it waits for getting an Advertisement from the parent of the targeted device. When done, it will use the join link of the parent device to send to it the forged Disconnect DLPDU. At the reception of this DLPDU, the parent device validates it with the Network Key and processes it by removing the sending device from its neighbors table and also all links related to this device. By so, the targeted device is automatically disconnected from the network since it has not anymore any connection with its parent and has to go through the entire join procedure. The attack is summarized in Figure 4.4.

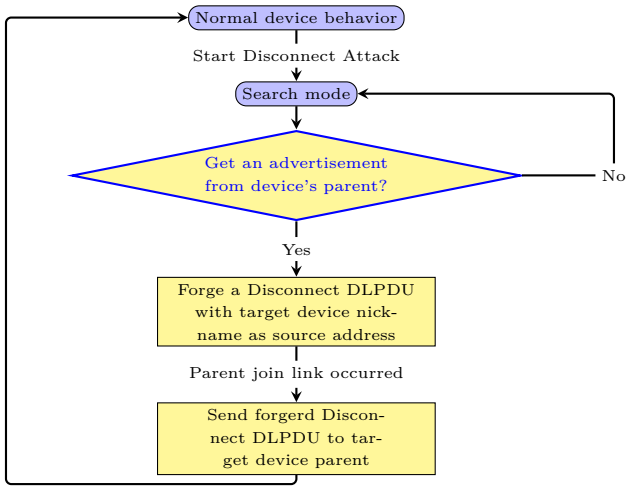
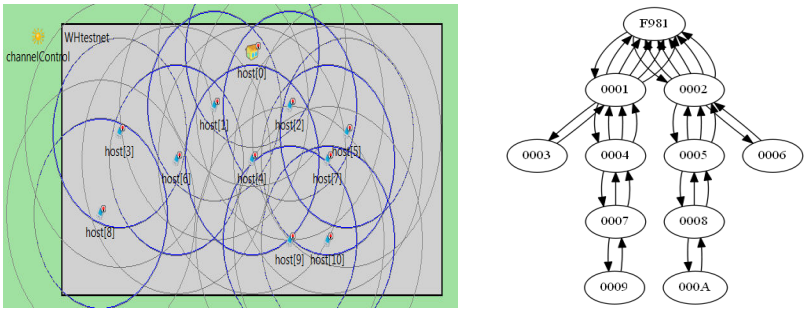


Figure 4.4: Sybil Disconnect Attack

4.4.6 Sybil Attack threats analysis

To conduct our simulation, we build a wireless network composed of one Network Manager and ten wireless devices as shown in Figure 4.5(a). We start the simulation by initiating the network: the Network Manager begins to send Advertisement DLPDU and wireless devices enter joining procedure. Each time a new device joins the network, it will start to send sensing data at a periodic time of 4s and advertisement DLPDU.

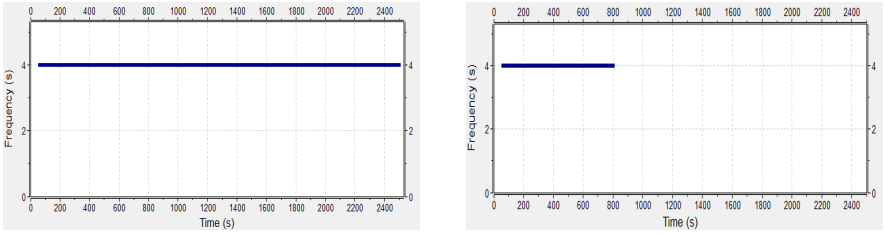
Figure 4.5(b) illustrates the global topology of the wireless network as seen by the Network Manager.



(a) Physical topology (b) Logical topology

Figure 4.5: Simulation network topology

We restart the simulation and we launch the Sybil Disconnect Attack at T=800s. The device with nickname 0x0003 is configured to be the "malicious" device and the device with nickname 0x0004 will be the "target" device. The "parent" device will be the device with nickname 0x0001. According to Figure 4.6(a), in normal case the Network Manager receives sensing data from target device at a fixed frequency of 4s. In 4.6(b) we can see that just after the attack was launched, the Network Manager stops to receive data from target node. Figure 4.7(b) shows that the data send success rate for target node falls quickly from 100% to 0% immediately after the attack was conducted.

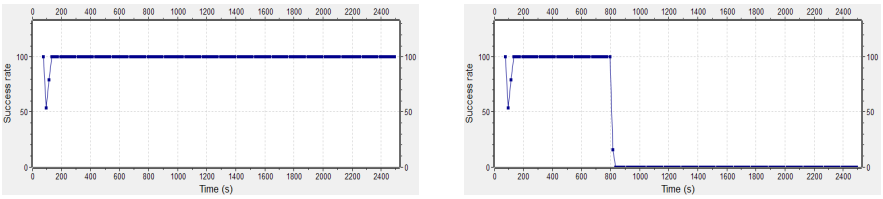


(a) Without attack (b) With attack

Figure 4.6: Data sensing time arrival to Network Manager frequency from target device

Comparatively to Figure 4.7(a) in 4.7(b) we can see clearly that the target device is completely disconnected from the network and even if it continues to try to send its own packets or to forward packets received from its children devices, the success rate is 0%. So by disconnecting a device we disconnect also its children (devices 0x0007 and 0x0009 in this case).

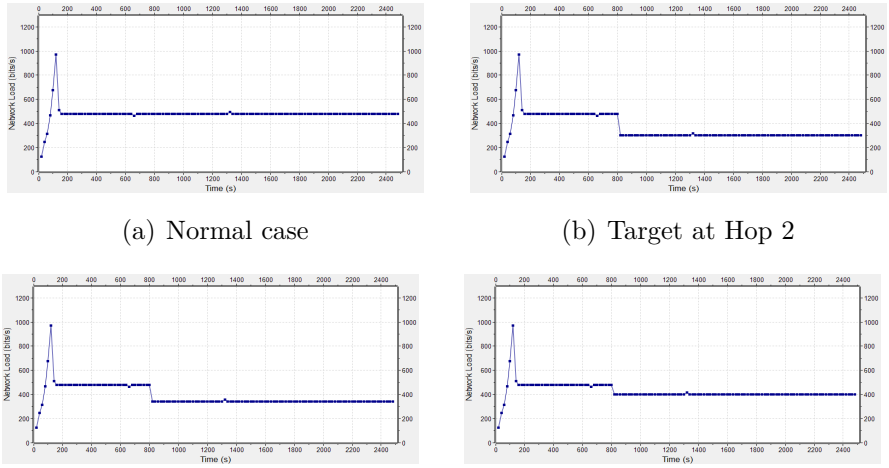
In Figure 4.8 we variate the position of the target device to show the impact of the Disconnect Attack on the network charge. We can see that the decrease of the network load is directly correlated with the number of hops to reach the Network Manager and



(a) Without attack (b) With attack

Figure 4.7: Data Send success rate for target node

the number of its children. Consequently, an attack on a device situated at two hops (see Figure 4.8(b)) decreases the network load by almost 37%, an attack against a device situated at three hops (see Figure 4.8(c)) decreases the network charge by 29% and an attack against a device at four hops decrease it by 16% (see Figure 4.8(d)). As expected, more the target device is near the Network Manager more the impact of the attack on the network load is significant. Indeed, this is due to the number of its children.



(a) Normal case (b) Target at Hop 2

(c) Target at Hop 3 (d) Target at Hop 4

Figure 4.8: Network load in byte "before and after attack"

Table 4.1 indicates the time spent by a malicious node in search mode and also the total attack duration from the beginning of the search mode to the sent of the Disconnect DLPDU. The duration of the attack depends on the size of the management superframe (used for commands transmission and reception). In our simulation, it is set to 200 slots (2s) and the advertisement DLPDU sending frequency is set to 4s.

In the general case, T_{adv} the time to get an advertisement depends on its frequency sending: $MAX(T_{adv}) = 15 \times F_{sending\_adv}$ where $F_{sending\_adv}$ is the frequency of sending advertisement and 15 is the number of used channels. When the attacker gets an

Target	Search mode duration (s)			Total attack duration (s)		
	Avg	Min	Max	Avg	Min	Max
Device 0004	2	0.01	3.99	4.58	2.01	5.96

Table 4.1: Attack duration

advertisement DLPDU from the parent, it must wait at worst the duration of the superframe to send the forged disconnect DLPDU. So, $MAX(T_{total\_attack}) = 15 \times F_{sending\_adv} + S$ where S is the size of the superframe. In our case, we get $MAX(T_{adv}) = 4s$ (Advertisement are sent on channel 11) and $MAX(T_{total\_attack}) = 4 + 2 = 6s$

4.4.7 Proposed solution

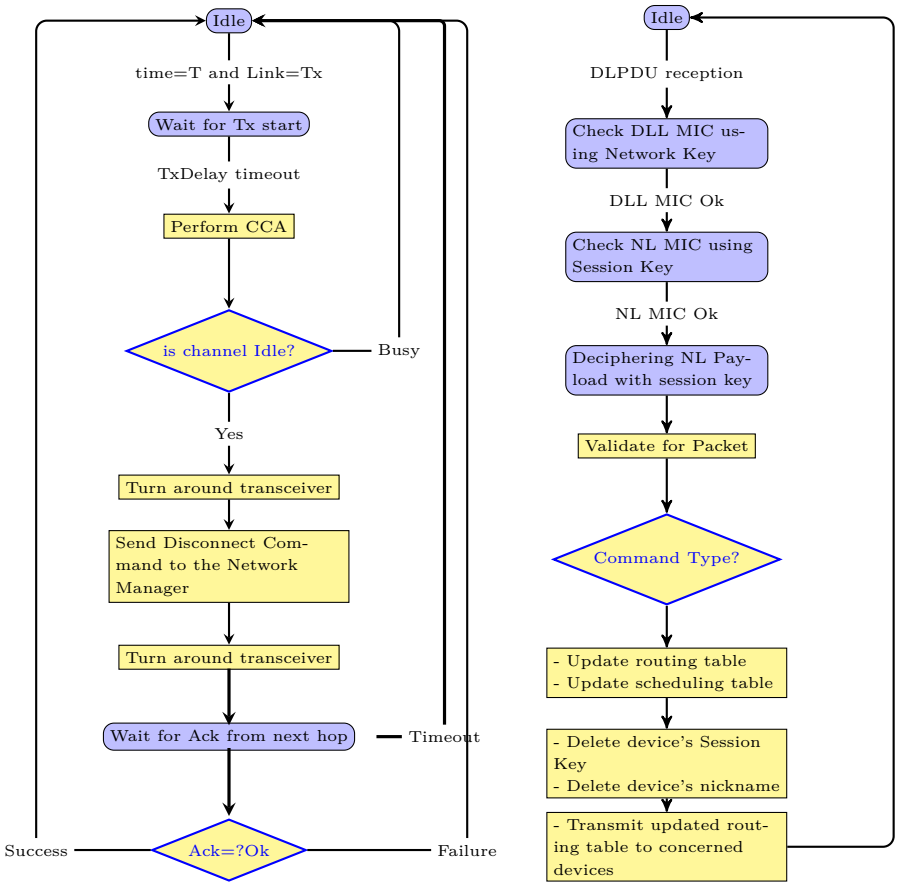
Disconnect attack uses two security weaknesses in the WirlessHART protocol: the implementation of a critical feature in the Data Link Layer, i.e. Disconnect DLPDU. Probably, the Disconnect DLPDU is a feature inherited from the IEEE 802.15.4-2006 standard [IEEE 802.15.4-2006]; and the use of a shared secret key in the Data Link Layer.

The combination of these two weaknesses can lead to harmful consequences on network behavior: disturbing routing protocol, isolating a group of nodes, etc.

In order to mitigate such attacks, we should prohibit the use of critical features in DLL and move them to the application layer. Indeed, AL Commands are secured by a Session key known only by the field device and the Network Manager.

As illustrated in Figure 4.9(a), a field device sends a Disconnect Command instead of a Disconnect DLPDU. The Command is forwarded through the network to the Network Manager. Figure 4.9(b) shows actions executed by the Network Manager when receiving a Disconnect Command. It deciphers the network layer payload using the session key and after authenticating the sender, it updates routing tables and reallocates sending resources. In that way, the attacker will not be able to spoof the identity of any other device as it does not know the secret key shared by both of them. The overall message exchange is summarized in Figure 4.10.

We analyze the impact of the solution on the overall functioning of the network. For that, we analyze two parameters: the network overload and time elapsing between the disconnection of a device and when the Network Manager is informed of it.



(a) Modified Disconnect DLPDU Command by device

(b) Modified Disconnect Command reception by NM

Figure 4.9: Modified Disconnect Command processing

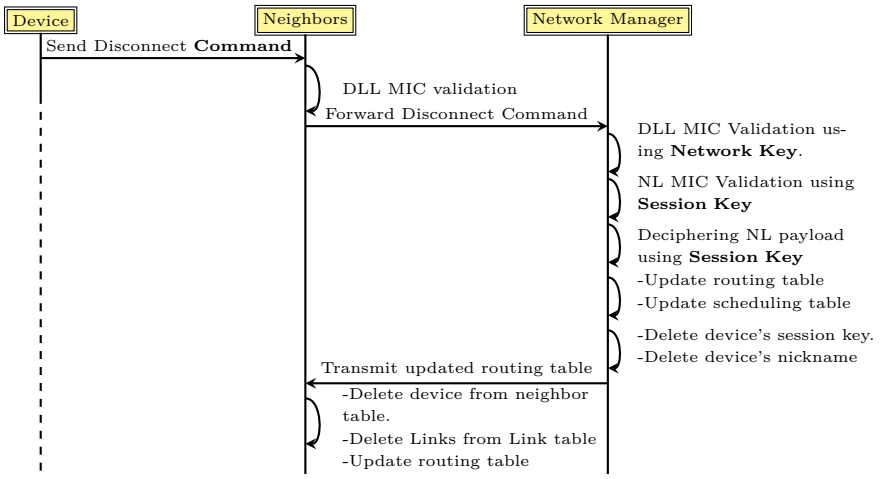


Figure 4.10: Modified Disconnect message exchange sequence

In the case of the disconnect DLPDU the disconnecting device sends one packet of 22 bytes to its neighbors. When send_health_report_timer (set by default to 15

Number of hops	**Disconnect DLPDU (Bytes)**	**Disconnect Command (Bytes)**
2	174	72
3	326	144
4	478	216

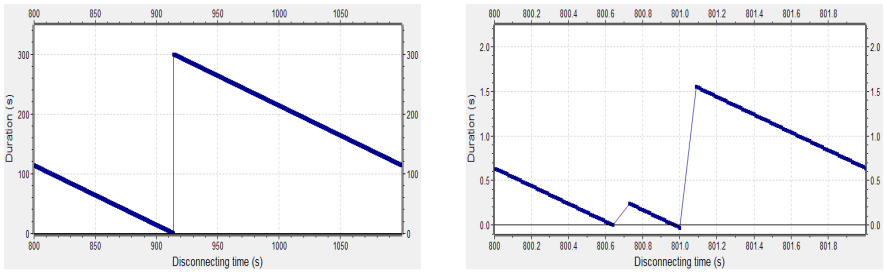
Table 4.2: Network overload by number of hops.

minutes) elapsed each neighbor will report to Network Manager the list of devices present in its neighborhood. By this, the Network Manager will deduce that a device has disconnected. Health reports are application level commands encapsulated in packets of 127 bytes and forwarded to the Network Manager hop-by-hop. The cost of the transmission is : $cost = 22 + (N - 1) \times (127 + 25)$ where 25 bytes is the size of the acknowledgment DLPDU and N is the number of hops from disconnecting device to the Network Manager.

However, if the disconnecting device has more than one parent the Network Manager needs to know about the device disconnection to wait till it receives all health reports from each parent. So then, the total cost is $cost = 22 + \sum_{i=1}^{M}(N_i) \times (127 + 25)$ where M is the number of neighbor devices that disconnecting device is not the parent and N_i is the number of hops from each device to the Network Manager.

For the case of the use of a disconnect Command, a packet of 47 bytes is sent from the disconnecting device to its parent neighbor and then forwarded hop-by-hop to the Network Manager. $cost = N \times (47+25)$ where 25 bytes is the size of the acknowledgment DLPDU.

For the second parameter, we variate the time when a device disconnects and we report time elapsed before the Network Manager is informed about the disconnection. Figure 4.11 illustrates that the time elapsed before the Network Manager is informed about the disconnection of a device in the case of the use of the disconnect DLPDU (Figure 4.11(a)) is significantly bigger than the one elapsed in the case of the use of the disconnect Command (Figure 4.11(b)). Indeed, disconnect Command is forwarded directly, hop-by-hop, each time a slot is available from the disconnecting device to the Network Manager. In the other case the information is transmitted to the Network Manager by the neighbors of the disconnecting device in their health reports which sending frequency is set by default to 15 minutes. Therefore, the average time in the case of using disconnect DLPDU will be 7,5 minutes. In our simulation we set this frequency to 5 minutes (300s) and even with this three times high sending frequency of health reports we get big values (Max=300s, Avg=150s) comparatively to the case of using disconnect command (Max=1.5s, Avg=0.67s).



(a) Disconnect DLPDU (b) Disconnect Command

Figure 4.11: Time duration before the NM is informed about a device disconnection

4.4.8 Conclusion

We describe in this Section a serious security issue in WirelessHART. We demonstrate that an insider attacker can cause harmful disturbance to the network. We also give a fully-automated way to take advantage of this weakness to isolate partially or more again totally the wireless sensors from the SCADA network. The conducted tests confirm the feasibility of this attack and its dangerous potentiality. They demonstrate that this attack is easily conducted and does not require any additional means. Moreover the time to collect needed information to launch the attack is quite short.

4.5 Broadcast attack in WirelessHART Network

4.5.1 Introduction

We show in this Section, that although the WirelessHART protocol implements several security mechanisms, an inside attacker can use his own credential to bypass them and inject false commands in the network. Using this weakness, we describe three scenarios that can be used to launch an attack against a WSN.

In the following, we describe WirelessHART communication scheme. Then, we detailed the broadcast attack and present several ways to perform it. We also demonstrate the harmful impact of each of these scenarios. Finally, we propose countermeasures and discuss their ability to mitigate this kind of attack.

4.5.2 Communication scheme

WirelessHART implements unicast and broadcast communications in both the Data Link and the Network Layers. In the Data link layer, the unicast or broadcast com-

munication is set by configuring the packet with unicast or a broadcast destination address, by using the unicast or the broadcast graph and also by using the dedicated transmission slots. Indeed, the Network Manager configures each wireless sensor to be at the beginning of each slot either a sender, a receiver or to stay idle.

As illustrated in Figure 4.12, when a device receives unicast packet, it starts by authenticating it in the Data link layer (DLL) using the network key and then it is transmitted to the Network layer. There, the destination NL address is checked. If it matches the device's address, the packet is authenticated a second time using the unicast session key and its payload is deciphered and sent to the Application Layer to be executed. Otherwise, the packet is sent back to the DLL to be forwarded to the next hop device.

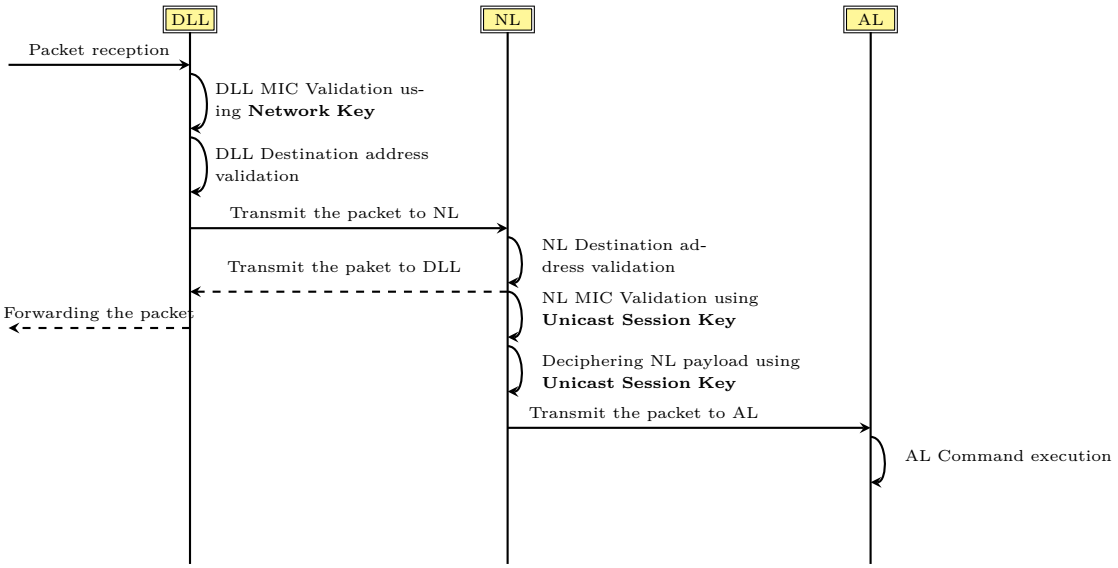


Figure 4.12: Unicast packet processing sequence

In a broadcast communication, a packet sent by the Network manager is propagated to all devices in the wireless network. As illustrated in Figure 4.13, each time a device receives a broadcast packet, it starts by authenticating it firstly in the Data link layer (DLL) using the network key and then in the Network layer (NL) using the broadcast session key. If the packet passes authentication validations, it will be deciphered and sent to the Application Layer (AL) to be executed. A copy of the packet is also sent back to the DLL to be forwarded to other devices.

On another hand, in the Network Layer, four sessions are set up as soon as any device joins the network. They allow the transmission of sensing data from a device to the Network Manager, and the transmission of commands from the Network Manager to a field device. These sessions are the following:

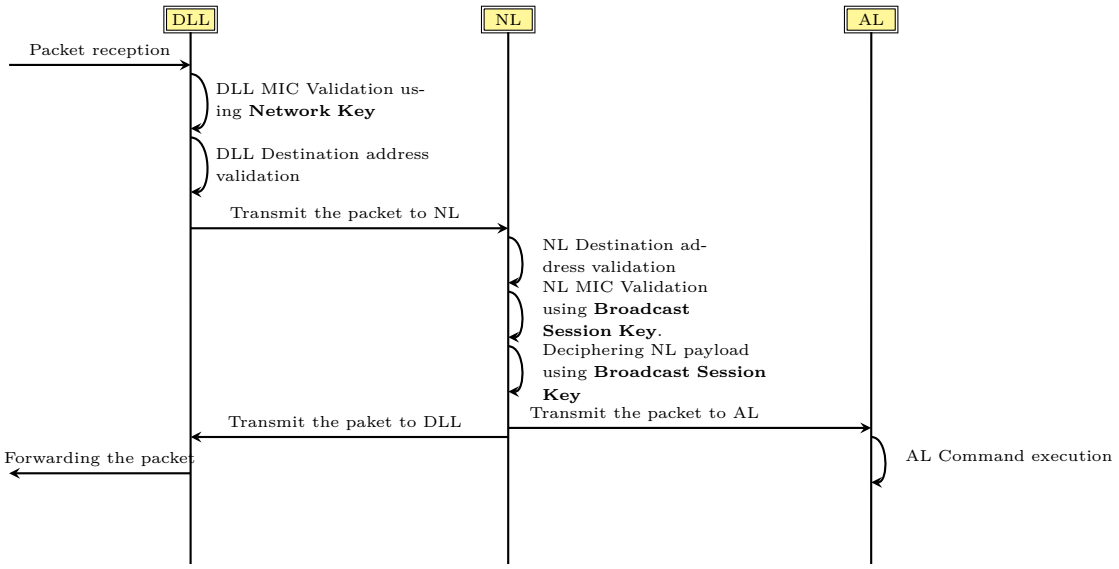


Figure 4.13: Broadcast packet processing sequence

1. unicast session with the NM: it is used by the Network Manager to manage the device.

2. broadcast session with the NM: it is used to globally manage devices. For example this can be used to roll a new network key out to all network devices. All devices in the network have the same key for this session.

3. unicast session with the Gateway: it carries normal communications (for example process data) between the gateway and the device.

4. broadcast session with the Gateway: it is used by the gateway to send the identical application data to all devices.

In addition, each device has a join session key which cannot be deleted. The *join_key* is the only key that is written once connecting to the device's maintenance port. It can also be updated by the Network Manager once the device is successfully connected. All other keys are distributed by the Network Manager.

4.5.3 Communication Scheme Attack

The idea of the attack is that a malicious inside attacker uses his own credentials to bypass the authentication mechanism and injects false command into the network. These false commands will be authenticated as legitimate commands and executed by

receiving devices. Depending on the nature of injected false commands, consequences on the network can be more or less harmful.

As indicated in the previous Section, end-to-end communications are secured by session keys. In unicast communications, the session key is only known by the two communicanting devices while in broadcast communications, the session key is shared by all devices connected to the network.

Therefore to launch the command injection attack, the malicious inside attacker will use Broadcast Session credentials to perform this kind of attacks. Indeed, as part of the network, the malicious node is configured with *the broadcast session key* and the *session counter*.

The command injection attack can be performed in several ways such as: a Direct command injection attack, a Bounced command injection attack and an On-the-fly command injection attack.

Scenario 1: Direct command injection attack

In a Direct command Injection attack a malicious insider node forges a fake broadcast packet and forwards it to its neighbors.

As illustrated in Figure 4.14, at the moment T the malicious node *Device*5 uses its knowledge on the broadcast session credential i.e., the broadcast session key and the session counter, to forge a broadcast packet. The source address in the NL is set to the Network Manager address and the destination addresses in both network and data link layers are set to the broadcast address. The malicious insider node will send the forged packet using its own broadcast link in the same way as if it was a legitimate packet sent by the network manager. Receiving nodes, *Device*8 and *Device*9, will authenticate the packet using the broadcast session key and execute the injected false command.

Using this attack, a malicious insider node can inject any false command and send it to its neighbors using the broadcast graph.

Scenario 2: Bounced Command injection attack

In WirelessHART both DLL and NL destination addresses can be either unicast or broadcast addresses and all combinations are allowed. So, a packet can have unicast DLL destination address and a broadcast NL destination address.

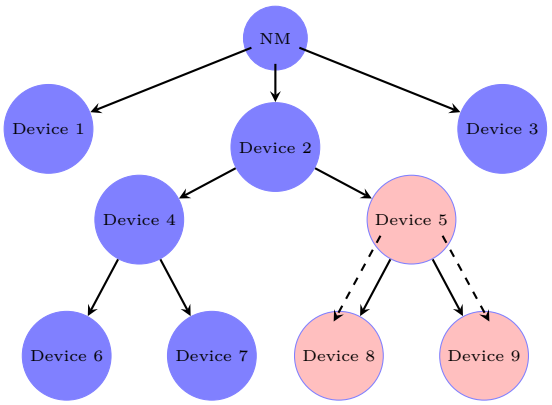


Figure 4.14: Direct Broadcast attack

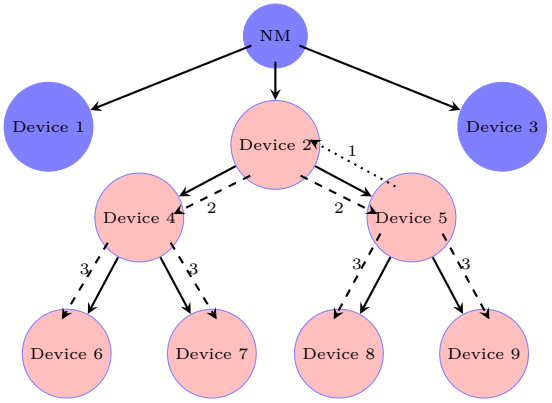


Figure 4.15: Bounced Broadcast attack

In a Bounced command Injection Attack a malicious insider node forges a fake broadcast packet and sends it to its parent node. As illustrated in Figure 4.15, this kind of attacks is composed of the following steps:

1. At the moment T the malicious node *Device*5 uses its knowledge of the broadcast session credential i.e., the broadcast session key and the session counter, to forge a broadcast packet. The source address in the NL is set to the Network Manager address and the NL destination address is set to the broadcast address.

   In the DLL, the source address is set to the *Device*5 address and the destination address is set to its parent's address i.e., *Device*2. The malicious insider node will send the forged packet using its own normal link between itself and the parent node.

2. The receiving node *Device*2 authenticates the packet in the DLL as a legitimate unicast packet and transmitted it to the upper layer.

In the NL, the packet is identified as a broadcast packet sent by the Network Manager. It is authenticated and deciphered using the broadcast session key. The packet is then transmitted to the application layer to be executed.

A copy of the packet is also transmitted to the DLL to be forwarded to *Device*2 neighbors i.e., *Device*4 and *Device*5.

3. Both *Device*4 and *Device*5 process the received packet as a legitimate broadcast packet sent by the Network manager and propagate it to their neighbors.
4. As results, the injected false command packet is received and executed by *Device*2, *Device*4, *Device*5, *Device*6, *Device*7, *Device*8 and *Device*9.

This scenario allows a malicious insider node by using its parent node as a relay to increase the impact of the attack. By this way, the injected false command is propagated to all parent node's children.

Scenario 3: On-the-fly Command injection attack

In an On-the-fly command injection attack, a malicious insider node that receives a broadcast packet, will forward to its neighbors a modified version of the received packet.

As illustrated in Figure 4.16, this attack is performed according to the following steps:

1. The Network Manager sends a broadcast packet.
2. The broadcast packet is forwarded to devices and received by the malicious insider node *Device*5.
3. All receiving nodes execute the command sent by the network manager and forward it to devices in their neighborhood.
4. The malicious node *Device*5 uses its knowledge of broadcast session credential i.e., the session key and the session counter, to modify the received broadcast packet and send it to its neighbors.
5. As results, the injected false command packet is received and executed by *Device*8 and *Device*9.

As in the direct command injection attack, a malicious insider node can inject any false command and send it to its neighbors using the broadcast graph. The difference is that an on-the-fly injection command attack is a stealth attack as the injected packet is hidden inside a legitimate communication flow.

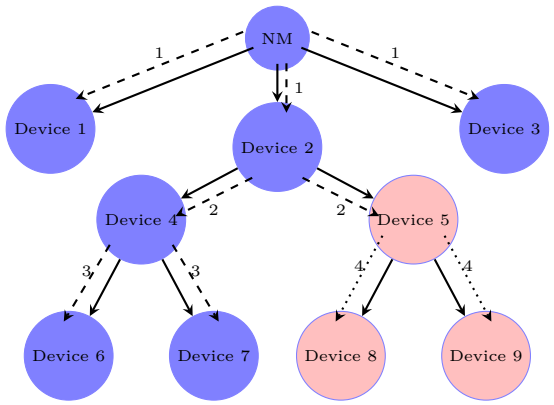


Figure 4.16: On-the-fly Broadcast attack

Discussion

Described scenarios showed the feasibility of the broadcast attack and that it can be performed in several ways. We must note that although we can launch the attack at any chosen time T, the malicious node must wait for an appropriate time slot to be able to send the forged packet. For example in the case of the direct command injection, the malicious node must wait for the next broadcast slot to send the false command to its neighbors. But as all devices are configured with this kind of slots, it is always possible for a malicious node to send its false command. According to the WirelessHART [HART Communication Foundation], by default each device is configured with one sending unicast slot and one sending broadcast slot each 1 minute. Thus, the average waiting time T_{Avg} between the attack launching time and the false command injection time is: $T_{Avg} = T_{sending\_broadcast}/2 = 30s$ in the case of a direct attack and $T_{Avg} = T_{sending\_unicast}/2 + T_{sending\_broadcast}/2 = 60s$ for a bounced attack. The on-the-fly attack duration depends on the industrial process and broadcast commands sending frequency. In average, this frequency is around 1 hour.

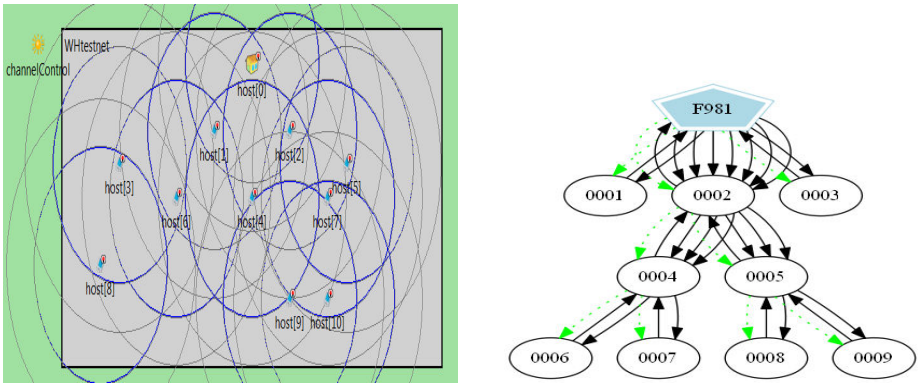
By comparing the 3 scenarios, we can see that the bounced command injection increases the spreading area of the attack by using the parent of the malicious node as a relay. Also, the on-the-fly command injection attack is interesting as it hides the attacks inside a legitimate flow. Nevertheless, the drawback of this attack is that the malicious node must wait to the transmission by the network manager of a broadcast packet which can take a long time to happen.

Finally, we must note that in all these scenarios, the malicious insider node has the choice between executing or not the injected false command. Indeed, depending on the attack's goal, the malicious node can launch the attack with or without executing

it. For example, by not executing the false command, the malicious node can mislead administrators in their investigations to discover the origin of the network disturbances.

4.5.4 Attack implementation

We use WirelessHART NetSIM to test and evaluate the 3 scenarios of the broadcast attack. Thus, as illustrated in Figure 4.17(a), the simulated wireless network is composed of a network manager and 9 wireless sensors. Wireless sensors are configured to send periodically each 4s simulated sensing data to the Network Manager. Figure 4.17(b) illustrates the routing graphs. The broadcast graph is indicated by dotted green arrows.



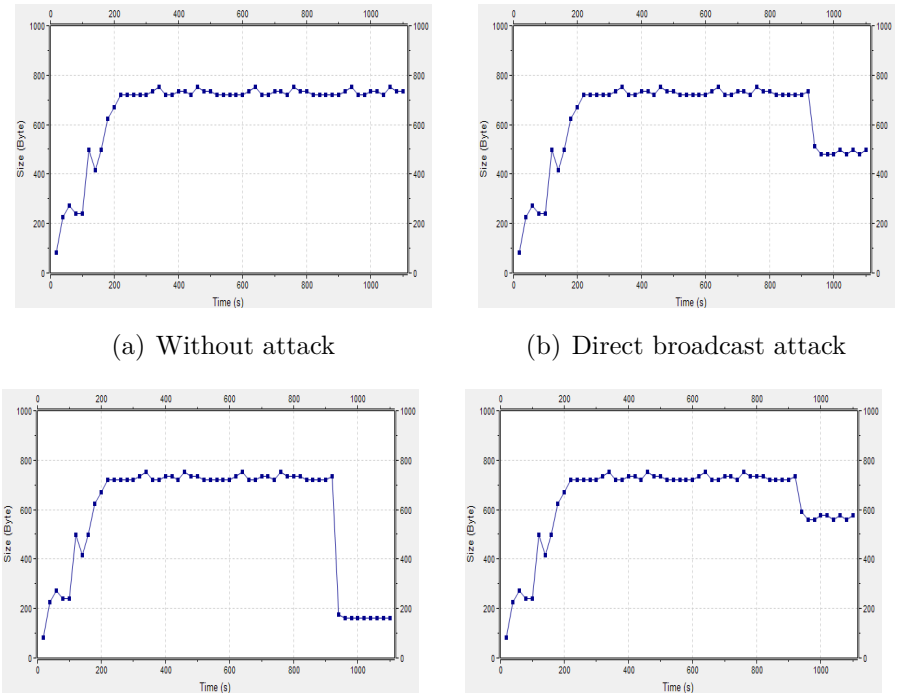
(a) Physical topology (b) Logical topology

Figure 4.17: Simulation network topology

For testing the three scenarios, we launched the broadcast attack at $T = 800s$ and the *Device*5 is configured to be the malicious insider attacker. The injected false command is the command 961 that is used to set a new *network key*. This command has 2 parameters: *the new network key*, and T' the time when it will be changed. In all the three scenarios $T' = 920s$.

As illustrated in Figure 4.18(a) i.e., in the normal case, the size of sensing data received by the Network Manager is about 720 bytes each 4s. We observe that for the three scenarios of the broadcast attack, the size of received data by the Network Manager falls immediately at $T = 920$. This indicates that the Network manager stops receiving sensing data from some wireless sensors.

Indeed at $T = 920$ *infected devices* will execute the injected false command and start to use the received network key to calculate the DLL MIC. When received by a device that has not been infected by the attack, the packet do not pass the MIC



(a) Without attack

(b) Direct broadcast attack

(c) Bounced broadcast attack

(d) On-the-fly broadcast attack

Figure 4.18: Sensing data received by the Network Manager.

validation step and is rejected. Consequently, packet sent by *infected devices* will be rejected and not received by the Network Manager.

In comparison with the normal case, in the direct command injection attack the data received by the Network Manager, illustrated in Figure 4.18(b), falls from 720 bytes to 480 bytes. This represents a decrease of 33%. Indeed, 3 devices i.e., $Device5, Device8$ and $Device9$, are infected by this attack.

In the case of the bounced command injection attack, shown in Figure 4.18(c), we record a decrease of 77% in the data received by the Network Manager. This indicates that this kind of attacks, allows a malicious node to use its parent device as a relay to propagate the attack to a great number of devices. As result, 7 devices are infected by the attack, i.e, $Device5$, $Device2$, $Device4$, $Device6$, $Device7$, $Device8$ and $Device9$.

In the on-the-fly command injection attack, we configure the Network Manager to broadcast, at $T = 800s$ to all devices, a command to change the network key at $T' = 920s$. The malicious attacker will modify this command and send a false command to its children devices. This attack has the same impact as in the case of a direct command injection command. As a variant, we choose that the malicious node does not execute the false command, which explains the difference of the impact between the direct and on-the-fly broadcast attacks. As indicated in Figure 4.18(d), the received

data by the network manager decreased by 22% as only 2 devices are infected i.e., *Device*5 and *Device*6.

4.5.5 Countermeasures discussion

The broadcast communication is an important feature in WirelessHART. It allows the Network Manager to configure all devices composing the wireless network by only sending a single packet. It avoids a costing time and resources process of sending a single packet to each device. But as shown in this Section, this feature creates a dangerous breach in the communication scheme security. As it is complicated to ban broadcast communications, we propose hereafter, some ideas to reduce the exposition to this vulnerability.

- Broadcast packet validation after the reception of 2 identical packets: this condition aims to stop direct and on-the-fly command injections. Indeed, as WirelessHART builds a meshed network, best practices in industrial sensor networks recommend that each node has at least 2 or 3 parents. Consequently, each sensor will receive the broadcast packet more than once. Thus, according to this rule, each node must wait till the reception of the same packet from another of its parents before it executes and forwards it. Nodes located at one hop do not have to apply this rule as they receive the broadcast packet directly from the Network Manager. This countermeasure adds a latency in the transmission of broadcast packets and can, in some cases, block their forwarding.

- DLL and NL addresses validation: in the case of the bounced command injection attack, DLL and NL headers of the injected packet indicate contradictory informations. Indeed, the source address in the DLL header indicates that the packet has been sent by a children node i.e., the malicious node, while the source address in the NL header indicates that the packet has been sent by a parent node i.e., the network manager. Therefore, implementing in the NL a security mechanisms that rejects packets indicating such contradictory informations can mitigate this kind of attacks. We must note that even if this solution does not comply with the layer separation principle, in practice WirelessHART layers already use information provided by other layers such as addresses.

- Use of an IDS for monitoring node's behavior: indeed, except rethinking deeply the communication scheme of WirelessHART, as implementing asymmetric cryptography for packet's authentication, that is a costly process, the use of an IDS will increase significantly the security of such networks. Indeed, this kind of system

by monitoring exchanged packets, are able to detect the injection of a false packet or the modification of a packet during its transmission.

In conclusion, the two first countermeasures are partial solutions that do not prevent all scenarios. The second solution is the costless one as it adds a reduced overhead. The use of an IDS is the more efficient solution. Indeed, although it requires the installation of dedicated equipments for traffic monitoring, it is the only solution that detects all possible scenarios. Nevertheless, given that WSNs are distributed systems, we must pay attention to the scheme used to deploy the IDS as it directly impacts the information gathering capability.

4.6 Conclusion

In this Chapter, we have shown that an inside attacker can bypass security mechanisms implemented by the WirelessHART protocol and perform attacks on the network. These attacks are based on the use of cryptographic keys that are shared by all devices composing the wireless network.

Conducted tests, using a simulator dedicated to WirelessHART security assessment, have confirmed the feasibility of these attacks and their potentially harmful impact.

On the other hand, proposed solutions do not totally mitigate these attacks. Therefore and except changing deeply the communication scheme implemented by WirelessHART, the use of an Intrusion Detection System (IDS) is the best operational manner to detect and mitigate these attacks.

CHAPTER 5

wIDS: a multilayer IDS for Wireless-based SCADA Systems

5.1 Introduction

Wireless Industrial Sensor Network (WISN) are now established as a widely used technology in industrial environments. Indeed, comparing to wired technologies, they allow significant decreases in deployment and maintenance costs. In the same time, they increase the system sensing capabilities as wireless sensors can be deployed in hardly reachable and adversarial environments.

On the other hand, ensuring security in WISN is a challenging task. Indeed, WISN are subject to the same attacks as other wireless networks. Mainly, attackers use wireless communication as a vector to launch their attacks. Furthermore, sensor's limited capabilities in terms of processing power, memory space and energy make hard the implementation of strong security mechanisms. Consequently, in WSN security is generally a neglected aspect in favor of the reliability and the availability of the network

Thus, in addition to sensor's embedded mechanisms that ensure authentication, confidentiality and availability, Intrusion Detection Systems can be used as a second line of defense for enforcing the overall system security and in particular for detecting unknown attacks.

The exchanged traffic in a SCADA system is highly predictable in terms of amount and frequency. Indeed, it involves limited human interaction and is mainly composed of automated devices that execute defined actions at defined times. Therefore, by modeling the normal expected behavior of wireless nodes, we can detect malicious actions that do not respect the established model.

In this Chapter, we present wIDS, a multilayer specification-based Intrusion Detection System specially tailored for Wireless Industrial Sensor Networks. The proposed IDS checks the compliance of each action performed by a wireless node towards the formal model of the expected normal behavior. To do that, access control rules are used for modeling authorized actions that a wireless node can perform. These rules are mainly built on the base of the specifications of each layer of the communication protocol, node's localization and the industrial process configuration. They also take into consideration the capabilities and limitations of the wireless nodes. Thus, by specifying security policy at an abstract level, we are able to define and manage more accurate and efficient security rules independently from nodes and network characteristics such as sensor natures and density or the network topology. Then, these characteristics are used later when deriving concrete security rules. Also, in addition to alerts that are raised by actions deviating from the normal model, we define additional intrusion rules that aim to detect basic attacker actions such as injecting, deleting, modifying and delaying packets.

The rest of the Chapter is organized as follows. Section 5.2 presents previous work done in this field and emphasis their limits. We present in Section 5.3, the proposed IDS for Wireless Industrial Sensor Networks. We describe its two-level detection architecture and present the formalism used to build node's normal behavior. Section 5.5 details security rules defined on the base of WirelessHART specifications. In Section 5.6, we present how detection rules are defined to detect suspicious actions. The performances of the proposed wIDS are presented and discussed in Section 5.7. Finally, Section 5.8 presents the conclusion and future works.

5.2 Related Work

In the literature, there are only few studies on the security of Wireless Sensor Networks used in industrial environments. Mainly, proposed solutions for SCADA systems focus on applying IDS techniques to wired-based networks [Huitsing et al. 2008, Fovino et al. 2010, Mitchell and Chen 2013] and neglect those using wireless communications.

More generally, several IDS are proposed for generic WSN [Mitchell and Chen 2014]. However, proposed solutions are not suitable for WISN. Firstly, WISN has harder requirements than generic WSN such as real-time and reliable communications. Indeed, dropped or delayed data may lead to physical losses. Secondly, these proposed IDS are mainly restricted to detect specific kinds of attacks

while WISN must be secured towards a broad spectrum of known and unknown attacks. Nevertheless, these studies should be considered in order to propose a solution designed for WISN.

Thus, in [da Silva et al. 2005], Da Sila et al. propose one of the first intrusion detection systems for WSN. They designed a decentralized system in which a set of nodes is designated as *monitor* and is responsible of monitoring their neighbors. The proposed IDS is based on the statistical inference of the network behavior. It only monitors data messages and ignores other kinds of exchanged messages. It includes seven types of rules that aim to detect common attacks.

Roosta et al. propose in [Roosta et al. 2008] an intrusion detection system for wireless process control systems. The system consists of two components: a central IDS and multiple field IDS that passively monitor communications in their neighborhood. They periodically send collected data to the central IDS that checks their conformity with the security policy. This IDS models normal behavior of the different wireless devices on the base of some network specifications and traffic characteristics inferred statistically. Attacks are detected when there is a deviation from the model. However, it defines a few numbers of rules (8 rules) that do not cover all well-known attacks. Furthermore, as the detection logic is centralized, this solution requires continuous communications with field IDS which can add a significant network overload.

In [Coppolino et al. 2010], Coppolino et al. propose an architecture for an intrusion detection system for critical information infrastructures using wireless sensor network. Their solution is a hybrid approach combining misuse and anomaly based techniques. It is composed of a Central Agent and several IDS Local Agents that monitors exchanged messages in their neighborhood. They calculate a statistical model of exchanged traffic and raise a temporary alert when nodes actions deviate from this model. The central agent combines these alerts and confirms them on the base of misuse rules. This IDS focuses on attacks against routing protocols and detects only two kinds of attacks i.e., sinkhole and sleep deprivation attacks.

Shin et al. [Shin et al. 2010] propose a hierarchical framework for intrusion detection for WISN. It is based on two-level clustering; multihop clusters for data aggregation and one-hop clusters for intrusion detection. This results in a four layers hierarchy: member nodes (MN) are the leaves, cluster heads (CH) manage MNs, gateways (GW) bundle clusters and a base station (BS) is the root of the hierarchy. These different levels implement the same detection logic, however they respond differently. Thus, MN only report to CH while other roles have the ability to react to attacks.

In our study, we aim to propose a solution that is able to detect either known and unknown attacks. Furthermore, such solution should have a multilevel detection architecture to monitor both local and end-to-end communications (generally encrypted) and also in order to provide global coordination. Low detection level should have full detection capabilities in order to avoid overloading the network by additional exchanges and to have quick and accurate detections.

5.3 Multilayer specification-based IDS

In this Section, we present *wIDS* a multilayer specification based IDS for securing Wireless Industrial Sensors Networks. We describe its architecture, its components and its analyzing process.

Specification-based intrusion detection approaches formally define the model of legitimate behavior and raise intrusion alerts when user's actions deviate from the model [Mitchell and Chen 2013][Mitchell and Chen 2014]. WISN are composed of nodes that have a predictable behavior and involves few human interactions. Consequently, on the base of the communication protocol specifications, the process configuration and wireless nodes capabilities, we can build an accurate model of the expected nodes's behavior.

We should also note that specification-based intrusion detection system does not require any training step. Therefore, they can be applied and used directly.

In this study, we assume that the aim of the attacker is to disturb the industrial process. This goal can be achieved by dropping some packets, injecting into the network false packets or modifying packets during their transmission. Furthermore, the attacker can also choose to delay the transmission of some important packets (alarms, sensing data, etc) in order to lead the process to an uncertain state or to hide his malicious actions. Therefore, we consider an attacker that can intercept, modify, forge or delay packets. It can be an inside or an outside attacker.

5.3.1 wIDS architecture

As indicated in Figure 5.1, wIDS has a two-level architecture consisting in a central IDS-agent and several IDS-agents.

- The central IDS-agent: It is implemented in the Network Manager (resp. in the sink) in the case of WirelessHART (resp. in the case of a WISN). In addition of

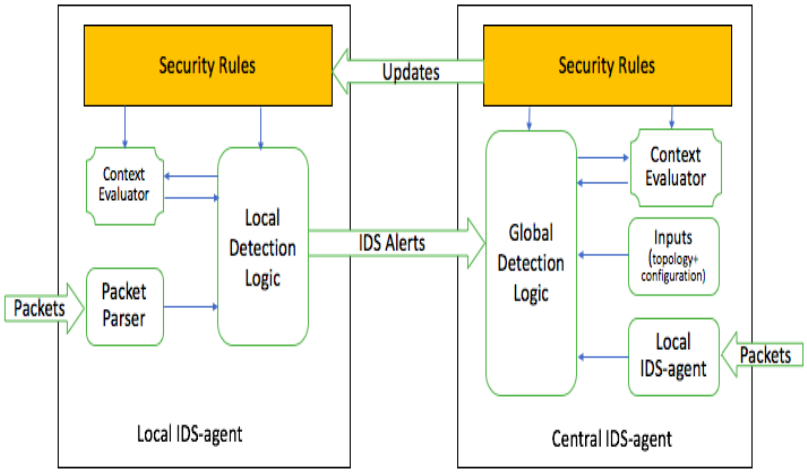


Figure 5.1: The Central IDS-agent and IDS-Agents Architecture

playing the role of an IDS-agent in its neighborhood, it monitors end-to-end communications after they are deciphered It may check routing tables and transmission scheduling consistency, and performs global coordination between IDS-agents.

- local IDS-agents: They are implemented in selected sensor nodes. They are responsible for monitoring local communications of sensor nodes inside their neighborhood. They listen in promiscuous mode to all packets exchanged in their neighborhood. Then, they extract from them, relevant information in order to check their compliance with the security policy.

The abstract security policy is defined at the central-IDS agent using the wirelessOrBAC formalism. It is also provided with several inputs such as node localizations, industrial process parameters and nodes configuration. The central-IDS agent provisions IDS-agents with security rules and several inputs related to nodes available in their neighborhood (i.e., the list of monitored nodes). It also updates if necessary all these information. Each IDS-agent is in charge of the application of the security policy in its area and alerts the central-IDS agent when policy violation occurs.

5.3.2 IDS-agents deployment scheme

The scheme used for the deployment of IDS-agents, is an important issue. Indeed, as WSN are decentralized systems, the localization of monitoring devices must be chosen carefully otherwise a part of exchanged traffic will not be monitored.

This scheme must present the following characteristics: *a)* each IDS-agent is able to detect basic attacks occurring in its neighborhood without any cooperation with other IDS-agents; *b)* it creates a secure and reliable communication channel between each IDS-agent and the central-IDS; *c)* it requires an acceptable IDS-agents number to ensure an efficient network monitoring and coverage.

In wIDS, IDS-agents are implemented in sensors with enhanced capabilities. These nodes, called *Super-Nodes*, will act as classical sensor nodes by fulfilling sensing tasks and implement in the same time the detection logic.

We propose in Chapter 6, a scheme that fulfills all the above mentioned requirements. This scheme uses the graph theory concept of *Connected Dominating Set* to ensure the gathering of the whole exchanged traffic.

By using the aforementioned deployment scheme, selected *Super-Nodes* represents between 20%-25% of the total network node number.

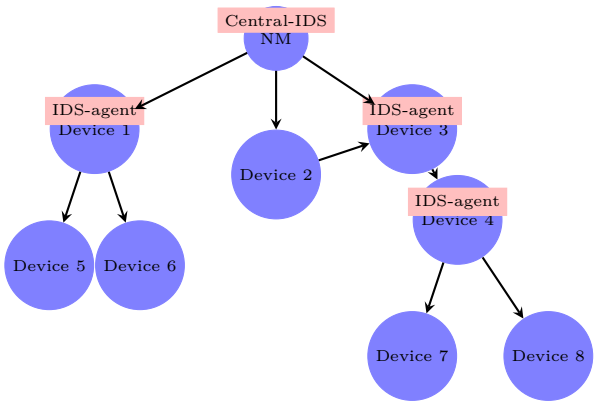


Figure 5.2: IDS-agents deployment

5.4 WirelessOrBAC: An access Control-based Intrusion detection formalism

WirelessOrBAC is an extension of OrBAC, specially tailored for Wireless Sensor Networks. OrBAC has already been used to specify network security policy, especially in firewall management [Cuppens et al. 2004], intrusion detection (IDS) and intrusion prevention systems (IPS) [Debar et al. 2007]. In OrBAC, the use of the concept of *context* makes it possible to define dynamic rules that fit system changes. However, as simplicity and flexibility are very important aspects in WSN, we adapt the OrBAC model in order to be easily applicable to these networks.

Thus, wirelessOrBAC aims to enforce WSN security requirements by relying on an access control policy. Indeed, as wireless nodes are expected to follow a defined behavior, the set of authorized actions that could be executed by a node are known. This set of actions is formalized in wirelessOrBAC using access control rules to define the expected behavior of wireless nodes. By relying on this model, malicious actions can be detected by checking the compliance of each nodes actions toward the defined security policy.

WirelessOrBAC allows the definition of both access control and intrusion detection rules. Access control rules describe authorized node actions regarding several constraints (i.e., device capabilities, communications protocol specification, network topology) and requirements (i.e., network purpose). Intrusion detection rules explicitly specify unauthorized actions or those regarded as malicious.

5.4.1 Wireless sensor Networks limitations

Nowadays, Wireless Sensor Networks (WSN) have a wide application range. This is mainly due to their low-cost and flexible deployment and management. A wireless sensor consists of four basic parts: a sensing unit, a processing unit, a transceiver unit, and a power unit [Wang et al. 2006]. It may also have additional application-dependent components such as a location finding system or a power generator. They are able to monitor a variety of phenomenons such as [Akyildiz and Vuran 2010]: temperature, humidity, object's movement, speed or direction, luminosity condition, pressure, noise levels and the presence or absence of objects.

In order to fit these different applications, wireless sensors have several requirements as they must [Akyildiz et al. 2002]: a) consume extremely low power; b) operate in high volumetric densities; c) have low production cost; d) be autonomous; e) be adaptive to the environment.

As results these sensor nodes have several limitations [Wang et al. 2006]:

- Energy: for sensor nodes, the energy is one of the most important and limited resources. The tasks performed by sensor nodes (such as transmitting/receiving and processing data) must be well planned in order to increase the network lifetime. Indeed communication is more costly than computation as each bit transmitted consumes about as much power as executing 800-1000 instructions [Lee et al. 2010].
- Computation: sensor nodes have limited computational capabilities. They can hardly execute complex algorithms such as cryptographic operations.

- Memory: sensor nodes include limited memory space that is mainly used for the storage of a specialized Operating System, application programs and sensor data. There is usually not enough space to load additional functionalities.
- Transmission range: the communication range of sensor nodes is limited both for technical reason and also by the need to save energy.

All these limitations and constraints make ensuring security in WSN a challenging task. Indeed, on the one hand WSN are decentralized systems that are deployed in hostile environments which make them vulnerable to several kinds of attacks. On the other hand, their limited capabilities make it hard to implement in them strong security mechanisms. Thus, generally security in WSN is a neglected aspect in favor of the reliability and the availability of the network.

By using wirelessOrBAC, it becomes easier for the network administrator to define, manage and deploy security policy that fits its WSN characteristics. Indeed, specifying security policy at an abstract level, allows the expression and update of more accurate and efficient security rules independently from specific network characteristics such as sensor's natures, node density, network topology and communication protocols. These characteristics are used later when deriving concrete security rules.

5.4.2 WSN access control models

Access control policy is a well studied field in classical computer science but has not received much attention in the context of WSN. Indeed, as WSN have inherent constraints, mainly limited computation and battery resources, classical security solutions are practically hard to implement.

A literature review of available access control models in WSNs are proposed in [Maw et al. 2014b]. It indicates that research efforts focus mainly on ensuring node's authentication through cryptography, key distribution and management schemes and access control lists.

Thus in [Benenson et al. 2005], the access control problem for sensor networks is divided into the separate subproblems of node's authentication and authorization. It aims particularly to deal with the node capture problem and malicious node presence.

In [Marsh et al. 2009] a policy language called the WSN Authorization Specification Language (WASL) is proposed. Access control rules are defined using this language. These rules are processed and required authorizations are distributed to each node. This reduces both the memory and computational requirements at node level.

Authors in [Maw et al. 2012] propose an adaptive access control model with privileges overriding and behavior monitoring, specially designed for WSN. The proposed model can dynamically grant and deny permission based on the overriding concept and user behavior monitoring. Authors have not provided details on how the user behavior model is built.

In [Maw et al. 2014a], authors propose the Break-The-Glass Access Control (BTG-AC) model, a light-weight and flexible access control model. It introduces the concept of break-the-glass (BTG) rules that grant users an immediate and urgent access to the system in emergency and some defined important cases.

Most of the proposed access control models for WSN [Maw et al. 2014b] focused on node authentication and exchange authentication, but other requirements such as availability and data time delivery have received a little attention. On the other hand, proposed access control models leak flexibility, especially in the case of unexpected events. Indeed, WSN evolve in uncertain environments where it is complex to predict all situations. Furthermore, the design of access control models must consider WSN constraints i.e., available energy, computation capacity and storage memory.

Thus, with wirelessOrBAC we aim to cover all WSN requirements and integrate the limitations and constraints of this kind of network. And also it is an adaptive model that can handle different kinds of contexts.

5.4.3 Background on the OrBAC model

Organization, role, activity and view

The OrBAC [Kalam et al. 2003] model is an extension of the role-based access control (RBAC) model. It introduces the concept of organization that corresponds to any entity in charge of managing a set of security rules.

In OrBAC model, rules do not directly apply to subject, action and object. Instead, subject, action and object are respectively abstracted into *role*, *activity* and *view*. Thus, subjects obtain permission based on the role they play in the organization.

Once a security policy has been specified at the organizational level, it is possible to instantiate it by assigning concrete entities to abstract entities. This is done by using the following predicates which assign a subject to a role, an action to an activity and an object to a view:

- *empower(org, subject, role)*: means that in organization *org*, *subject* is empowered in *role*.
- *consider(org, action, activity)*: means that in organization *org*, *action* is considered an implementation of *activity*.
- *use(org, object, view)*: means that *object* is used in *view* of organization *org*.

Context

In OrBAC, the notion of *context* is used to model extra conditions that a subject, an action and an object must satisfy to activate a security rule [Kalam et al. 2003][Cuppens and Cuppens-Boulahia 2008]. A context is defined through logical rules called *context definition* that can be combined in order to express conjunctive, disjunctive and negative contexts.

A context is declared as follows:

$$hold(org, subject, action, object, context) \tag{5.1}$$

that means that in organization, *org*, *subject* performs *action* on *object* in context *context*.

The taxonomy of different available types of context is as follows [Cuppens and Cuppens-Boulahia 2008]:

- the *Temporal context* that depends on the time at which the subject is requesting for an access to the system,
- the *Spatial context* that depends on the subject location,
- the *User-declared context* that depends on the subject objective (or purpose),
- the *Prerequisite context* that depends on characteristics that join the subject, the action and the object,
- the *Provisional context* that depends on previous actions the subject has performed in the system.

Abstract and concrete OrBAC Rules definition

An OrBAC policy is defined as follows:

$$security\_rule(organization, role, activity, view, context) \tag{5.2}$$

where security_rule belongs to permission, prohibition, obligation and dispensation.

As an example, security rules are defined as follows:

$$Permission(org, r, a, v, c) \tag{5.3}$$

that means that in organization *org,* role r is granted *permission* to perform activity a on view v within context c.

Based on the above definitions, a concrete permission policy could be derived by the following rule:

$$\begin{aligned} is\_Permitted(subject, action, object) \leftarrow \\ permission(org, role, activity, view, context) \\ \wedge empower(org, subject, role) \wedge consider(org, action, activity) \\ \wedge use(org, object, view) \wedge hold(org, subject, action, object, context) \end{aligned} \tag{5.4}$$

5.4.4 WirlessOrBAC rules semantic

In order to be more accurate and easy to be applied in a WSN, wirelessOrBAC defines the following concepts that are inherited from the OrBAC model:

wNetwork

In wirelessOrBAC, a security policy will be modeled using one or several organizations. A WSN may be considered as an organization. Sub-organization may also be defined if for example a WSN is composed of several networks with different locations such as floors in a building, or according to their purpose.

Definition 1 (wNetwork)**.** *It is used to specify an organization that represents a WSN. It can be composed of several sub-wNetworks.*

wRole and wDevice

In wirelessOrBAC, subjects correspond to wireless nodes. So, we consider roles that may be assigned to them. Thus, wireless nodes may have one or several of the following roles:

- Sensor: It is a device that has the capability to collect sensing data.
- Sink: It is the final destination and the end user of sensing data.
- Base station: It is a device in charge of the network management.
- Forwarder: In a multi-hop network, it is a device that has the capability of forwarding packets of other nodes to their final destination.
- Aggregator: It is a device that gathers sensing data from several sensors, processes them and sends the computed data (such as the average value of received data) to the sink.
- Cluster-head: It is a device in charge of providing some services to the sensor nodes in its neighborhood.
- Provisioning: It is a device that has the capability to provide information about the network to new devices attempting to join the network.

Definition 2 (wDevice and wRole)**.** *wRoles are predefined roles used in WSN. A wDevice is composed of one or several wRole in a conjunctive way.*

wActivity

WirelessOrBAC considers all actions that wireless nodes are able to execute.

Definition 3 (wActivity)**.** *It is an abstraction of wireless actions that a wDevice can perform.*

In wirelessOrBAC, *wActivity* are specified according to the communication protocol implemented by the wireless network. More generally, the following *wActivity* may be considered: sending, receiving, forwarding and aggregating.

wView

Exchanged messages between wireless nodes are regarded in wirelessOrBAC as the object on which subject actions are performed.

Definition 4 (wView)**.** *It is an abstraction of messages sent from a wDevice to another wDevice.*

Similarly to *wActivity*, *wView* depends on the communication protocol implemented by wireless nodes.

wContext

As indicated in Section 5.4.3, security rules are activated when related contexts hold. Therefore several kinds of context are defined into wirelessOrBAC.

Definition 5 (wContext)**.** *It represents predefined wirelessOrBAC contexts related to WSN.*

5.4.5 wRules expression

A wirelessOrBAC policy is defined through the specification of security rules that indicate if an action performed by a wireless node is allowed or not.

Definition 6 (wirelessOrBAC Policy)**.** *It is the set of wRules considered in a wNetwork.*

Definition 7 (wRule)**.** *It is a 6-ary predicate that indicates in a wNetwork, permission, prohibition, obligation or dispensation, that wDevice is granted to perform wActivity on wView.*

In wirelessOrBAC, a security rule is defined as follows:

$$wRule(security\_rule, wnet, d, a, v, c) \tag{5.5}$$

that means that in wNetwork *wnet,* wDevice d is granted *security_rule* to perform wActivity a on wView v within wContext c.

Concrete permission security rules are derived as follows:

$$\begin{aligned} Is\_Permitted(s, a, o) \leftarrow \\ wRule(perm, wnet, wRole, wActivity, wView, wContext) \wedge \\ empower(wnet, d, wRole) \wedge consider(wnet, a, wActivity) \wedge \\ use(wnet, o, wView) \wedge hold(wnet, s, a, o, wContext) \end{aligned} \tag{5.6}$$

5.5 Expected behavior modeling rules

In this Section, we define using wirelessOrBAC and on the base WirelessHART specification [HART Communication Foundation], rules that model the expected node's normal behavior. These rules express authorized actions at each protocol layer. We gather them in several categories and present hereafter, examples of each of them.

5.5.1 Meshed wireless network rules

In a WirelessHART network, all devices have the capability to forward packets of devices that are located several hops away from the Network Manager. That means that a device can send packets to any of its neighbors:

$$wRule(perm, wnet, wDevice, sending, packets, neighbors) \tag{5.7}$$

where *neighbors* is a wContext indicating that $s1$ performs action *sending* object *packet* to node $s2$ that is in its neighborhood:

$$\begin{aligned} hold(WSN, s1, sending, packet, neighbors) \\ \leftarrow is\_dstAddr(packet, s2) \wedge is\_neighbor(s1, s2) \end{aligned} \tag{5.8}$$

5.5.2 Packets construction rules

The WirelessHART specifications give guidelines on how packets should be built and the possible values of each field. Thus, the length and value of each field must be checked. Also, fields of the same packet must be consistent with each other.

$$\begin{aligned} wRule(perm, wnet, wDevice, sending, packet, \_) \\ \leftarrow is\_ValidPacket(packet) \end{aligned} \tag{5.9}$$

where $is\_ValidPacket(packet)$ is a predicate indicating if *packet* fulfills WirelessHART construction rules. The symbol "_" indicates that this rule is valid for any wContext.

5.5.3 Communication level

WirelessHART defines 5 packet types: *Ack*, *Advertise*, *Keep-Alive*, *Disconnect* and *Data* packets. The first 4 types are generated and processed in the Data Link Layer and are not propagated to the Network Layer or forwarded through the network. This means that these packets are only used in local communication between neighbors.

$$\begin{aligned} wRule(perm, wnet, wDevice, sending, packet, neighbors) \\ \leftarrow is\_packetType(packet, Ack|Advertise|keepAlive|Disconnect|Data) \end{aligned} \tag{5.10}$$

The Data packet type is the only kind of packets that is transmitted in an end-to-end communication. This means that only data packets can be *forwarded* throughout the network:

$$\begin{aligned} wRule(perm, wnet, wDevice, forwarding, packet, \_) \\ \leftarrow is\_packetType(packet, Data) \end{aligned} \tag{5.11}$$

On the other hand, data packets are only exchanged between the Network Manager (*wSink*) and wireless sensors (*wSensor*) in both ways. This means that a data packet is never sent from a wireless sensor to another wireless sensor. Thus:

$$\begin{aligned} wRule(perm, wnet, wSensor, sending, packet, \_) \\ \leftarrow is\_packetType(packet, Data) \\ \wedge is\_dstNLAddr(packet, s2.addr) \wedge empower(wnet, s2, wSink) \end{aligned} \tag{5.12}$$

and

$$\begin{aligned} wRule(perm, wnet, wSink, sending, packet, \_) \\ \leftarrow is\_packetType(packet, Data) \\ \wedge is\_dstNLAddr(packet, s2.addr) \wedge empower(wnet, s2, wSensor) \end{aligned} \tag{5.13}$$

5.5.4 Communication scheduling rules

As explained in Section 3.3.4, WirelessHART uses *Time Division Multiple Access* (TDMA) and *Channel hopping* to control the access to the wireless medium. The time is divided in consecutive periods of the same duration called *slots*. Each communication between two devices occurs in one slot of 10 ms.

Typically, two devices are assigned to one time slot (one as the sender and a second as the receiver). Only one packet is transmitted in one slot from the sender to the receiver which has to reply with an acknowledgment packet in the same slot.

In addition, WirelessHART uses channel hopping to provide frequency diversity and avoid interferences. Thus, the 2.4 GHz band is divided into 16 channels numbered from 11 to 26 which provide up to 15 communications in the same slot (Channel 26 is not used). Thus, we have the following rules:

$$wRule(perm, wnet, wDevice, sending, packet, startSlot \wedge assignedFq) \tag{5.14}$$

where *startSlot* is a wContext indicating that s performs action *sending* object *packet* when a slot time assigned to s starts:

$$\begin{aligned} hold(WSN, s, sending, packet, startSlot) \\ \leftarrow is\_slotStartTime(s, packet) \end{aligned} \tag{5.15}$$

and *assignedFq* is a wContext indicating that s uses its assigned frequency when performing action *sending* object *packet* :

$$\begin{aligned} hold(WSN, s, sending, packet, assignedFq) \\ \leftarrow is\_assignedFq(s, packet) \end{aligned} \tag{5.16}$$

For the acknowledgment, we have the following rule:

$$wRule(perm, wnet, wDevice, sending, packet, sendingAck) \tag{5.17}$$

where *sendingAck* is a wContext indicating that $s1$ performs action *sending* object *packet*′ when $s1$ received *packet* from s (at time t), and *packet*′ is destined to s and of type *ack* and $s1$ uses the slot and frequency assigned to s:

$$\begin{aligned} &hold(WSN, s1, sending, packet', sendingAck) \\ \leftarrow\ &packet\_received(s1, packet, t) \wedge is\_srcAddr(packet, s) \\ &\wedge is\_packetType(packet', Ack) \wedge is\_dstAddr(packet', s) \\ &\wedge is\_assignedFq(s, packet) \wedge is\_slotStartTime(s, packet) \end{aligned} \tag{5.18}$$

We should note that the Network Manager is responsible of building, managing and updating slots and frequencies planning.

5.5.5 Packets transmission rules

Sensor nodes are configured to send different kind of packets (i.e., sensing data, keep-alive, advertisement) at a defined time. Thus, sensing data must be sent periodically to the Network Manager.

$$\begin{aligned} &wRule(perm, wnet, wDevice, sending, packet, packetPeriodicity) \\ &\leftarrow is\_packetType(packet, Data) \end{aligned} \tag{5.19}$$

where *packetPeriodicity* is a temporal context indicating that s performs action *sending* object *packet* in the planned sending time:

$$\begin{aligned} &hold(WSN, s, sending, packet, packetPeriodicity) \\ &\leftarrow is\_packetPeriodicity(s, packet) \end{aligned} \tag{5.20}$$

5.5.6 Packets forwarding rules

A WirelessHART network has a meshed topology. Thus, wireless devices that are located several hops from the Network Manager, relay on their neighbors for forwarding their packets to their final destination.

$$wRule(perm, wnet, wDevice, forwarding, packet, ForwardPacket) \tag{5.21}$$

where *ForwardPacket* is a provisional context (i.e., based on previous device actions) indicating that subject s received object *packet* (at time t) and s must forward this object:

$$\begin{aligned} hold(WSN, s, forwarding, packet, ForwardPacket) \\ \leftarrow packet\_received(s, packet, t) \wedge is\_toBeForwarded(packet) \\ \wedge empower(org, s, forwarder) \end{aligned} \tag{5.22}$$

5.5.7 Routing rules

As indicated in Section 3.3.4, WirelessHART uses graphs as routing method. A graph consists in a set of directed paths that connect network devices. It is built by the Network manager based on its knowledge of the network topology and connectivity. Every graph has a unique graph identifier that is inserted in the network packet header. Each device receiving this packet, forwards it to the next hop belonging to that graph.

$$wRule(perm, wnet, wDevice, sending, packets, graphNextHop) \tag{5.23}$$

where *graphNextHop* is a spatial context indicating that s performs *sending* object *packet* to $s2$ that is the next hop of s following graph g:

$$\begin{aligned} hold(WSN, s, sending, packet, graphNextHop) \\ \leftarrow is\_dstAddr(packet, s2.addr) \wedge \\ is\_usedGraph(packet, g) \wedge is\_NextHop(s.addr, g, s2.addr) \end{aligned} \tag{5.24}$$

In graph routing there are two kinds of graphs: an *upstream* graph directed from all devices to the Network Manager and several *downstream* graphs directed from the Network Manager to each device. Thus, sensor nodes use the *upstream* graph for sending packets to the Network Manager:

$$\begin{aligned} wRule(perm, wnet, wSensor, sending, packet, \_) \\ \leftarrow is\_packetType(packet, Data) \\ \wedge is\_usedGraph(packet, g) \wedge is\_upStream(g) \end{aligned} \tag{5.25}$$

and the Network Manager uses *downstream* graphs for sending packets to sensors:

$$\begin{aligned} wRule(perm, wnet, wSink, sending, packet, \_) \\ \leftarrow is\_packetType(packet, Data) \\ \wedge is\_usedGraph(packet, g) \wedge is\_downStream(g) \end{aligned} \tag{5.26}$$

5.5.8 Cross layer consistency rules

As indicated in Section 5.5.2, packet's fields must comply with the protocol specifications and also be consistent between them. This verification is done according to each layer rules. However, an attacker can bypass this verification giving contradictory information that fulfills each layer rules. Therefore, for some fields a cross layer verification must be applied. For example, in the case of routing information, DLL and NL fields must be consistent:

$$\begin{aligned} & wRule(perm, wnet, wSensor, sending, packet, \_) \\ \leftarrow\ & is\_dstAddr(packet, s1.addr) \wedge is\_dstNLAddr(packet, s2.addr) \\ & \wedge is\_usedGraph(packet, g) \wedge is\_NextHop(s1.addr, g, s2.addr) \end{aligned} \tag{5.27}$$

5.6 wIDS detection rules

In order to detect malicious actions, wIDS applies a close policy requirement. This means that each action initiated by wireless nodes is compared to the defined security policy and a security alert is raised if the verification failed. Thus, all node's actions must:

1. be explicitly allowed by the security policy;
2. and is compliant with all rules that match the action.

Thus, each IDS-agent implements Algorithm 1 in order to check the compliance of actions performed by wireless nodes.

Each time a node s performs an action a on object o, we first build M the set of security rules that matches the tuple {s,a,o}. If the set M is empty this indicates that there is not a security rule that explicitly permits that s performs an action a on object o. Otherwise, the tuple {s,a,o} is compared towards each rule $m \in M$ to check if it is compliant with that rule (see Rule 5.6). If the tuple {s,a,o} is not compliant with a security rule m, it is considered as a malicious action and an intrusion rule is raised. Else, if the tuple {s,a,o} is compliant with all security rules $m \in M$, it is considered as a legitimate action.

We define in wIDS two kinds of IDS alerts: *default* and *basic malicious actions* alerts. The default IDS alerts aim to detect any action deviating from the expected node's behavior. The basic malicious actions alerts aim to bring additional information about the detect malicious action and its purpose. These two kinds of alerts are described hereafter.

Algorithm 1 Conformity checking algorithm

```
1: procedure ACTIONVALIDATION(s, a, o)        ▷ subject s performs action a on o
2:     M=matchingRule(s,a,o);                 ▷ Build M the set of rules matching s,a and o
3:     if M is empty then
4:         return false;                      ▷ s is not permitted to perform action a on o
5:     end if
6:     validAction ← true;
7:     while M is not empty ∧ validAction do  ▷ repeat until all rules are checked
8:         Select m from the set M;
9:         M = M − {m};
10:        validAction ← checkValidity(s,a,o,m);   ▷ checks if rule m allows that s
   performs a on o
11:    end while
12:    return validAction
13: end procedure
```

5.6.1 Default IDS alert

We chose to model IDS alert as *wContext*. This permits not only the accurate identification of the malicious action but also can allow an automatic reaction by for example the activation or deactivation of some security rules. It also allows the global coordination of alerts in the central-IDS.

To do that, we define *idsAlertCtx* a default context that is activated when an action performed by a wireless node violates a security rule defined in Section 5.5:

$$\begin{aligned} &\forall s \in S, o \in O, a \in A, \\ &hold(wnet, s, a, o, idsAlertCtx) \leftarrow \\ &Action(s, a, o) \wedge \neg actionValidation(s, a, o) \end{aligned} \tag{5.28}$$

where *Action(s,a,o)* indicates that subject s performed action a on object o and $\neg actionValidation(s, a, o)$ (See Algorithm 1) indicates that $Action(s, a, o)$ does not match any defined security rules.

5.6.2 Basic malicious action IDS alert

In order to enforce wIDS detection capabilities, we define additional IDS alerts that aim to detect basic actions that an attacker can perform such as intercepting, deleting, modifying, forging or delaying packets.

1. *Packets and fields specification*: According to the communication protocol used by the WSN, exchanged packets must follows some rules in terms of packets size and fields value. Indeed, a malicious node can inject into the network malformed packets in order to lead receiving nodes to unstable state.

$$\begin{aligned} hold(wnet, s, sending, packet, not\_valid\_packet) \\ \leftarrow \neg is\_ValidPacket(packet) \end{aligned} \tag{5.29}$$

2. *Forging a fake packet*: In this attack, the subject s forwards a packet o however the context *forwardingPacket* is not active. This means that the packet o is a packet forged by s that pretends forwarding a received packet.

$$\begin{aligned} hold(wnet, s, a, o, forged\_packet) \leftarrow \\ empower(wnet, s, wForwarder) \wedge consider(wnet, a, sending) \\ \wedge use(wnet, o, packets) \\ \wedge \neg hold(wnet, s, a, o, forwardPacket) \end{aligned} \tag{5.30}$$

3. *Delaying a packet*: In this attack, the subject s forwards a received packet o after that the maximum forwarding time has expired.

$$\begin{aligned} hold(wnet, s, a, o, delayed\_packet) \leftarrow \\ empower(wnet, s, wForwarder) \\ \wedge consider(wnet, a, sending) \wedge use(wnet, o, packets) \\ \wedge packet\_received(s, o, t) \wedge packet\_sent(s, o', t') \\ \wedge is\_forwadedVersion(o, o') \\ \wedge hold(wnet, s, a, o, ForwardPacket) \wedge (t + \delta) < t' \end{aligned} \tag{5.31}$$

   where δ represents the maximal time a packet must be forwarded within since it is received.

4. *Deleting a packet*: In this attack, the subject s does not forward a received packet o within the defined time δ'.

$$\begin{aligned} hold(wnet, s, a, o, deleting\_packet) \leftarrow \\ empower(wnet, s, wForwarder) \\ \wedge consider(wnet, a, sending) \wedge use(wnet, o, packets) \\ \wedge packet\_received(s, o, t) \wedge \neg packet\_sent(s, o, t') \\ \wedge is\_forwadedVersion(o, o') \\ \wedge hold(wnet, s, a, o, ForwardPacket) \wedge (t' < t + \delta') \end{aligned} \tag{5.32}$$

   Thus, a packet is considered as deleted if it has not been forwarded within the time δ' (with $\delta < \delta'$). Between δ and δ' a packet not forwarded is considered as delayed.

5. *Modifying a packet*: In this attack, the subject s forwards a modified version of a received packet o that does not comply with the used communication protocol.

$$\begin{aligned} hold(wnet, s, a, o, modified\_packet) \leftarrow \\ empower(wnet, s, forwarder) \wedge consider(wnet, a, sending) \\ \wedge use(wnet, o, packets) \\ \wedge packet\_received(s, o, t) \wedge packet\_sent(s, o', t') \\ \wedge is\_forwadedVersion(o, o') \\ \wedge hold(wnet, s, a, o, forwardPacket) \wedge is\_ValidPacket(o') \end{aligned} \tag{5.33}$$

In conclusion, the IDS default alert allows us to detect that a node performed a malicious action and the basic malicious actions alerts allow us to identify the nature of the perform malicious action.

5.7 Implementation and Evaluations

5.7.1 Implementation

To evaluate wIDS performances, we use WirelessHART NetSIM [Bayou et al. 2015b]. The simulated network is composed of a network manager and 9 wireless sensors. We implement IDS-agents into 3 of them and launched randomly attacks from random nodes .

We test the proposed wIDS towards the attacks described in Section 3.5.

5.7.2 Experimental results

As indicated in Table 5.1, wIDS detects all tested well-known attacks. Each of these attacks, is not compliant with one or several security rules.

Performed tests report 100% correct identification of malicious actions and less than 2% of false positives. Depending on which security rule is violated, false positives rate is about 0% for sybil or broadcast attacks and about 5% for jamming, DoS or forced delay attacks. Indeed, first cited attacks are composed of actions that are clearly identified as malicious while the second cited attacks can be assimilated to transitory transmission perturbations such as interferences. This rate may be reduced by the use of a threshold.

Attacks	Detection Rules
Jamming	Rule (5.9), Rule (5.14), Rule (5.19)
Denial of Service (DoS)	Rule (5.14), Rule (5.19)
Sinkhole and blackhole	Rule (5.7)
Selective forwarding	Rule (5.21)
Hello Flood	Rule (5.7)
Forced delay	Rule (5.21)
Sybil	Rule (5.7)
Broadcast	Rule (5.21), Rule (5.26), Rule (5.27)

Table 5.1: Well-known Attacks Detection

5.7.3 Discussion

Previous results confirm the correctness of wIDS conception. They show that the normal behavior of wireless nodes can be modelized. As expected, the detection rate is 100% and depends highly of the accuracy of node normal behavior. By combining local and central detection, wIDS can be applied to networks of several sizes both in terms of nodes number and geographical area. Indeed, IDS-agents have the capabilities to detect basic malicious actions without any cooperation between them.

Also, by focusing on the detection of basic malicious actions, wIDS is able to detect known attacks as well as unknown ones and this without requiring any training phase.

5.8 Conclusion and Future Works

In this Chapter, we have presented wIDS an efficient intrusion detection system specially designed for enforcing wireless-based SCADA systems. It builds the normal behavior model of wireless nodes on the base of used protocol specification. The model is then used for monitoring nodes actions and a security alert is raised when any node action deviate from this model. Conducted tests have confirmed that wIDS is able to detect a large number of attacks with a low false-positive rate. These performances rely mainly on the quality of the nodes normal behavior model that depends on expert knowledges.

On the other hand, as tests were conducted in a simulated environments, some physical phenomenons were not considered. Indeed, WISN are expected to be deployed in industrial harsh environment characterized by wide temperature range, vibrations,

reflections due to metallic structures, etc. Such an environment can impact communication reliability which can increase the false-positive rate.

Furthermore, we should pay attention to the generation and deployment of security rules. Indeed, wireless nodes have limited storage and computation capabilities. Therefore, deployment scheme should provides wireless nodes with only required rules. This will avoid to overload both the network and wireless nodes.

Regarding wirelessOrBAC usability and applicability in WSN, an important point that should be considered is modeling node's normal behavior. Indeed, this task depends mainly on the used protocol complexity. At this step, wirelessOrBAC through its abstract concepts, particularly *wContext*, is enough flexible and powerful to deal with the different protocol specifications.

Finally, we should note that wIDS can be applied indifferently to any kind of wireless communication protocol. Indeed, wirelessHART is used in this work as an illustrative example.

CHAPTER 6

Towards a CDS-Based Intrusion Detection Deployment Scheme for Securing Industrial Wireless Sensor Networks

6.1 Introduction

As a specific application of WSN, Wireless Industrial Sensor Networks (WISN) present the followings key characteristics [Moyne and Tilbury 2007]:

- They are expected to work reliably in industrial harsh environment: wide temperature range, vibrations, reflections due to metallic structures, etc.
- There is an online trusted party (the base station).
- There is no data aggregation. All sensing data are sent to the base station.
- Used protocols must be energy efficient. The battery life of sensors is expected to last several years.

In terms of security, WISN are subject to the same attacks as WSN and other wireless networks. Indeed, attackers mainly use wireless communication as a medium to launch their attacks. Moreover, as WISN manage sensitive installations and facilities, attacks against them can lead to harmful economical consequences or even can

threaten human lives [Huang et al. 2009]. Therefore, several mechanisms were developed to enhance the security of these networks (Cryptography, Authentication, etc.). But as these security solutions cannot prevent all attacks, especially in the case of node compromise attacks [Bayou et al. 2015a], Intrusion Detection Systems (IDS) are used as a second line of defense [Onat and Miri 2005].

Nevertheless, we cannot directly apply intrusion detection techniques used in other wireless networks such as *ad hoc networks*, to supervise Wireless Industrial Sensor Networks without their adaptation. Indeed, there are three features that distinguish WSN and more specifically WISN from other wireless networks [Mitchell and Chen 2014]:

1. processor, memory, energy and channel are limited resources;
2. WISN are usually not mobile;
3. the behavior of a WISN is highly predictable since it is composed of devices with few human interventions. So, communication and exchanged data respond to specific profiles in terms of quantity and frequency.

Depending on where the intrusion detection logic is implemented, these systems can be divided in two categories [Coppolino et al. 2010]: centralized and distributed systems. In centralized systems, an IDS agent connected to the WSN, mainly through the base station, analyzes information sent from wireless sensors in order to detect potential attacks. In decentralized systems, the detection logic is implemented directly into sensors called IDS-agents. These IDS-agents monitor independently the behavior of adjacent sensors. Hybrid systems consist of a central agent connected to the main station and IDS-agents deployed among sensors. By this way, both local and end-to-end communications are analyzed.

An important issue in such architectures is the deployment of IDS-agents. Indeed, the detection efficiency depends greatly on the collected data quality. Therefore, the localization of devices used to collect data must be well studied otherwise a part of communication will not be monitored.

We must note that as in *classical* WSN, clustering techniques are widely used for providing routing features and data aggregation, IDS-agents are generally implemented in the *cluster head* [Mitchell and Chen 2014]. Nevertheless, this placement method is not efficient especially in WISN. Indeed, with this method we do not have the guaranty that all communications are monitored as only communications between sensors and cluster heads are checked by IDS-agents.

In this Chapter, we present a deployment scheme for the placement of the IDS-agent of a decentralized IDS in a Wireless Industrial Sensor Network. It presents the best trade-off between the number of used IDS-agents and the detection efficiency. We use the graph theory concept of *Dominating Set* to select nodes that will be substituted by super-nodes. Super-nodes have enhanced storage and processing capacities that allow them to act in the same way as normal sensors and also as detection agents. By this way, a virtual wireless backbone network providing intrusion detection capabilities will be created upon the WSN.

To validate the deployment scheme, communication in the context of WSN were modeled and then it was proven that this scheme fulfills the defined security requirements.

The rest of the paper is organized as follows. In Section 6.2, we discuss several intrusion detection techniques used in WSN. We show in Section 6.3 how the deployment scheme has been ignored in almost all previous studies. Section 6.4 describes security requirements and the attacker model. Our deployment scheme is detailed in Section 6.5. A formal validation of this deployment is given in Section 6.6. The performances of the proposed scheme are presented in Section 6.7. Finally, Section 6.8, presents the conclusion and future work.

6.2 Intrusion detection techniques for WISN

Several techniques are used in IDS to detect attacks such as watchdogs or local monitoring [Khalil et al. 2007], spontaneous watchdogs [Roman et al. 2006], edge self-monitoring [Dong et al. 2011][Neggazi et al. 2014], etc. These techniques rely on the broadcast nature of wireless communication. Indeed, each node is able to overhear all packets sent by nodes in its neighborhood. Nevertheless, these techniques suffer from several drawbacks [Abduvaliyev et al. 2013]. In local monitoring or watchdog, selected nodes are used for monitoring specific part of the wireless network. This technique requires that watchdog nodes overhear and store all exchanged packets in their neighborhood. Consequently, it is a very energy and computational resources consuming technique as watchdog nodes must be active continuously. In industrial process management, such technique is in practice not applicable since sensors have limited resources.

To reduce some of these drawbacks, spontaneous watchdog technique was proposed [Roman et al. 2006]. In this technique, all nodes implement a local agent that monitors information relative to the sensor node itself. Also, a global agent is activated

randomly and acts as a watchdog. Thus, as global agent is not active continuously in each node, the added overload is lower in comparison to the previous technique. However, this technique does not ensure that all packets are overheard by a global agent which reduces significantly the IDS efficiency.

In edge self-monitoring technique [Dong et al. 2011], nodes are put in sleep or active mode in such a way that each transmission link is always monitored by k nodes (*k-self monitoring*). This technique ensures that all the traffic is monitored and node resources are not overused. The drawback is that monitoring nodes have partial information about monitored nodes. Indeed, the same node is not monitored each time by the same monitoring nodes. Consequently this technique is not efficient for intrusion detection.

6.3 Related Work

After studying many intrusion detection systems specially designed for wireless sensor networks [Mitchell and Chen 2014], we can conclude that the detection logic deployment issue is rarely mentioned. Indeed, although these studies use selected sensors for implementing totally or partially an intrusion detection logic, there is no indication how these sensors are selected. In [da Silva et al. 2005], Da Silva et al. proposed one of the first intrusion detection systems for WSN. They designed a decentralized system in which a set of nodes is designated as *monitor* and is responsible of monitoring their neighbors looking for intruders. Nevertheless, it is not specified how these monitoring nodes are selected except that all nodes must be monitored.

In a more recent study, Roosta et al. proposed in [Roosta et al. 2008] an intrusion detection system for wireless process control systems. The system consists of two components: a central IDS and multiple field IDS distributed among sensor nodes. These field IDS are deployed in *super-nodes* that passively monitor communications in their neighborhood. They periodically send collected data to the central IDS that will check their conformity with the security policy. Even if it is mentioned that the central IDS is implemented in the Network Manager (i.e., the base station in this kind of networks) there is no indication about the deployment of field IDS.

We also must note that in Wireless Sensor Networks, there are two others methods that are more or less similar to the IDS deployment which are *Base Station deployment* and *Network Clustering*.

In Base Station deployment studies, the aim is to find the optimal location of one or several base stations in order to ensure the radio coverage, reduce communication latency or increase the network lifetime [Akkaya et al. 2007]. The most important cri-

teria here is the determination of the most suitable location for the base station that ensures reliable communications with nodes.

In the Network clustering studies, the aim is to organize the wireless network into a collection of small-size networks [Abbasi and Younis 2007]. This is mainly used in routing protocol or in transmission bandwidth optimization. In both cases only nodes designated as *cluster heads* implement the routing table or have the ability to aggregate received data which reduces the redundancy and the network load. Clusters are built in such a way all nodes are located at k-hops on maximum from the *cluster head* or by sitting equal-size clusters to perform load balancing [Abbasi and Younis 2007].

In IDS deployment issue, nodes selection criteria are different from those used in base station deployment or cluster heads selection. Indeed, IDS system must be deployed in such a way all exchanged packets are monitored and their conformity with the security policy is checked.

6.4 Security requirements and Attacker model

Industrial systems rely on processing the sensing data gathered from several kinds of sensors deployed throughout the facility. Therefore, to ensure the industrial process continuity in safe condition, it is important that data sent by these sensors are effectively received by the base station and in the appropriate time. In other words, we must be able to check that the right information arrives to the right destination at the right time without any modification, alteration or delay.

Consequently, we must be able to check the following security requirements:

- the packet source (non repudiation).
- the packet integrity (non modification).
- the packet delivery.
- the packet delivery time.

In this study, we assume that the aim of the attacker is to disturb the industrial process. This can be done either by dropping packets, injecting false packets or modifying packets. The attacker can also delay transmission of some important packets (alarms, sensing data, etc.) to hide its malicious activity. Therefore, we consider in our study a

Dolev-Yao like attacker [Dolev and Yao 1983] that can intercept, modify, forge or delay packets. For that it can only use his own credential (cryptographic keys) without any attack against used cryptographic mechanisms.

6.5 The proposed CDS-Based deployment scheme

In this Section, we present our CDS-Based deployment scheme for securing Wireless Industrial Sensors Networks. We use for that the *Connected Dominating Set* (CDS), a well-known concept in the Graph Theory.

The aim of this study is to propose an efficient scheme to select sensor nodes in which intrusion detection logic will be implemented. Thus, on the basis of an existing sensor network topology, selected nodes are substituted by enhanced nodes called *Super-Nodes*. These super-nodes will act as classical sensor nodes by fulfilling sensing tasks and will also implement intrusion detection capabilities. As a result, a virtual wireless backbone that provides security purposes will be created upon the network.

To achieve this goal, our approach relies on WISN communication characteristics:

1. Local communication: used by adjacent nodes, that act as relay nodes, to forward packets from the sender to their final destination. It is also used to exchange messages between adjacent nodes to maintain the network connectivity.
2. End-to-end communication: used to transmit sensing data or commands between nodes and the base station. Usually this kind of communication is encrypted.

In order to be efficient, an IDS must collect all exchanged packets in both local and end-to-end communication. Consequently, a two-level architecture is the appropriate choice. It consists in a central agent and several IDS-agents. The central agent is responsible for monitoring end-to-end communications and global coordination. It is implemented in the base station. IDS-agents are responsible for monitoring local communications of sensor nodes inside their neighborhood. They are implemented in selected sensor nodes.

Also, to be more efficient, the deployment scheme must fulfill two requirements:

1. each IDS-agent must be able to monitor its neighborhood without any cooperation with other IDS-agents. This requirement aims to decrease the overload added by the IDS and thus avoid to disturb the industrial process by adding a great amount

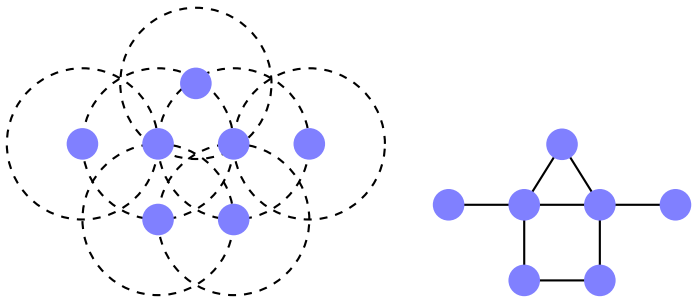
of packets. Also, it allows a better detection effectiveness since an IDS-agent can by itself detect the attack.

2. each IDS-agent must be adjacent to at least another IDS-agent. This requirement aims to ensure that we have a secure communication channel between each IDS-agent and the central-IDS. Indeed, as IDS-agents are resilient nodes, we can always trust them for forwarding alarm packets especially in the case of nodes compromission.

A final but not less important point, is that the deployment scheme must be able to fulfill above requirements with an acceptable IDS-agents number. Indeed, the implementation of the detection logic in enhanced sensors capabilities must be cost efficient by reducing the number of these sensors.

According to the above requirements, our deployment scheme, that we call CDS-based deployment scheme, includes the three following steps: (i) *Connectivity Graph Construction*: A preparatory step in which the wireless sensor network is modeled by a graph called *Connectivity Graph*. (ii) *Connected Dominating Set Construction*: In this step, the connected dominating set is computed for selecting nodes that will be substituted by *IDS-agents*. (iii) *Uncovered Links Removal*: A final step that selects additional nodes for enforcing the monitoring coverage of some links.

6.5.1 Connectivity Graph Construction



(a) Wireless Sensor Network (b) Connectivity Graph

Figure 6.1: Connectivity graph construction

In this step we model the wireless sensor network as a graph $G = (V, E)$, where V represents the set of all nodes in the network and E represents the set of all links in the network. Building the set V is straightforward as each node in the wireless networks is represented by a vertex. Modeling E is more sophisticated. Indeed, two nodes are *linked* if they can communicate which means that each node is in the transmission range of the other node.

Several models have been proposed to specify the transmission range. The most used model is the Unit disk graph (UDG). As illustrated in Figure 6.1, two nodes are adjacent if and only if their Euclidean distance is at most 1 (or in general case less than a radius r). This model is idealistic as it assumes that the transmission range is uniform and omnidirectional and does not consider obstacles.

Other models try to be more realistic by considering waves propagation and path loss. Due to its simplicity and efficiency, the COST231 multi-wall model [COST 231 1999] is widely used in indoor environment [Andrade and Hoefel 2010]. In this model, the path loss in dB for environments with just one floor is given by L_{dB}, where d represents the distance, the integer k_w represents the number of wall types, k_{wi} and L_{wi} represent respectively the number and the loss of the i^{th} wall type and $L_{0,dB}$ is the free space propagation to 1 meter.

$$L_{dB} = L_{0,dB} + 20 \log_{10} d + \sum_{i=1}^{k_{wi}} k_{wi} L_{wi} \tag{6.1}$$

The reader can refer to [Andrade and Hoefel 2010] and [Yu et al. 2013] to have more details about other models.

6.5.2 Connected Dominating Set (CDS)

In Graph Theory, a *Dominating Set* (DS) D of a graph G is a subset of nodes such that each node in G is adjacent to at least one node in D. A node in D is called a *dominator node*.

A *Connected Dominating Set* (CDS) is a dominating set in where each dominator node is adjacent to at least one other dominator.

In WSN, CDS have been used for creating a virtual backbone of the network (VBN). The VBN is mainly used as a spin for routing purposes [Yu et al. 2013]. Only nodes in the dominating set have routing features. Other nodes must send their messages to their closest dominator. Then, messages will be routed to the final destination throughout the VBN. The CDS can also be used for [Yu et al. 2013]:

- improving multicast/broadcast routing by restricting the forwarding of such messages to dominator nodes only,
- managing power consumption by making more nodes in a sleep mode,

- providing reliable and stable links.

Finding the *minimum* (connected) dominated set (M(C)DS) i.e., a CDS with the smallest size, is a NP-hard problem [Yu et al. 2013, Guha and Khuller 1998]. Therefore many algorithms for constructing an approximate M(C)DS have been proposed. These algorithms can be divided into two categories: centralized algorithms and decentralized algorithms. Centralized algorithms are used under the assumption that the complete network topology is known. In decentralized algorithms, the dominator nodes are selected after a message exchange process between nodes.

The proposed deployment scheme relies on a centralized algorithm. Firstly, because this deployment scheme will be performed off-line on a topology-known WISN and also because centralized algorithms in general yield to a smaller CDS with a better performance ratio than decentralized algorithms [Yu et al. 2013].

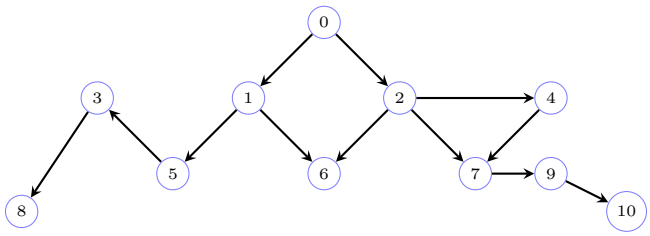
Guha and Khuller propose in [Guha and Khuller 1998], a greedy centralized algorithm to construct a *Minimal* CDS and prove that it performs in a polynomial time. This algorithm builds a spanning tree rooted at the node that has a maximum degree. Each time a node is selected as a dominator, its neighbors are marked. The marked node with the maximum degree is selected as a dominator for the next step. The tree grows until all nodes are added to it. The non-leaf nodes in the tree form a CDS.

We propose a modified algorithm based on the Guha and Khuller algorithm. Indeed, instead of starting by the node with the maximum degree, we use the node representing the base station as a tree root. The pseudo-code of the proposed algorithm is given in Algorithm 2.

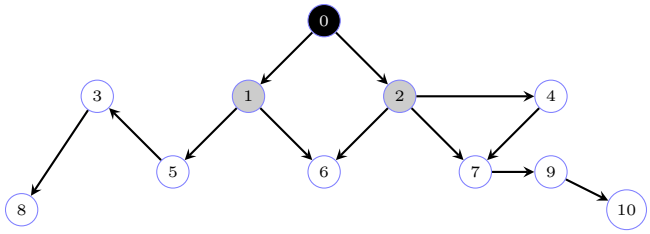
Algorithm 2 CDS construction algorithm

1: ▷ Black nodes are dominators
2: ▷ Gray nodes are neighbors of dominators
3: ▷ White nodes are not yet dominated
4: ▷ Initially all nodes are white
5: Mark the node *root* as black; ▷ Start from the base station
6: Mark *root* neighbors as gray nodes;
7: **while** Exist a white node **do** ▷ repeat until all nodes are marked
8: Select n the gray node with the maximum degree;
9: Mark n as black;
10: Mark node n neighbors as gray nodes;
11: **end while**

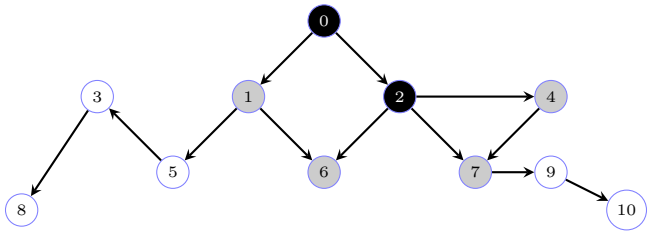
Figure 6.2 shows an example of the application of Algorithm 2.



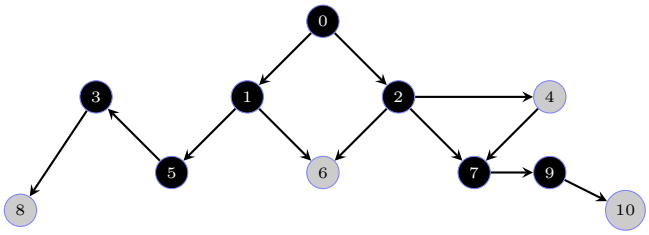
(a) Initial state



(b) Node 0 is a dominator



(c) Node 2 is a dominator



(d) Final state

Figure 6.2: Alg. 2 application example

6.5.3 Uncovered Links Removal

To detect efficiently some attacks such as selective forwarding, message injecting or dropping, the IDS-agent should overhear packets received by a node and also those transmitted by that node. By construction, the IDS-agent CDS-based deployment scheme ensures that each packet transmitted by any node in the wireless network is overheard by at least one IDS-agent.

But as illustrated in Figure 6.3, this does not guarantee that all packets received by that node are always overheard by at least one of IDS-agents monitoring the node

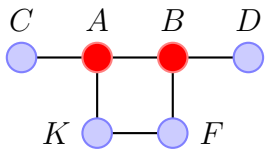


Figure 6.3: Example of an uncovered link

receiving the packet. Indeed, IDS-agent in A cannot check that K actually retransmits packets received from F and the same holds for IDS-agent B for packets received by F from K.

Thus, after applying the CDS algorithm, we *can* have some *uncovered links* as the link $K - F$ in Figure 6.3 .

Uncovered links are links that fulfill the two following conditions:

1. no one of their vertices is an IDS-agent (a dominator).
2. and also their vertices are monitored by different IDS-agents.

For monitoring these links, we implement an algorithm for their detection as illustrated in Algorithm 3. This algorithm starts by building the list of uncovered links. Then, it marks as dominator the node that is part of the maximum number of uncovered links. After updating the uncovered links list, it repeats previous actions until all uncovered links are monitored.

Algorithm 3 Uncovered links detection and monitoring algorithm

```
1: Build L the list of uncovered links;
2: while L is not empty do          ▷ repeat until all uncovered links are monitored
3:     Select n the node part of the maximum number of uncovered links;
4:     Mark n as black;
5:     Update L;
6: end while
```

6.6 Deployment scheme formal validation

To validate the proposed deployment scheme, we first define communication properties of wireless communication. Then, we specify both attacker capabilities and security requirements. Finally, we prove that the defined security requirements are completely fulfilled by the deployment scheme properties.

6.6.1 Notation :

Let us assume the following:

- V and D represent respectively, the set of nodes and the set of IDS-agents with $D \subset V$.
- M represents the set of all exchanged packets.
- $sendPacket(n_1, n_2, n_3, n_4, m, t)$ means that node n_1 sends to node n_2 the packet m originated from the node n_3 and destined to n_4 at time t.
- $receivePacket(n_1, n_2, n_3, n_4, m, t)$ means that the node n_1 receives from the node n_2 the packet m originated from the node n_3 and destined to the node n_4 at the time t.
- $neighbors(n_1, n_2)$ means that the node n_1 is the neighbor of the node n_2.
- $equivalent(m, m')$ means that packet m' is the forwarded version of the packet m and only fields in the header have been changed according to the used communication protocol.
- ϵ represents the propagation delay of a packet.
- δ represents the maximal time a packet must be forwarded within.
- δ' (with $\delta << \delta'$) represents the maximal time a packet is considered as deleted if it has not been forwarded.

We must note that in the predicates *sendPacket* and *receivePacket* the final destination of the packet m is the node n_4 and that the node n_2 is used as relay to forward this packet to its final destination.

6.6.2 Properties definitions

- WISN properties:

1. Medium broadcast property: as regards to the broadcast nature of wireless medium, a packet m sent by a node n_1 is received by all its neighbors.

$$\begin{aligned} &\forall n, n_1, n_2, n_3, n_4 \in V, \\ &sendPacket(n_1, n_2, n_3, n_4, m, t) \wedge \\ &neighbors(n_1, n) \\ &\Rightarrow receivePacket(n, n_1, n_3, n_4, m, t + \epsilon) \end{aligned}$$

2. Channel symmetry property: If node n_1 is a neighbor of node n_2, node n_2 is also a neighbor of node n_1.

$$\begin{aligned} &\forall n_1, n_2 \in V, \\ &neighbors(n_1, n_2) \Leftrightarrow neighbors(n_2, n_1) \end{aligned}$$

3. Multi-hop property: If node n_1 receives a unicast packet m originated from the node n, so either n_1 and n are neighbors, or there is a node n_2 neighbor of node n_1 that has forwarded this packet.

$$\begin{aligned} &\forall n_1, n_2, n, n' \in V, m \in M, t \in T, \\ &receivePacket(n_1, n_2, n, n', m, t) \\ &\Rightarrow (sendPacket(n_2, n_1, n, n', m, t - \epsilon) \wedge \\ &neighbors(n_2, n_1)) \end{aligned}$$

- Attacker properties:

1. Forging a fake packet: in this attack, a malicious node n_1 pretends retransmitting to n_3 a packet m received from the node n_2.

$$\begin{aligned} &\forall n_1, n_3, n, n' \in V, n_1 \neq n, m \in M, t \in T, \\ &forgePacket(n_1, n_3, n, n', m, t) \\ &\Rightarrow sendPacket(n_1, n_3, n, n', m, t) \\ &\wedge \neg(\exists m' \in M, \exists n_2 \in N, \exists t' \in T, t' < t, \\ &receivePacket(n_1, n_2, n, n', m', t') \\ &\wedge equivalent(m, m') \\ &\wedge neighbors(n_1, n_2) \wedge neighbors(n_1, n_3)) \end{aligned}$$

2. Deleting a packet: in this attack, a malicious node n_1 does not forward to the next node n_2 a received packet m destined to the node n' within the defined

time δ'.

$$\begin{aligned}
&\forall n_1, n_2, n_3, n, n' \in V, n_1 \neq n', m \in M, t \in T, \\
&deletePacket(n_1, n_2, n, n', m, t + \delta) \Rightarrow \\
&receivePacket(n_1, n_3, n, n', m, t) \\
&\wedge \neg(\exists m' \in M, \exists t' \in T, t + \epsilon < t' < t + \delta', \\
&sendPacket(n_1, n_2, n, n', m', t') \\
&\wedge equivalent(m, m') \wedge neighbors(n_1, n_2))
\end{aligned}$$

3. Modifying a packet: in this attack, a malicious node n_1 forwards to the next node n_2 a packet m' which is a modified version of a received packet m.

$$\begin{aligned}
&\forall n_1, n_2, n_3, n, n' \in V, m \in M, t \in T, \\
&modifyPacket(n_1, n_2, n, n', m, t) \Rightarrow \\
&receivePacket(n_1, n_3, n, n', m, t') \\
&\wedge (\exists m' \in M, \exists t' \in T, \\
&receivePacket(n_2, n_1, n, n', m', t)) \\
&\wedge \neg equivalent(m, m') \wedge neighbors(n_1, n_2)
\end{aligned}$$

4. Delaying a packet: in this attack, a malicious node n_1 forwards to the next node n_2 the received packet m after the defined time δ.

$$\begin{aligned}
&\forall n_1, n_2, n_3, n, n' \in V, m \in M, t \in T, \\
&delayPacket(n_1, n_2, n, n', m, t) \Rightarrow \\
&\exists m' \in M, \exists t' \in T, t' + \delta < t < t' + \delta', \\
&receivePacket(n_1, n_3, n, n', m, t') \\
&\wedge sendPacket(n_1, n_2, n, n', m', t) \\
&\wedge equivalent(m, m') \wedge neighbors(n_2, n_3)
\end{aligned}$$

- WISN Security requirements:

1. Traffic monitoring property: In order to gather all exchanged traffic, the IDS system i.e., all IDS nodes, must receive all sent messages.

$$\begin{aligned}
&\forall n_1, n_2, n, n' \in V, \forall m \in M, \forall t \in T, \\
&\exists d \in D, sendPacket(n_1, n_2, n, n', m, t) \Rightarrow \\
&receivePacket(d, n_1, n, n', m, t + \epsilon) \wedge \\
&neighbors(d, n_1)
\end{aligned}$$

2. Forged packet Detection property: an IDS-agent d receives the packet m sent by n_1 to n_2 without receiving the equivalent packet m' sent to n_1 by n_3.

$$\begin{aligned}
&\forall n_1, n_2, n_3, n, n' \in V, n_1 \neq n, \forall m \in M, \forall t \in T, \\
&\exists d \in D, \\
&detectForgedPacket(d, n_1, n_2, n, n', m, t) \\
&\Rightarrow \neg(\exists m' \in M, \exists t' \in T, t' < t, \\
&receivePacket(d, n_3, n, n', m', t') \\
&\wedge receivePacket(d, n_1, n, n', m, t) \\
&\wedge equivalent(m, m') \\
&\wedge neighbors(d, n_1) \wedge neighbors(d, n_3))
\end{aligned}$$

3. Deleted packet Detection property: an IDS-agent d receives the packet m sent by n_3 to n_1 but does not receive the equivalent packet m' forwarded by n_1 to n_2 within the defined time δ'.

$$\begin{aligned}
&\forall n_1, n_2, n_3, n, n' \in V, n_1 \neq n', \forall m \in M, \\
&\forall t, t' \in T, t < t' < t + \delta', \exists d \in D, \\
&detectDeletePacket(d, n_1, n_2, n, n', m, t + \delta') \\
&\Rightarrow \exists m' \in M, receivePacket(d, n_3, n, n', m', t) \\
&\wedge \neg receivePacket(d, n_1, n, n', m, t') \\
&\wedge equivalent(m, m') \wedge neighbors(n_1, n_3)
\end{aligned}$$

4. Modified packet Detection property: an IDS-agent d detects that the packet m sent by n_3 to the node n_1 and the packet m' forwarded by n_1 to n_2 are not equivalent.

$$\begin{aligned}
&\forall n_1, n_2, n_3, n, n' \in N, \forall m \in M, \forall t, t' \in T, \\
&\exists d \in D, \\
&detectModifyPacket(d, n_1, n_2, n, n', m, t) \\
&\Rightarrow \exists m' \in M, receivePacket(d, n_3, n, n', m, t) \\
&\wedge receivePacket(d, n_1, n, n', m', t') \\
&\wedge \neg equivalent(m, m') \\
&\wedge neighbors(n_1, n_3) \wedge neighbors(n_1, n_2)
\end{aligned}$$

5. Delayed packet Detection property: an IDS-agent d that receives the packet m sent from the node n_3 to the node n_1, also receives the packets m' that n_1

forwarded to the next node n_2 after the defined time δ.

$$\begin{aligned}
&\forall n_1, n_2, n_3, n, n' \in V, \forall m \in M, \\
&\forall t \in T, \exists d \in D, \\
&detectDelayPacket(d, n_1, n_2, n, n', m, t') \\
&\Rightarrow \exists m' \in M, \exists t' \in T, t + \delta < t' < t + \delta', \\
&receivePacket(d, n_3, n, n', m, t) \\
&\wedge receivePacket(d, n_1, n, n', m', t') \\
&\wedge equivalent(m, m')
\end{aligned}$$

- IDS Deployment properties:

  1. CDS property: a node is either an IDS or has at least one IDS as a neighbor.

$$\forall n \in V, \exists d \in D, neighbors(n, d)$$

  As each node has at least an IDS as a neighbor (CDS property) and each packet sent by a node is received by all its neighbors (the medium broadcast property), each sent packet is received by at least one IDS.

$$\begin{aligned}
&\forall n_1, n_2, n \in V, \forall m \in M, \forall t \in T, \exists d \in D, \\
&sendPacket(n_1, n_2, n_3, n, m, t) \\
&\Rightarrow receivePacket(d, n_1, n_3, n, m, t + \epsilon) \wedge \\
&neighbors(n_1, d)
\end{aligned}$$

  2. Uncovered link monitoring property: this property guarantee that there is always an IDS-agent d neighbor of two neighbor nodes n_1 and n_2.

$$\begin{aligned}
&\forall n_1, n_2 \in V, \exists d \in D, \\
&neighbors(n_1, n_2) \\
&\Rightarrow neighbors(d, n_1) \wedge neighbors(d, n_2)
\end{aligned}$$

6.6.3 Security requirements guarantee proof

Theorem 1. *Deployment scheme properties guarantee WISN security requirements validation:*

$$\forall n_1, n_2, n_3, n, n' \in V, \forall m \in M, \forall t \in T$$
$$neighbors(n_1, n_2) \wedge neighbors(n_1, n_3) \Rightarrow$$
$$\exists d \in D, detectForgePacket(d, n_1, n_2, n, n', m, t)$$
$$\wedge detectDeletePacket(d, n_1, n_2, n, n', m, t)$$
$$\wedge detectModifyPacket(d, n_1, n_2, n, n', m, t)$$
$$\wedge detectDelayPacket(d, n_1, n_2, n, n', m, t)$$

Indeed, according to *the Uncovered link monitoring property*, if n_1 sends a packet m to its neighbor n_2 there is always an IDS-agent d, neighbors of n_1 and n_2 that receives the sent packet. Also, according to the medium broadcast property, the IDS-agent d receives all packets sent by n_2 and particularly the packet m' i.e., the forwarded version of the packet m.
Consequently, the IDS-agent d can always compare packets m and m' and checks if ever a packet has been forged, deleted, modified or delayed.

6.7 Performances Evaluation

6.7.1 Dominating Nodes Ratio

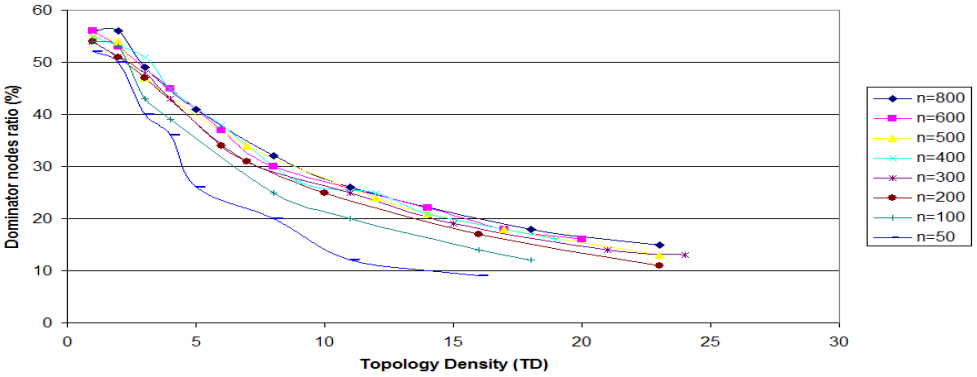


Figure 6.4: Dominating nodes ratio compared to the Topology Density

For evaluating the proposed deployment scheme performances, we conduct series of test on simulated wireless sensor networks. For this purpose, we use NS3 to deploy randomly n nodes in a rectangular field. Then, we vary the radius r, representing the transmission range of nodes. By that way, we get networks with different topology density (TD) that is the average node degree.

The dominating node ratio indicates the number of dominating nodes in a WSN compared to the total number of its nodes. For assessing the impact of the topology density on this ratio, we generate for each graph size, 50 random graphs with different topology density. Then, we measure for each generated graph, the dominating node ratio. In order to get accurate results, we measure the dominating node ratio of several generated random graphs with the same size and topology density. Then, we take the average of these measures that we illustrate in Figure 6.4.

As intuitively expected, the dominating node ratio decreases, for all graph sizes, with the increase of the topology density. This ratio is about 30% with a TD equal to 7-8 and reaches 20-25% with a TD above 10-12.

We should note, that according to the best practices in WISN deployment [HART Communication Foundation], 25% of sensors should have a direct connection to the main station; each node should have at least 3 direct neighbors; and each node should not be 4 hops away from the main station.

Table 6.1 illustrates the dominating node ratio result for Alg. 2 and Alg. 3. We can see clearly that Algorithm 3 does not add a great number of IDS-agents. Indeed, the maximum ratio of added IDS-agents is about 3.5 %. Thus, detecting and removing uncovered links strengthen the efficiency of the solution without increasing significantly the number of IDS-agents.

Table 6.1: Dominating node ratio by algorithm.

n	Alg.2		Alg. 3		Total Ratio (%)
	Result	Ratio (%)	Result	Ratio (%)	
50	24.23	48.46	0.15	0.30	48.76
100	42.38	42.38	1.15	1.15	43.53
200	72.15	35.92	4.69	2.23	38.15
300	102.33	33.75	9.08	3.00	36.75
400	109.07	27.00	13.43	3.57	30.57
500	133.69	26.46	16.84	3.30	29.76
600	149.76	24.46	21.39	3.69	28.15
800	166.78	20.42	27.29	3.50	23.92

6.7.2 Dominating nodes selection execution Time

As the CDS-Based deployment scheme is executed once and offline, it does not require a fast execution. Nevertheless, the average time taken by the execution of both Algorithm 2 and Algorithm 3, illustrated in Figure 6.5, shows that it takes very acceptable values. These performances are mainly due to the Alg. 2 that is executed in a polynomial time.

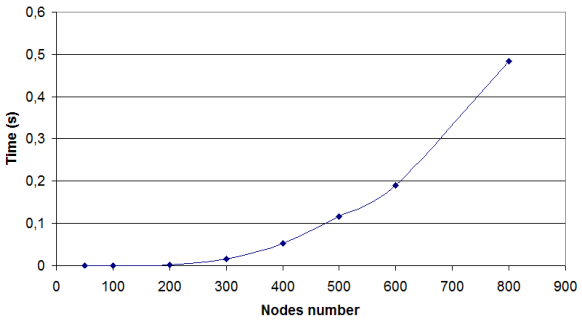


Figure 6.5: Dominating nodes selection time

6.7.3 Traffic monitoring efficiency

Table 6.2 illustrates the number of nodes monitored by the same dominator node. We can see that the average number of nodes dominated by the same dominator node is always bigger than the Topology density, i.e., the average node degree. This is due to the Algorithm 2 and Algorithm 3 that choose dominator nodes with higher degrees.

We can also see that a dominator node monitors at least 2 nodes which indicates that leaf nodes are never selected as dominating nodes.

In another hand, the maximum number of monitored nodes by the same dominator may seem higher particularly for networks with high density.

We must note in these cases that generally communication protocols for WISN use techniques such as Time Division Multiple Acces (TDMA) to manage transmission and avoid collisions. In these techniques, the bandwidth is divided into several channels (Typically 15 or 16) and only one transmission is allowed in the same channel at the same time. Consequently, the maximum number of communication that a dominator node has to monitor is equal to the number of transmission channels (15 or 16).

Table 6.2: Number of monitored nodes by a dominator.

n	TD	Number of dominated nodes		
		Min	Max	Avg
50	2.09	2.01	11.63	4.80
100	3.69	2.20	20.76	6.74
200	7.07	2.83	38.67	10.80
300	6.81	2.41	43.94	9.90
400	7.10	2.32	49.78	9.86
500	8.50	2.48	60.08	11.79
600	8.11	2.20	64.13	10.90
800	8.87	2.17	75.65	11.49

Table 6.3: Number of dominators monitoring a node.

n	TD	Number of dominated nodes		
		Min	Max	Avg
50	2.09	1	4.81	2.12
100	3.69	1	5.56	2.42
200	7.07	1	6.30	2.82
300	6.81	1	6.63	2.80
400	7.10	1	7.12	2.90
500	8.50	1	7.20	3.05
600	8.11	1	7.24	2.99
800	8.87	1	7.50	3.06

In Table 6.3, we report the number of dominator nodes that monitors the same node. As expected, all nodes are at least monitored by one dominator. We can also see that on average, a node can be monitored by 2, 3 or more dominator nodes. Thus, this increases the detection efficiency as a node is monitored by several dominator nodes.

6.8 Conclusion

In this Chapter, we have presented an efficient IDS-agent deployment scheme for wireless sensor networks. This deployment scheme can be used either in decentralized or clustered architectures. It creates a virtual backbone that adds security purposes to an existing wireless sensor network. To the best of our knowledge, it is the only complete deployment scheme implemented for security purposes. It presents good results both in terms of selected IDS-agent and execution time.

We must also note that the proposed deployment scheme fulfills totally WISN requirements especially in terms of communication specifications.

This work can be improved by different ways. For example, as several nodes are monitored by a great number of dominators, we can try to eliminate redundant dominators. Also, we can adapt the deployment scheme to be used in heterogeneous networks in which devices do not have the same capabilities in terms of transmission range, storage and computational resources. In this case, we can use weighted graphs to select nodes with higher capabilities firstly.

CHAPTER 7

Conclusion and perspectives

The main objective of this thesis was to propose solutions that ensure communication security and reliability in wireless sensor networks used in industrial environments. Therefore, we first have proposed a landscape of vulnerabilities and threats targeting industrial installations. We argue that ICS are targeted as they are used to manage several kinds of facilities that play important economic and social roles. We also have detailed attackers profiles and motivations and describe some techniques used for launching attacks against ICS. Thus, nation-states linked attackers are the most serious threat to ICS as they possess required resources and knowledge to launch harmful actions. However, the availability of several automated tools make it easier for attackers with lower skills to prepare attacks against ICS.

In our second contribution, we have studied particularly, security issues in WSN-based industrial control systems. We have assessed security mechanisms of WirelessHART protocol, a widely-used wireless industrial communication protocol. Thus, we have assessed the strength and weaknesses of implemented mechanisms used to ensure communications security. We have shown that these mechanisms have several issues that can be used for targeting such networks. Indeed, the use of shared cryptographic keys introduces several weaknesses in the communication scheme that allow malicious nodes to inject forged packets into the network.

Based on our security analysis, we have described in this thesis two attacks towards WSN-based industrial systems. The first attack allows a malicious node to disconnect partially or totally a large number of wireless nodes from the network. The second one, allows the injection of false commands into the network. These commands are executed by receiving nodes since this latter considers them as being legitimate commands sent by the network's managing device. For both of these attacks, we have provided detailed scenarios and conducted several tests to demonstrate their potential harmfulness. Fur-

thermore, we have proposed some solutions and improvements that aim to mitigate them. These were our third and forth contributions respectively

Moreover, in our fifth contribution, we have implemented WirelessHART NetSim simulator to test and validate several scenarios that use the aforementioned weaknesses. To the best of our knowledge, this simulator is the only available one that implements fully the WirelessHART protocol stack and allows conducting security tests.

On the other hand, among the security improvements that we have proposed in this thesis in order to mitigate attacks targeting wireless-based ICS, we have proposed the use of intrusion detection systems (IDS) as a second line of defense. Thus, our sixth contribution was the implementation of wIDS a specification-based IDS for WISN. Indeed, as communication in a wireless sensor networks are predictable in terms of traffic's nature and frequency, any deviation from communication specifications has very likely a malicious source. The normal expected node's behavior is build using the wirelessOrBAC formalisms. This latter uses a control access model to express in a comprehensive and easy way the security requirements of WSN. Thus, using wirelessOrBAC, we have implemented in wIDS rules derived from the specification of wirelessHART protocol.

Conducted tests have demonstrated wIDS capabilities to detect and identify a wide range of well-known attacks targeting WSN, including those described in this thesis. Thus, performed tests have reported 100% correct identification of malicious actions and less than 2% of false positives.

Finally, in our last contribution, we have proposed an efficient scheme for the deployment of IDS-agent for monitoring communications in a WSN. This scheme allows the building of a security backbone upon the wireless network and ensures its full coverage. We use for that the *Connected Dominating Set* (CDS), a well-known concept in the Graph Theory. Furthermore, to validate the deployment scheme, communications in the context of WSN were modeled and then it was proven that this scheme fulfills the defined security requirements. To the best of our knowledge, it is the only complete deployment scheme implemented for security purposes. It presents good results both in terms of the number of required IDS-agents and execution time.

In conclusion, in this thesis we have explored several issues related to the security of industrial facilities ranging from the origin of threats, through the profiles of attackers and their motivations and arriving at the analysis of used techniques. This has allowed us to better understand the security challenges of these systems in order to propose the most efficient and well-tailored solutions to mitigate these threats.

Furthermore, several research topics were explored such as protocol analysis, industrial protocols, wireless sensor networks technology, formal modeling and validation, the graph theory, etc.

7.1 Perspectives

We discuss hereafter some research topics that we aim to explore to extend the work achieved in this thesis.

7.1.1 Extend our study to other communication protocols

In this thesis, we focus our study on WirelessHART as it is the most used and the first standard for wireless industrial sensor networks. As perspective, we aim to apply the same analysis methodology to assess security mechanisms of other available protocols such as ZigBee Pro [ZigBee Alliance] and ISA 100.11a [Wireless System for Automation].

On the other hand, proposed solutions such as wIDS can also be applied for other protocols either wired or wireless.

7.1.2 Implementation of wIDS in real motes

In this thesis, security tests were conducted using the WirelessHART NetSim simulator (see Section 4.3). This allowed us to implements both attack scenarios and remediation solutions. However, a simulated environment does not reflect all real conditions that a WISN could faces. Thus, as perspective, we aim to implement and test proposed attacks and solution in real wireless sensor motes. Indeed, WISN are expected to be deployed in industrial harsh environment characterized by wide temperature range, vibrations, reflections due to metallic structures, etc. Such an environment can impact communication reliability which can increase the false-positive rate.

One of the main obstacles to this work is the accessibility of industrial wireless sensors. Indeed, these devices are provided by industrials as a "black box" that we can not access or modify its internal hardware components and software.

To bypass this difficulty, a solution is to use wireless sensors that implement the IEEE 802.15.4 protocol [IEEE 802.15.4-2006]. Indeed, as indicated in Section 3.3.2, the WirelessHART protocol implements partially the IEEE 802.15.4 physical layer

and extends its MAC layer with the add of new functionalities. The physical and the MAC headers of a WirelessHART packet are respectfully the same as those of an IEEE 802.15.4 packet. Consequently, a sensor that implements IEEE 802.15.4 is able to process WirelessHART packets. As a second step, we need to implements additional WirelessHART specific functionalities. For conducting our tests, we mainly need routing features and security mechanisms in the Network Layer. The code implemented in WirelessHART NetSim can be reused to implement these functionalities.

Furthermore, the use of a great number of sensor motes in real conditions, permits the evaluation of proposed solutions capabilities. In particular, we can for example measure if wIDS can deal and process in real-time a huge amount of traffic and its impact on its detection capabilities.

7.1.3 False data injection detection

The second perspective of this work is the false data injection issue. Indeed, an attacker can inject into the network false sensing data which could lead to harmful effects to the installation. Thus, in [Gollmann 2012], the author propose that additionally to the *confidentiality*, *integrity* and *availability* proprieties, *the veracity* should also considered as a relevant security property. Indeed, authentication and non-repudiation verify the claimed origin of an assertion but the assertion itself may be true or false. Thus, veracity property ensures that an assertion truthfully reflects the aspect it makes a statement about [Gollmann 2012][Krotofil et al. 2015].

If a sensor sends wrong sensing data, these latter will be normally processed as only the identity of the sender is checked. To achieve such an action, an attacker can either measled the target sensor about its environment or by taking the control over it.

Security mechanisms like wIDS (see Chapiter 5), ensure the identity of the sender and the packet delivery without being modified, delayed or deleted. They can also detect if a sensor usurps the identity of another sensor. However, they cannot detect if a sensor sends deliberately false sensing data.

One solution that mitigates this kind of attacks and ensures sensing data veracity is to perform consistency checks. This means comparing sensing data sent by each sensors to a prediction model [Gollmann 2012].

Several techniques have been proposed to solve this issue such as watermarking [Mo et al. 2015][Rubio-Hernán et al. 2016], correlation entropy [Krotofil et al. 2015], physical process invariants [Adepu and Mathur 2016], neural networks [Goh et al. 2017].

For our part, we aim to apply machine learning techniques to detect false data injection. Among available techniques, we choose to apply the Long Short Term Memory Networks (LSTM) neural networks [Hochreiter and Schmidhuber 1997]. The LSTM is a recurrent neural network that uses "memory cells" that allow the network to learn when to forget previous memory states or when to update the hidden states when new information is provided. Recurrent Neural Networks can learn and train long temporal sequences.

Our idea is to train an LSTM network on the information provided by each sensor. Then, the trained network is used to predict the next value to be returned by the sensor. Finally, the returned value is compared to the predicted one. An alert is raised if the two values differ significantly.

7.1.4 Explore the security of IoT in industry

As indicated in Section 2.2, Industrial Control Systems are entering in a new era which one of the main characteristics is the heavy use of Internet of Things (IoT) devices. This technology adds new services that increase ICS sensing and monitoring capabilities. On the other hand, IoT devices have enhanced storage, processing and connectivity resources that make them more powerful than traditional sensors. Consequently, we cannot directly apply security solutions tailored for WSN to IoT without their adaptation.

Therefore, we aim to study IoT communication protocols in the light of their use in industrial environments and propose solution that consider their inherent characteristics.

APPENDIX A

Résume en français: Évaluation et mise en œuvre de la sécurité dans les systèms SCADA à base de réseaux de capteurs sans fil

A.1 Introduction

Les systèmes de contrôle industriel (SCI) sont des systèmes informatisés utilisés pour la surveillance et la gestion d'installations industrielles. Nous pouvons trouver de tels systèmes dans les aéroports, les centrales électriques, les raffineries de gaz, etc. L'architecture de ces systèmes repose sur plusieurs capteurs et actionneurs déployés sur l'ensemble de l'installation industrielle. Les capteurs sont responsables de la collecte de différents types d'informations sur le processus industriel, telles que la température, la pression, le débit, etc. Ces informations sont envoyées à un contrôleur qui les traite et renvoie des commandes aux actionneurs. Ainsi, un actionneur peut par exemple ouvrir une vanne pour augmenter le débit d'un composant chimique ou arrêter une pompe lorsque un réservoir est rempli.

La sécurité dans les systèmes de contrôle industriel est une préoccupation majeure. En effet, ces systèmes gèrent des installations qui jouent un rôle économique important. En outre, les attaques contre ces systèmes peuvent non seulement entraîner des pertes économiques, mais aussi menacer des vies humaines.

Par conséquent, et comme ces systèmes dépendent des données collectées, il devient évident qu'en plus des exigences temps réel, il est important de sécuriser les canaux de communication entre ces capteurs et les contrôleurs principaux. Ces problèmes sont plus difficiles à résoudre dans les réseaux de capteurs sans fil (WSN), car l'utilisation des communications sans fil ajoute ses propres faiblesses en matière de sécurité.

Dans ce cadre, cette thèse a pour but d'aborder les questions de sécurité des WSN. Tout d'abord, nous effectuons une étude de sécurité en profondeur du protocole WirelessHART. Ce dernier est le protocole leader pour les réseaux de capteurs industriels sans fil (WISN) et est la première norme internationale approuvée. Nous évaluons ses forces et soulignons ses faiblesses et ses limites. En particulier, nous décrivons deux vulnérabilités de sécurité nuisibles dans le schéma de communication du protocole WirelessHART et proposons des améliorations afin de les atténuer.

Ensuite, nous présentons wIDS, un système de détection d'intrusion multicouches qui se base sur les spécifications, spécialement développé pour les réseaux de capteurs industriels sans fil. L'IDS proposé vérifie la conformité de chaque action effectuée par un nœud sans fil sur la base d'un modèle formel du comportement normal attendu.

A.2 Panorama des vulnérabilités et des menaces visant les installations industrielles

Le Système de contrôle industriel (SCI) est un terme général qui englobe plusieurs types de systèmes de contrôle, y compris les systèmes de contrôle de supervision et d'acquisition de données (SCADA), les systèmes de contrôle distribué (DCS) et d'autres configurations de systèmes de contrôle comme les automates programmables (PLC) que l'on trouve souvent dans les secteurs industriels et les infrastructures critiques [Stouffer et al. 2015, National Communications System 2004].

Historiquement, le SCI était très différent des systèmes informatiques. Ainsi, les SCI étaient des systèmes isolés exécutant des protocoles propriétaires utilisant du matériel et des logiciels dédiés. Cependant, comme ces systèmes ont commencé à adopter des solutions informatiques afin d'améliorer la connectivité de l'entreprise et les capacités d'accès à distance, ils commencent à ressembler de plus en plus à des systèmes informatiques.

Néanmoins, les SCI présentent encore de nombreuses caractéristiques qui diffèrent des systèmes informatiques traditionnels, y compris des risques et des priorités différents [Stouffer et al. 2015, Zhu et al. 2011].

Notamment, les SCI fonctionnent en continu avec peu de temps d'arrêt, qu'ils sont conçus pour répondre à des performances élevées en termes de fiabilité et de sécurité, et qu'on s'attend à ce qu'ils fonctionnent pendant 10 ou 20 ans.

En outre, les SCI ont des exigences de performance et de fiabilité différentes, et utilisent également des systèmes d'exploitation et des applications qui peuvent être considérés comme non conventionnels dans un environnement de réseau informatique typique [Lemay and Fernandez 2013].

Pour ces raisons, les mécanismes de sécurité traditionnels utilisés en informatique doivent être adaptés avant d'être déployés dans les systèmes SCADA.

Pendant longtemps, les attaques contre les systèmes SCADA ont semblé faire partie de la science-fiction. En effet, les systèmes SCADA étaient considérés comme des réseaux sécurisés. Ainsi, des croyances largement partagées étaient [Pietre-Cambacedes et al. 2011] : que personne ne veut attaquer ces systèmes ; ils sont isolés des réseaux externes ; ils utilisent des protocoles obscurs connus seulement par les experts ; et les mécanismes de sécurité intégrés tels que la cryptographie assurent un niveau de sécurité élevé.

Cependant, au cours des dernières décennies, les systèmes SCADA ont été confrontés à des défis de sécurité auxquelles ils n'avaient pas été initialement conçus pour y faire face [Anton et al. 2017]. Cette situation est principalement due aux évolutions technologiques et architecturales suivantes [Stouffer et al. 2015, Igure et al. 2006] :

- L'augmentation de l'interconnectivité des réseaux.
- Le passage de l'utilisation de normes propriétaires pour les protocoles de communication SCADA à des normes internationales ouvertes.
- L'utilisation d'équipements et des technologies sur étagère (COTS).

Depuis 1997 et la divulgation des premières vulnérabilités des SCI, le nombre de vulnérabilités d'ecouverte des composants du SCI a considérablement augmenté [Anton et al. 2017, Byres et al. 2004].

Depuis les dernières décennies, les attaques ciblant les SCI non seulement ne cessent d'augmenter mais aussi de changer et d'évoluer [Anton et al. 2017]. En effet, jusqu'en 2000, près de 70 % des incidents signalés étaient dus soit à des accidents, soit à des employés mécontents [Byres et al. 2004]. Depuis 2001, en plus de l'augmentation continue du nombre d'attaques, les rapports indiquent également que près de 70 % des incidents étaient dus à des attaques provenant de l'extérieur du réseau SCADA [Igure et al. 2006, Anton et al. 2017].

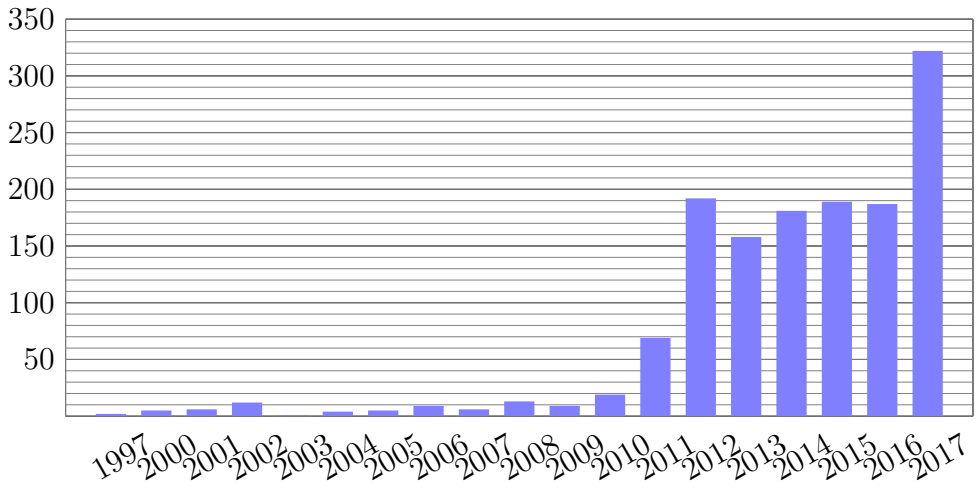


Figure A.1: Vulnérabilités d'ecouverte par année

A.3 Analyse de la sécurité des réseaux de capteurs sans fil industriels : Cas du protocol WirelessHART

Dans cette thèse, nous proposons une étude approfondie de la sécurité du protocole WirelessHART. Ce dernier est le protocole leader pour les réseaux de capteurs industriels sans fil (WISN) et est la première norme internationale approuvée. Nous donnons une description détaillée de ses mécanismes de sécurité. Nous montrons comment ces mécanismes sont utilisés avec d'autres mécanismes non sécuritaires pour assurer les exigences de sécurité. Ensuite, nous évaluons leurs forces et soulignons leurs faiblesses et leurs limites.

WirelessHART [HART Communication Foundation] est un important protocole sans fil développé par HART Communication Foundation pour l'automatisation des processus industriels. Il est inclus dans la version 7 de la norme HART, un protocole filaire largement utilisé dans l'industrie. Il a été publié en 2007 et a été approuvé en tant que norme internationale IEC 62591 en 2010. Il utilise une architecture de maillage synchronisée, auto-organisée et auto-réparatrice pour fournir une communication fiable, sécurisée et en temps réel.

WirelessHART [HART Communication Foundation] a été développé pour fournir des communications fiables et sécurisées pour les besoins de l'automatisation des processus industriels. En particulier, la sécurité est l'une de ses caractéristiques importantes. Par conséquent, il met en œuvre plusieurs mécanismes pour assurer la confidentialité,

l'authenticité et l'intégrité des données dans les transmissions de saut par saut et de bout en bout.

La sécurité de transmission hop-by-hop est assurée par la couche de liaison de données (DLL) à l'aide d'une clé cryptographique appelée "Network Key" partagée par tous les dispositifs faisant partie du réseau sans fil. Elle protège contre les attaquants qui sont en dehors du réseau et ne partagent pas son secret (Attaquants extérieurs). La sécurité de bout en bout est assurée par la couche réseau (NL) à l'aide d'une clé cryptographique appelée "Session Key" connue uniquement par les deux dispositifs communicants. Elle protège contre les attaquants qui peuvent se trouver sur le chemin du réseau entre la source et la destination (Attaquants internes).

Ainsi, les mécanismes de sécurité WirelessHART sont capable d'atténuer un grand nombre d'attaques de sécurité. Cependant, ces mécanismes ne sont pas conçus pour faire face à des attaques massives telles qu'une attaque de brouillage sur tous les canaux de transmission ou une attaque DoS lourde.

D'autre part, ces mécanismes de sécurité reposent principalement sur des opérations cryptographiques qui utilisent la même clé. Par conséquent, le contournement de ce mécanisme permettra à un attaquant d'affaiblir les autres. Cela rompt le principe de sécurité en profondeur.

Enfin, nous pouvons noter que les attaques Sybil et Broadcast, deux attaques spécialement conçues pour cibler les réseaux WirelessHART, sont capables de contourner son mécanisme de sécurité. Ces attaques peuvent avoir des conséquences néfastes sur le fonctionnement du réseau.

Aussi, nous montrons que, bien que WirelessHART mette en place plusieurs mécanismes pour assurer les exigences de sécurité en termes d'authentification, de disponibilité, de confidentialité et de non-répudiation, il reste vulnérable à une large palettes d'attaques. Ceci résulte principalement de l'utilisation de clés cryptographiques partagées connues de tous les nœuds appartenant au réseau.

D'autre part, les solutions proposées ne préviennent pas totalement toutes les attaques possibles. Ainsi, sauf à modifier profondément le schéma de communication mis en œuvre par WirelessHART, l'utilisation d'un système de détection d'intrusion (IDS) est la meilleure façon opérationnelle de détecter et de prévenir les attaques.

A.4 Problématiques de sécurité du protocole WirelessHART

Sur la base de ses faiblesses, nous présentons deux attaques contre WirelessHART : une attaque Sybil qui peut isoler un grand nombre de capteurs du réseau et l'attaque broadcast qui permet à un attaquant interne d'injecter de fausses commandes dans le réseau.

Afin de prouver la faisabilité de ces attaques et d'évaluer leur impact potentiel sur le fonctionnement du processus industriel, nous mettons d'abord en place un simulateur dédié aux études de sécurité du protocole WirelessHART.

Ainsi, nous donnons la première description d'une attaque Sybil spécialement conçue pour cibler un réseau WirelessHART. Cette attaque peut causer des dommages nuisibles à l'installation en déconnectant partiellement ou entièrement les capteurs sans fil du système SCADA. Conduite contre des installations réelles, une telle attaque peut perturber profondément son fonctionnement et peut conduire à l'arrêter ou plus encore induire sa destruction.

Selon la norme WirelessHART [HART Communication Foundation], un appareil peut être déconnecté par le gestionnaire de réseau ou se déconnecter lui-même ou simplement mourir. Dans le premier cas, le gestionnaire de réseau envoie une commande de déconnexion (960) à l'appareil, tandis que dans le second cas, l'appareil envoie un DLPDU de déconnexion pour informer ses voisins qu'il quitte le réseau. Cette DLPDU provient de la couche de liaison de données et est sécurisée par la clé réseau. Il est transmis dans la première liaison disponible.

Dans WirelessHART, la sécurité est assurée par deux clés cryptographiques. La clé réseau qui protège contre les attaques extérieures et la Clé de Session qui se défend contre les attaques intérieures. L'utilisation de ces clés a pour but de fournir une défense en profondeur contre les menaces de sécurité sans fil. Nous décrivons ici une attaque nuisible ne nécessitant que la clé réseau et utilisant la déconnexion DLPDU.

Une attaque de déconnexion est une attaque sybil dans laquelle un attaquant usurpe l'identité d'un dispositif légitime en falsifiant un Disconnect DLPDU et en fixant l'adresse source à l'adresse du dispositif ciblé. En conséquence, le périphérique ciblé sera déconnecté du réseau puisque ses voisins l'enlèveront de leurs tables. Cette attaque est se base sur le fait que le DLPDU de déconnexion provient de la couche liaison de données et que tous les dispositifs du réseau partagent la même clé (clé réseau) pour générer et valider le code d'intégrité de message (MIC) dans la DLL.

L'idée de la deuxième attaque est qu'un attaquant interne malveillant utilise ses propres informations d'identification pour contourner le mécanisme d'authentification et injecte de fausses commandes dans le réseau. Ces fausses commandes seront authentifiées en tant que commandes légitimes et exécutées par les dispositifs de réception. Selon la nature des fausses commandes injectées, les conséquences sur le réseau peuvent être plus ou moins dommageables.

En effet, les communications de bout en bout sont sécurisées par des clés de session. Dans les communications unicast, la clé de session n'est connue que par les deux dispositifs communicants, alors que dans les communications broadcast, la clé de session est partagée par tous les dispositifs connectés au réseau.

Par conséquent, pour lancer l'attaque par injection de commande, l'attaquant interne malveillant utilisera les informations d'identification de session de diffusion pour effectuer ce type d'attaque. En effet, en tant que partie du réseau, le nœud malveillant est configuré avec *la clé de session de diffusion* et le *compteur de session*.

L'attaque par injection de commande peut être exécutée de plusieurs manières telles qu'une attaque par injection de commande directe, une attaque par injection de commande par rebond et une attaque par injection de commande à la volée.

A.5 wIDS un système de détection d'intrusion multicouches pour les réseaux de capteurs sans fil industriels

wIDS un système de détection d'intrusion multicouche sur la base des spécifications du protocole de communication, spécialement conçu pour les réseaux de capteurs industriels sans fil. L'IDS proposé vérifie la conformité de chaque action effectuée par un nœud sans fil vers le modèle formel du comportement normal attendu. Pour ce faire, des règles de contrôle d'accès sont utilisées pour modéliser les actions autorisées qu'un nœud sans fil peut effectuer. Ces règles sont principalement construites sur la base des spécifications de chaque couche du protocole de communication, de la localisation du nœud et de la configuration du processus industriel. Ils prennent également en compte les capacités et les limites des nœuds sans fil. Ainsi, en spécifiant la politique de sécurité à un niveau abstrait, nous sommes en mesure de définir et de gérer des règles de sécurité plus précises et plus efficaces indépendamment des nœuds et des caractéristiques du réseau telles que la nature et la densité des capteurs ou la topologie du réseau. Ensuite, ces caractéristiques sont utilisées plus tard lors de l'élaboration de règles de sécurité

concrètes. En plus des alertes qui sont déclenchées par des actions s'écartant du modèle normal, nous définissons des règles d'intrusion supplémentaires qui visent à détecter les actions de base de l'attaquant telles que l'injection, la suppression, la modification et le retardement des paquets.

Les approches de détection d'intrusion basées sur les spécifications définissent formellement le modèle de comportement légitime et déclenchent des alertes d'intrusion lorsque les actions de l'utilisateur s'écartent du modèle [Mitchell and Chen 2013, Mitchell and Chen 2014]. Les WISN sont composés de nœuds dont le comportement est prévisible et qui implique peu d'interactions humaines. Par conséquent, sur la base des spécifications du protocole de communication, de la configuration du processus et des capacités des nœuds sans fil, nous pouvons construire un modèle précis du comportement des nœuds attendus.

Il convient également de noter que les systèmes de détection d'intrusion basés sur des spécifications ne nécessitent aucune étape de formation. Ils peuvent donc être appliqués et utilisés directement.

Les tests effectués rapportent 100% d'identification correcte des actions malveillantes et moins de 2 % de faux positifs. En fonction de la règle de sécurité violée, les taux de faux positifs sont d'environ 0% pour les attaques sybil ou broadcast et d'environ 5% pour les attaques de brouillage, de DoS ou de retard forcé. En effet, les premières attaques sont composées d'actions clairement identifiées comme malveillantes tandis que les secondes attaques peuvent être assimilées à des perturbations transitoires de transmission telles que des interférences. Ce taux peut être réduit par l'utilisation d'un seuil.

A.6 Application de l'ensemble dominant connecté pour la sécurisation des réseaux de capteurs sans fil

Selon l'endroit où la logique de détection d'intrusion est mise en œuvre, ces systèmes peuvent être divisés en deux catégories [Coppolino et al. 2010] : systèmes centralisés et distribués. Dans les systèmes centralisés, un agent IDS connecté au WSN, principalement par l'intermédiaire de la station de base, analyse les informations envoyées par des capteurs sans fil afin de détecter les attaques potentielles. Dans les systèmes décentralisés, la logique de détection est implémentée directement dans des capteurs appelés agents IDS. Ces agents IDS surveillent le comportement des capteurs adjacents. Les

systèmes hybrides se composent d'un agent central connecté à la station principale et d'agents IDS déployés parmi les capteurs. De cette façon, les communications locales et de bout en bout sont analysées.

Une question importante dans de telles architectures est le déploiement d'agents IDS. En effet, l'efficacité de la détection dépend largement de la qualité des données collectées. Par conséquent, la localisation des dispositifs utilisés pour recueillir des données doit être bien étudiée, sinon une partie de la communication ne sera pas surveillée.

Ainsi nous présentons un schéma de déploiement pour le placement de l'agent IDS d'un IDS décentralisé dans un réseau de capteurs industriels sans fil. Il présente le meilleur compromis entre le nombre d'agents IDS utilisés et l'efficacité de détection. Nous utilisons le concept de théorie des graphes de *Dominating Set* pour sélectionner les nœuds qui seront remplacés par des super-nœuds. Les super-nœuds ont des capacités de stockage et de traitement améliorées qui leur permettent d'agir de la même manière que les capteurs normaux et aussi en tant qu'agents de détection. De cette façon, un réseau dorsal virtuel sans fil offrant des capacités de détection d'intrusion sera créé sur le WSN.

Conformément aux exigences ci-dessus, notre schéma de déploiement, que nous appelons schéma de déploiement basé sur CDS, comprend les trois étapes suivantes : (i) *Connectivity Graph Construction* : Une étape préparatoire dans laquelle le réseau de capteurs sans fil est modélisé par un graphe appelé *Connectivity Graph*. (ii) *Connected Dominating Set Construction* : Dans cette étape, l'ensemble dominant connecté est calculé pour sélectionner les nœuds qui seront substitués par des *IDS-agents*. (iii) *Uncovered Links Removal* : Une étape finale qui sélectionne des nœuds supplémentaires pour renforcer la couverture de surveillance de certains liens.

A.7 Conclusion

L'objectif principal de cette thèse était de proposer des solutions qui assurent la sécurité et la fiabilité des communications dans les réseaux de capteurs sans fil utilisés dans les environnements industriels.

En conclusion, dans cette thèse nous avons exploré plusieurs problematiques liées à la securité des installations industrielles allant de l'origine des menaces, en passant par le profils des attaquants et leurs motivation et arrivant à l'analyse des techniques utilisées. Cela nous a permis de mieux comprendre les enjeux de sécurité de ces systèmes afin d'apporter les solutions les plus à même de bloquer ces menaces.

De plus, plusieurs sujets de recherche étaient explorés tels que l'analyse des protocoles, les protocoles industriels, la technologie des réseaux de capteurs sans fil, la modélisation et la validation formelle, la théorie des graphes, etc.

Dans cette section, nous discutons de la façon dont nos contributions peuvent être améliorées avec de nouvelles orientations de recherche. En effet, en plus des propositions d'amélioration fournies à la fin de chacune de nos contributions, nous abordons ci-après quelques sujets de recherche que nous souhaitons explorer afin d'étendre le travail réalisé dans cette thèse.

Ainsi, en tant que perspective, nous visons à mettre en œuvre et à tester les attaques et les solutions proposées dans de véritables capteurs sans fil. En effet, les WISN sont destinés à être déployés dans des environnements industriels difficiles caractérisés par une large plage de température, des vibrations, des réflexions dues à des structures métalliques, etc. Un tel environnement peut avoir un impact sur la fiabilité de la communication, ce qui peut augmenter le taux de faux positifs.

Aussi dans cette thèse, nous nous sommes concentrés sur l'étude du protocole WirelessHART car c'est le standard le plus utilisé et le premier standard pour les réseaux de capteurs industriels sans fil. En tant que deuxième perspective, nous visons à appliquer la même méthodologie d'analyse pour évaluer les mécanismes de sécurité d'autres protocoles disponibles tels que ZigBee Pro [ZigBee Alliance] et ISA 100.11a [Wireless System for Automation].

D'autre part, les solutions proposées telles que wIDS peuvent également s'appliquer à d'autres protocoles filaires ou sans fil.

La troisième perspective de ce travail est la question de l'injection de fausses données. En effet, un attaquant peut injecter dans le réseau de fausses données de détection qui pourraient avoir des effets néfastes sur l'installation.

Notre objectif est d'appliquer des techniques d'apprentissage machine pour détecter les fausses injections de données. Parmi les techniques disponibles, nous choisissons d'appliquer les réseaux de mémoire à long terme et à court terme (LSTM) [Hochreiter and Schmidhuber 1997]. Le LSTM est un réseau neuronal récurrent qui utilise des "cellules de mémoire" qui permettent au réseau d'apprendre quand oublier les états de mémoire précédents ou quand mettre à jour les états cachés lorsque de nouvelles informations sont fournies. Les réseaux de neurone récurrents peuvent apprendre et entraîner de longues séquences temporelles.

Enfin, les systèmes de contrôle industriels sont entrés dans une nouvelle ère dont l'une des principales caractéristiques est la forte utilisation des dispositifs de l'Internet

des objets (IoT). Cette technologie ajoute de nouveaux services qui augmentent les capacités de détection et de surveillance du SCI. D'autre part, les dispositifs IoT ont des ressources de stockage, de traitement et de connectivité améliorées qui les rendent plus puissants que les capteurs traditionnels. Par conséquent, nous ne pourrons pas appliquer directement nos solutions de sécurité dédiées au WSN à l'IoT sans une adaptation.

Ainsi, notre quatrième perspective est d'étudier les protocoles de communication IoT à la lumière de leur utilisation dans des environnements industriels et de proposer des solutions qui tiennent compte de leurs caractéristiques inhérentes.

List of Publications

International Journals

- L. Bayou, D. Espes, N. Cuppens-Boulahia, and F. Cuppens, "Security analysis of WSN-based SCADA systems: Case of the WirelessHART protocol", ***under review***.
- L. Bayou, D. Espes, N. Cuppens-Boulahia, and F. Cuppens, "Industrial Control Systems under attacks: A Landscape of vulnerabilities and threats targeting Industrial Control Systems", ***under review***.

International Conferences

- L. Bayou, D. Espes, N. Cuppens-Boulahia, and F. Cuppens, "Security Issue of WirelessHART Based SCADA Systems", in Risks and Security of Internet and Systems - 10th International Conference, **CRiSIS 2015**, Mytilene, Lesbos Island, Greece, July 20-22, 2015, Revised Selected Papers, 2015, vol. 9572, pp. 225-241.
- L. Bayou, D. Espes, N. Cuppens-Boulahia, and F. Cuppens, "WirelessHART NetSIM: A WirelessHART SCADA-Based Wireless Sensor Networks Simulator", Secur. Ind. Control Syst. Cyber Phys. Syst. - First Work. **CyberICS 2015** First Work. WOS-CPS 2015, Vienna, Austria, Sept. 21-22, 2015, Revis. Sel. Pap., vol. 9588, pp. 63-78, 2015.
- L. Bayou, N. Cuppens-Boulahia, D. Espes, and F. Cuppens, "Towards a CDS-based Intrusion Detection Deployment Scheme for Securing Industrial Wireless Sensor Networks", in 11th International Conference on Availability, Reliability and Security, **ARES 2016**, Salzburg, Austria, August 31 - September 2, 2016, 2016, pp. 157-166.

- L. Bayou, D. Espes, N. Cuppens-Boulahia, and F. Cuppens, "Security Analysis of WirelessHART Communication Scheme", in Foundations and Practice of Security - 9th International Symposium, **FPS 2016**, Québec City, QC, Canada, October 24-25, 2016, Revised Selected Papers, 2016, vol. 10128, pp. 223-238.

- L. Bayou, D. Espes, N. Cuppens-Boulahia, and F. Cuppens, "wirelessOrBAC:Towards an access-control-based IDS for wireless sensor networks", in Proceedings of the 2017 the 7th International Conference on Communication and Network Security - **ICCNS 2017**, 2017, pp. 96-103.

- L. Bayou, D. Espes, N. Cuppens-Boulahia, and F. Cuppens, "wIDS: A Multilayer IDS for Wireless-Based SCADA Systems", in Information Systems Security - 13th International Conference, **ICISS 2017**, Mumbai, India, December 16-20, 2017, Proceedings, 2017, vol. 10717, pp. 387-404.

Bibliography

[Abbasi and Younis 2007] A. A. Abbasi and M. Younis. A survey on clustering algorithms for wireless sensor networks. *Computer Communications*, 30(14,Äì15):2826 – 2841, 2007. Network Coverage and Routing Schemes for Wireless Sensor Networks. 115

[Abduvaliyev et al. 2013] A. Abduvaliyev, A. K. Pathan, J. Zhou, R. Roman, and W. Wong. On the vital areas of intrusion detection systems in wireless sensor networks. *IEEE Communications Surveys and Tutorials*, 15(3):1223–1237, 2013. 56, 113

[Adepu and Mathur 2016] S. Adepu and A. Mathur. Using process invariants to detect cyber attacks on a water treatment system. In *IFIP Advances in Information and Communication Technology*, volume 471, pages 91–104, may 2016. Springer, Cham. 136

[Akkaya et al. 2007] K. Akkaya, M. Younis, and W. Youssef. Positioning of base stations in wireless sensor networks. *IEEE Communications Magazine*, 45(4):96–102, April 2007. 114

[Akyildiz and Vuran 2010] I. Akyildiz and M. Vuran. *Wireless Sensor Networks*. Advanced Texts in Communications and Networking. Wiley, 2010. 93

[Akyildiz et al. 2002] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393–422, 2002. 93

[Alcaraz and Lopez 2010] C. Alcaraz and J. Lopez. A security analysis for wireless sensor mesh networks in highly critical systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 40(4):419–428, 2010. 31, 59

[Amin 2002] M. Amin. Security challenges for the electricity infrastructure. *Computer*, 35(4):8–10, Apr 2002. 20

[Andrade and Hoefel 2010] C. B. Andrade and R. P. F. Hoefel. IEEE 802.11 WLANs: A comparison on indoor coverage models. In *Proceedings of the 23rd Canadian Conference on Electrical and Computer Engineering, CCECE 2010, Calgary, Alberta, Canada, 2-5 May, 2010*, pages 1–6, 2010. IEEE. 118

[Anton et al. 2017] S. D. Anton, D. Fraunholz, C. Lipps, F. Pohl, M. Zimmermann, and H. D. Schotten. Two decades of scada exploitation: A brief history. In *2017 IEEE Conference on Application, Information and Network Security (AINS)*, pages 98–104, Nov 2017. 13, 14, 15, 24, 141

[Auerswald et al. 2005] P. E. Auerswald, L. M. Branscomb, and T. M. L. Porte. The Critical Infrastructure Challenge. pages 3–16, 2005. 11

[AutoSploit 2018] AutoSploit. Automated mass exploiter. 2018. [Online; accessed 13-March-2018]. 23

[Bakhshi 2017] T. Bakhshi. Social engineering: Revisiting end-user awareness and susceptibility to classic attack vectors. In *2017 13th International Conference on Emerging Technologies (ICET)*, pages 1–6, dec 2017. IEEE. 25

[Bayou et al. 2016a] L. Bayou, N. Cuppens-Boulahia, D. Espes, and F. Cuppens. Towards a CDS-based Intrusion Detection Deployment Scheme for Securing Industrial Wireless Sensor Networks. In *11th International Conference on Availability, Reliability and Security, ARES 2016, Salzburg, Austria, August 31 - September 2, 2016*, pages 157–166, 2016. IEEE Computer Society. 3, 56

[Bayou et al. 2015a] L. Bayou, D. Espes, N. Cuppens-Boulahia, and F. Cuppens. Security issue of wirelesshart based SCADA systems. In C. Lambrinoudakis and A. Gabillon, editors, *Risks and Security of Internet and Systems - 10th International Conference, CRiSIS 2015, Mytilene, Lesbos Island, Greece, July 20-22, 2015, Revised Selected Papers*, volume 9572 of *Lecture Notes in Computer Science*, pages 225–241, 2015. Springer. 3, 44, 52, 112

[Bayou et al. 2015b] L. Bayou, D. Espes, N. Cuppens-Boulahia, and F. Cuppens. WirelessHART NetSIM: A WirelessHART SCADA-Based Wireless Sensor Networks Simulator. In A. Bécue, N. Cuppens-Boulahia, F. Cuppens, S. K. Katsikas, and C. Lambrinoudakis, editors, *Security of Industrial Control Systems and Cyber Physical Systems - First Workshop, CyberICS 2015 and First Workshop, WOS-CPS 2015, Vienna, Austria, September 21-22, 2015, Revised Selected Papers*, volume 9588 of *Lecture Notes in Computer Science*, pages 63–78, 2015. Springer. 3, 107

[Bayou et al. 2016b] L. Bayou, D. Espes, N. Cuppens-Boulahia, and F. Cuppens. Security analysis of wirelesshart communication scheme. In F. Cuppens, L. Wang, N. Cuppens-Boulahia, N. Tawbi, and J. García-Alfaro, editors, *Foundations and Practice of Security - 9th International Symposium, FPS 2016, Québec City, QC, Canada, October 24-25, 2016, Revised Selected Papers*, volume 10128 of *Lecture Notes in Computer Science*, pages 223–238, 2016. Springer. 3, 44, 45, 53, 56

[Bayou et al. 2017a] L. Bayou, D. Espes, N. Cuppens-Boulahia, and F. Cuppens. wids: A multilayer IDS for wireless-based SCADA systems. In R. K. Shyamasundar, V. Singh, and J. Vaidya, editors, *Information Systems Security - 13th International Conference, ICISS 2017, Mumbai, India, December 16-20, 2017, Proceedings*, volume 10717 of *Lecture Notes in Computer Science*, pages 387–404, 2017. Springer. 3, 56

[Bayou et al. 2017b] L. Bayou, D. Espes, N. Cuppens-Boulahia, and F. Cuppens. WirelessOrBAC: Towards an access-control-based IDS for wireless sensor networks. In *ACM International Conference Proceeding Series*, volume Part F1338, 2017. 3

[Benenson et al. 2005] Z. Benenson, F. C. Gartner, and D. Kesdogan. An algorithmic framework for robust access control in wireless sensor networks. In *Proceeedings of the Second European Workshop on Wireless Sensor Networks, 2005.*, pages 158–165, Jan 2005. 94

[Biasi et al. 2008] M. D. Biasi, C. Snickars, K. Landern, and A. Isaksson. Simulation of process control with wirelesshart networks subject to clock drift. *2014 IEEE 38th Annual Computer Software and Applications Conference*, 0:1355–1360, 2008. 60

[Blumbergs 2014] B. Blumbergs. Technical analysis of advanced threat tactics targeting critical information infrastructure. In *cybersecurity review*, pages 25–36, 2014. 24

[Byres et al. 2004] E. Byres, P. E. Dr, and D. Hoffman. The myths and facts behind cyber security risks for industrial control systems. In *In Proc. of VDE Kongress*, 2004. 14, 15, 141

[Cherepanov 2016] A. Cherepanov. BlackEnergy by the SSHBearDoor: attacks against Ukrainian news media and electric industry. 2016. [Online; accessed 13-March-2018]. 18

[Coppolino et al. 2010] L. Coppolino, S. D'Antonio, L. Romano, and G. Spagnuolo. An intrusion detection system for critical information infrastructures using wireless sensor network technologies. In *Critical Infrastructure (CRIS), 2010 5th International Conference on*, pages 1–8, Sept 2010. 30, 89, 112, 146

[COST 231 1999] European Commission. Digital mobile radio towards future generations systems. Final report, 1999. 118

[Cuppens and Cuppens-Boulahia 2008] F. Cuppens and N. Cuppens-Boulahia. Modeling contextual security policies. *Int. J. Inf. Sec.*, 7(4):285–305, 2008. 96

[Cuppens et al. 2004] F. Cuppens, N. Cuppens-Boulahia, T. Sans, and A. Miège. A formal approach to specify and deploy a network security policy. In T. Dimitrakos and F. Martinelli, editors, *Formal Aspects in Security and Trust: Second IFIP TC1 WG1.7 Workshop on Formal Aspects in Security and Trust (FAST), an event of the 18th IFIP World Computer Congress, August 22-27, 2004, Toulouse, France*, volume 173 of *IFIP*, pages 203–218, 2004. Springer. 92

[da Silva et al. 2005] da A. P. R. Silva, M. H. T. Martins, B. P. S. Rocha, A. A. F. Loureiro, L. B. Ruiz, and H. C. Wong. Decentralized intrusion detection in wireless sensor networks. In A. Boukerche and de R. B. Araujo, editors, *Q2SWinet'05 - Proceedings of the First ACM Workshop on Q2S and Security for Wireless and Mobile Networks, Montreal, Quebec, Canada, October 13, 2005*, pages 16–23, 2005. ACM. 89, 114

[De Dominicis et al. 2009] C. De Dominicis, P. Ferrari, A. Flammini, E. Sisinni, M. Bertocco, G. Giorgi, C. Narduzzi, and F. Tramarin. Investigating wirelesshart coexistence issues through a specifically designed simulator. In *Instrumentation and Measurement Technology Conference, 2009. I2MTC '09. IEEE*, pages 1085–1090, May 2009. 60

[Debar et al. 2007] H. Debar, Y. Thomas, F. Cuppens, and N. Cuppens-Boulahia. Enabling automated threat response through the use of a dynamic security policy. *Journal in Computer Virology*, 3(3):195–210, 2007. 92

[Deji et al. 2010] C. Deji, N. Mark, and M. Aloysius. *WirelessHART : Real-Time Mesh Network for Industrial Automation*. Springer US, 2010. 33, 42, 43, 48, 167

[Dine et al. 2016] A. V. Dine, M. Assante, and P. Stoutland. Outpacing Cyber Threats Priorities for Cybersecurity at Nuclear Facilities. pages 1–32, 2016. 17, 21, 24

[Dolev and Yao 1983] D. Dolev and A. C. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–207, 1983. 116

[Dong et al. 2011] D. Dong, X. Liao, Y. Liu, C. Shen, and X. Wang. Edge self-monitoring for wireless sensor networks. *Parallel and Distributed Systems, IEEE Transactions on*, 22(3):514–527, March 2011. 113, 114

[Douceur 2002] J. R. Douceur. The sybil attack. In P. Druschel, M. F. Kaashoek, and A. I. T. Rowstron, editors, *Peer-to-Peer Systems, First International Workshop, IPTPS 2002, Cambridge, USA*, volume 2429 of *Lecture Notes in Computer Science*, pages 251–260, 2002. Springer. 52, 65

[Dworkin 2004] SP 800-38C. recommendation for block cipher modes of operation: The CCM Mode for Authentication and Confidentiality. Technical report, Gaithersburg, MD, United States, 2004. 42, 43

[F-Secure Labs 2008] F-Secure Labs. Worm:W32/Downadup.A Description. 2008. [Online; accessed 13-March-2018]. 17

[Falliere et al. 2011] N. Falliere, L. O. Murchu, and E. Chien. W32. stuxnet dossier. *White paper, Symantec Corp., Security Response*, 5, 2011. 1, 18, 21

[Fovino et al. 2010] I. N. Fovino, A. Carcano, T. D. L. Murel, A. Trombetta, and M. Masera. Modbus/dnp3 state-based intrusion detection system. In *24th IEEE International Conference on Advanced Information Networking and Applications, AINA, Australia*, pages 729–736, 2010. IEEE Computer Society. 30, 88

[France 2013] M. d. l. D. France. French White Paper Defence and Nacional Security 2013. page 137, 2013. 11

[Goh et al. 2017] J. Goh, S. Adepu, M. Tan, and Z. S. Lee. Anomaly Detection in Cyber Physical Systems Using Recurrent Neural Networks. *2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)*, pages 140–145, 2017. 136

[Gollmann 2012] D. Gollmann. Veracity, plausibility, and reputation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7322 LNCS:20–28, 2012. 136

[GReAT 2016] GReAT. BlackEnergy APT Attacks in Ukraine employ spearphishing with Word documents. 2016. [Online; accessed 13-March-2018]. 19

[Guha and Khuller 1998] S. Guha and S. Khuller. Approximation algorithms for connected dominating sets. *Algorithmica*, 20(4):374–387, April 1998. 119

[Hallman et al. 2017] R. Hallman, J. Bryan, G. Palavicini, J. Divita, and J. Romero-Mariona. Ioddos ,Äî the internet of distributed denial of service attacks - a case study of the mirai malware and iot-based botnets. In *Proceedings of the 2nd International Conference on Internet of Things, Big Data and Security - Volume 1: IoTBDS,*, pages 47–58, 2017. INSTICC, ScitePress. 19, 20, 25

[Han et al. 2011] S. Han, X. Zhu, A. K. Mok, D. Chen, and M. Nixon. Reliable and real-time communication in industrial wireless mesh networks. In *17th IEEE RTAS, USA*, pages 3–12, 2011. IEEE Computer Society. 31

[HART Communication Foundation] HART Communication Foundation. WirelessHART. `http://www.hartcom.org`. 29, 31, 39, 41, 66, 81, 99, 128, 142, 144, 169

[Hilton 2016] S. Hilton. Dyn Analysis Summary Of Friday October 21 Attack. 2016. [Online; accessed 13-March-2018]. 19, 20

[Hochreiter and Schmidhuber 1997] S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, nov 1997. 137, 148

[Huang et al. 2009] Y.-L. Huang, A. Cárdenas, S. Amin, Z.-S. Lin, H.-Y. Tsai, and S. Sastry. Understanding the physical and economic consequences of attacks on control systems. *International Journal of Critical Infrastructure Protection*, 2(3):73 – 83, 2009. 1, 29, 112

[Huitsing et al. 2008] P. Huitsing, R. Chandia, M. Papa, and S. Shenoi. Attack taxonomies for the modbus protocols. *IJCIP*, 1:37–44, 2008. 30, 88

[ICS CERT 2016] ICS CERT. Alert (IR-ALERT-H-16-056-01): Cyber-Attack Against Ukrainian Critical Infrastructure. 2016. 1, 18, 19, 21

[IEEE 802.15.4-2006] IEEE 802.15.4-2006. IEEE 802.15.4-2006: Standard for Local and metropolitan area networks-Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs). `http://www.ieee.org`. 33, 42, 43, 62, 72, 135

[Igure et al. 2006] V. M. Igure, S. A. Laughter, and R. D. Williams. Security issues in SCADA networks. *Computers & Security*, 25(7):498–506, 2006. 10, 13, 15, 30, 141

[InetManet] InetManet. `https://github.com/aarizaq/inetmanet-2.0`. 62

[Joseph 2018] C. Joseph. New tool automatically finds and hacks vulnerable internet-connected devices - motherboard. 2018. [Online; accessed 13-March-2018]. 23

[Kalam et al. 2003] A. A. E. Kalam, R. E. Baida, P. Balbiani, S. Benferhat, F. Cuppens, Y. Deswarte, A. Miege, C. Saurel, and G. Trouessin. Organization based access control. In *Policies for Distributed Systems and Networks, 2003. Proceedings. POLICY 2003. IEEE 4th International Workshop on*, pages 120–131, June 2003. 95, 96

[Karlof and Wagner 2003] C. Karlof and D. Wagner. Secure routing in wireless sensor networks: attacks and countermeasures. *Ad Hoc Networks*, 1(2-3):293–315, 2003. 50, 51, 65

[Kaspersky Lab ICS CERT Threat 2016] Kaspersky Lab ICS CERT Threat. Industrial cybersecurity threat landscape. 2016. 14, 15

[Kaspersky Lab ICS CERT Threat 2017] Kaspersky Lab ICS CERT Threat. Threat Landscape for Industrial Automation Systems in H1 2017. 2017. 14

[Kaspersky Lab ICS CERT Threat 2018] Kaspersky Lab ICS CERT Threat. Threat Landscape for Industrial Automation Systems in H2 2017. 2018. 14, 15

[Keeney et al. 2005] M. Keeney, E. Kowalski, D. Cappelli, A. Moore, T. Shimeall, and S. Rogers. Insider Threat Study: Computer System Sabotage in Critical Infrastructure Sectors. *U.S. Secret Service and CERT Coordination Center/SEI*, (May):1–44, 2005. 22

[Khaitan and McCalley 2015] S. K. Khaitan and J. D. McCalley. Design techniques and applications of cyberphysical systems: A survey. *IEEE Systems Journal*, 9(2):350–365, 2015. 26

[Khalil et al. 2007] I. Khalil, S. Bagchi, and N. B. Shroff. Liteworp: Detection and isolation of the wormhole attack in static multihop wireless networks. *Computer Networks*, 51(13):3750 – 3772, 2007. 113

[Khan et al. 2016] R. Khan, P. Maynard, K. McLaughlin, D. Laverty, and S. Sezer. Threat analysis of blackenergy malware for synchrophasor based real-time control and monitoring in smart grid. In *Proceedings of the 4th International Symposium for ICS & SCADA Cyber Security Research 2016*, ICS-CSR '16, pages 1–11, Swindon, UK, 2016. BCS Learning & Development Ltd. 19

[Kieseberg and Weippl 2018] P. Kieseberg and E. Weippl. Security challenges in cyber-physical production systems. In *Lecture Notes in Business Information Processing*, volume 302, pages 3–16. 2018. 26

[Kim et al. 2008] A. N. Kim, F. Hekland, S. Petersen, and P. Doyle. When HART goes wireless: Understanding and implementing the wirelesshart standard. In *Proceedings of 13th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA, Hamburg, Germany*, pages 899–907, 2008. IEEE. 31

[Krotofil et al. 2015] M. Krotofil, J. Larsen, and D. Gollmann. The Process Matters : Ensuring Data Veracity in Cyber-Physical Systems. *ACM Symposium on Information, Computer and Communications Security*, pages 133–144, 2015. 136

[Langner 2011] R. Langner. Stuxnet: Dissecting a cyberwarfare weapon. *IEEE Security Privacy*, 9(3):49–51, May 2011. 18, 26

[Larkin et al. 2014] R. D. Larkin, J. L. Jr., J. W. Butts, and M. R. Grimaila. Evaluation of security solutions in the SCADA environment. *DATA BASE*, 45(1):38–53, 2014. 30

[Lee et al. 2014] J. Lee, B. Bagheri, and H.-A. Kao. Recent Advances and Trends of Cyber-Physical Systems and Big Data Analytics in Industrial Informatics. *Int. Conference on Industrial Informatics (INDIN) 2014*, (November 2015), 2014. 10

[Lee et al. 2010] J. Lee, K. Kapitanova, and S. H. Son. The price of security in wireless sensor networks. *Computer Networks*, 54(17):2967–2978, 2010. 93

[Lee et al. 2016] SANS ICS. Defense Use Case-Analysis of the Cyber Attack on the Ukrainian Power Grid. Technical report, United States, 2016. 18, 24

[Lemay et al. 2018] A. Lemay, J. Calvet, F. Menet, and J. M. Fernandez. Survey of publicly available reports on advanced persistent threat actors. *Computers & Security*, 72:26–59, 2018. 24

[Lemay and Fernandez 2013] A. Lemay and J. M. K. S. Fernandez. Defending the SCADA network controlling the electrical grid from advanced persistent threats. 3582693:187, 2013. 5, 10, 141

[Mansfield-Devine 2016] S. Mansfield-Devine. Ransomware: taking businesses hostage. *Network Security*, 2016(10):8–17, oct 2016. 24

[Marsh et al. 2009] D. W. Marsh, R. O. Baldwin, B. E. Mullins, R. F. Mills, and M. R. Grimaila. A security policy language for wireless sensor networks. *Journal of Systems and Software*, 82(1):101 – 111, 2009. Special Issue: Software Performance - Modeling and Analysis. 94

[Marsh 1997] R. Marsh. Critical foundations: Protecting America's infrastructure. *Commission on Critical Infrastructure Protection*, page 192, 1997. 12

[Maw et al. 2012] H. A. Maw, H. Xiao, and B. Christianson. An adaptive access control model with privileges overriding and behaviour monitoring in wireless sensor networks. In *Proceedings of the 8H ACM Symposium on QoS and Security for Wireless and Mobile Networks*, Q2SWinet '12, pages 81–84, New York, NY, USA, 2012. ACM. 95

[Maw et al. 2014a] H. A. Maw, H. Xiao, B. Christianson, and J. A. Malcolm. An evaluation of break-the-glass access control model for medical data in wireless sensor networks. In *2014 IEEE 16th International Conference on e-Health Networking, Applications and Services (Healthcom)*, pages 130–135, Oct 2014. 95

[Maw et al. 2014b] H. A. Maw, H. Xiao, B. Christianson, and J. A. Malcolm. A survey of access control models in wireless sensor networks. *Journal of Sensor and Actuator Networks*, 3(2):150–180, 2014. 94, 95

[McLaughlin and McDaniel 2012] S. McLaughlin and P. McDaniel. Sabot: Specification-based payload generation for programmable logic controllers. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, CCS '12, pages 439–449, New York, NY, USA, 2012. ACM. 26

[Metasploit 2018] Metasploit. Penetration testing software, pen testing security. 2018. [Online; accessed 13-March-2018]. 23

[Microsoft 2008] Microsoft. Microsoft Security Bulletin MS08-067 - Critical. 2008. [Online; accessed 13-March-2018]. 17

[Miller and Rowe 2012] B. Miller and D. Rowe. A survey scada of and critical infrastructure incidents. In *Proceedings of the 1st Annual Conference on Research in Information Technology*, RIIT '12, pages 51–56, New York, NY, USA, 2012. ACM. 16

[Mitchell and Chen 2013] R. Mitchell and I. Chen. A survey of intrusion detection techniques for cyber-physical systems. *ACM Comput. Surv.*, 46(4):55:1–55:29, 2013. 56, 88, 90, 146

[Mitchell and Chen 2014] R. Mitchell and I. Chen. A survey of intrusion detection in wireless network applications. *Computer Communications*, 42:1–23, 2014. 56, 88, 90, 112, 114, 146

[Mo et al. 2015] Y. Mo, S. Weerakkody, and B. Sinopoli. Physical Authentication of Control Systems. *IEEE Control Systems Magazine*, 35(1):93–109, 2015. 136

[Moore et al. 2003] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver. Inside the slammer worm. *IEEE Security & Privacy Magazine*, 1(4):33–39, jul 2003. 24

[Moyne and Tilbury 2007] J. R. Moyne and D. Tilbury. The emergence of industrial control networks for manufacturing control, diagnostics, and safety data. *Proceedings of the IEEE*, 95(1):29–47, Jan 2007. 111

[Nai Fovino et al. 2009] I. Nai Fovino, A. Carcano, M. Masera, and A. Trombetta. An experimental investigation of malware attacks on SCADA systems. *International Journal of Critical Infrastructure Protection*, 2(4):139–145, dec 2009. 24

[National Communications System 2004] National Communications System. Supervisory Control and Data Acquisition (SCADA) Systems. *Technical Information Bulletin 04-1*, (October):76, 2004. 6, 7, 8, 140

[Nazir et al. 2017] S. Nazir, S. Patel, and D. Patel. Assessing and augmenting SCADA cyber security: A survey of techniques. *Computers & Security*, 70:436–454, 2017. 26

[Neggazi et al. 2014] B. Neggazi, M. Haddad, V. Turau, and H. Kheddouci. A self-stabilizing algorithm for edge monitoring problem. In P. Felber and V. K. Garg, editors, *Stabilization, Safety, and Security of Distributed Systems - 16th International Symposium, SSS 2014, Paderborn, Germany, September 28 - October 1, 2014. Proceedings*, volume 8756 of *Lecture Notes in Computer Science*, pages 93–105, 2014. Springer. 113

[Newsome et al. 2004] J. Newsome, E. Shi, D. X. Song, and A. Perrig. The sybil attack in sensor networks: analysis & defenses. In K. Ramchandran, J. Sztipanovits, J. C. Hou, and T. N. Pappas, editors, *Proceedings of the Third International Symposium on Information Processing in Sensor Networks, IPSN, USA*, pages 259–268, 2004. ACM. 65

[Nicholson et al. 2012] A. Nicholson, S. Webber, S. Dyer, T. Patel, and H. Janicke. SCADA security in the light of cyber-warfare. *Computers & Security*, 31(4):418–436, 2012. 21, 23

[NIST 2011] National Institute of Standards & Technology. SP - 800-39. Managing Information Security Risk: Organization, Mission, and Information System View. Technical report, Gaithersburg, MD, United States, 2011. 24

[Nixon et al. 2016] A. Nixon, J. Costello, and Z. Wikholm. An After-Action Analysis of the Mirai Botnet Attacks on Dyn. 2016. [Online; accessed 13-March-2018]. 19, 20

[Nobre et al. 2010] M. Nobre, I. Silva, L. Guedes, and P. Portugal. Towards a wirelesshart module for the ns-3 simulator. In *Emerging Technologies and Factory Automation (ETFA), 2010 IEEE Conference on*, pages 1–4, Sept 2010. 60

[OMNeT++] OMNeT++. `http://www.omnetpp.org/`. 61, 62

[Onat and Miri 2005] I. Onat and A. Miri. An intrusion detection system for wireless sensor networks. In *Wireless And Mobile Computing, Networking And Communications, 2005. (WiMob'2005), IEEE International Conference on*, volume 3, pages 253–259 Vol. 3, Aug 2005. 112

[OVH 2016] OVH. The DDoS that didnt break the camels VAC. 2016. [Online; accessed 13-March-2018]. 19

[Petersen and Carlsen 2011] S. Petersen and S. Carlsen. Wirelesshart versus isa100.11a: The format war hits the factory floor. *Industrial Electronics Magazine, IEEE*, 5(4):23–34, Dec 2011. 31

[Pietre-Cambacedes et al. 2011] L. Pietre-Cambacedes, M. Tritschler, and G. N. Ericsson. Cybersecurity myths on power control systems: 21 misconceptions and false beliefs. *IEEE Transactions on Power Delivery*, 26(1):161–172, Jan 2011. 5, 13, 141

[Protais 2016] M. Protais. L'hydrolienne de Sabella attaquée par des pirates. *L'Usine Nouvelle*, mar 2016. [Online; accessed 13-March-2018]. 25

[Raza et al. 2009] S. Raza, A. Slabbert, T. Voigt, and K. Landernäs. Security considerations for the wirelesshart protocol. In *Proceedings of 12th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA, Spain*, pages 1–8, 2009. IEEE. 31, 59

[Rinaldi et al. 2001] S. M. Rinaldi, J. P. Peerenboom, and T. K. Kelly. Identifying, understanding, and analyzing critical infrastructure interdependencies. *IEEE Control Systems*, 21(6):11–25, Dec 2001. 12, 13, 167

[Rocchetto and Tippenhauer 2016] M. Rocchetto and N. O. Tippenhauer. On attacker models and profiles for cyber-physical systems. In I. G. Askoxylakis, S. Ioannidis, S. K. Katsikas, and C. A. Meadows, editors, *Computer Security - ESORICS 2016 - 21st European Symposium on Research in Computer Security, Heraklion,*

Greece, September 26-30, 2016, Proceedings, Part II, volume 9879 of *Lecture Notes in Computer Science*, pages 427–449, 2016. Springer. 21

[Roman et al. 2006] R. Roman, J. Zhou, and J. Lopez. Applying intrusion detection systems to wireless sensor networks. In *Consumer Communications and Networking Conference, 2006. CCNC 2006. 3rd IEEE*, volume 1, pages 640–644, Jan 2006. 113

[Roosta et al. 2008] T. Roosta, D. K. Nilsson, U. Lindqvist, and A. Valdes. An intrusion detection system for wireless process control systems. In *IEEE 5th International Conference on Mobile Adhoc and Sensor Systems, MASS,USA*, pages 866–872, 2008. IEEE. 56, 60, 89, 114

[Roosta et al. 2006] T. Roosta, S. Shieh, and S. Sastry. taxonomy of security attacks in sensor networks and countermeasures. In *In The First IEEE International Conference on System Integration and Reliability Improvements. Hanoi*, pages 13–15, 2006. 43

[Rubio-Hernán et al. 2016] J. Rubio-Hernán, L. De Cicco, and J. García-Alfaro. Revisiting a Watermark-based Detection Scheme to Handle Cyber-Physical Attacks. *Proceedings - 2016 11th International Conference on Availability, Reliability and Security, ARES 2016*, pages 21–28, 2016. 136

[Sagala et al. 2017] A. Sagala, R. Pardosi, A. Lumbantobing, and P. Siagian. Industrial control system security-malware botnet detection. *Proceeding - 2016 International Conference on Computer, Control, Informatics and its Applications: Recent Progress in Computer, Control, and Informatics for Data Science, IC3INA 2016*, pages 125–130, 2017. 25

[Schneier 2016] B. Schneier. Lessons From the Dyn DDoS Attack. 2016. [Online; accessed 13-March-2018]. 19, 26

[SecurityWeek.Com 2016] SecurityWeek.Com. Disgruntled Gamer 'Likely' Behind October US Hacking: Expert. 2016. [Online; accessed 13-March-2018]. 20

[Shin et al. 2010] S. Shin, T. Kwon, G. Y. Jo, Y. Park, and H. Rhy. An experimental study of hierarchical intrusion detection for wireless industrial sensor networks. *IEEE Transactions on Industrial Informatics*, 6(4):744–757, Nov 2010. 89

[Shodan 2018] Shodan. The search engine for internet-connected devices. 2018. [Online; accessed 13-March-2018]. 23

[Slay and Miller 2007] J. Slay and M. Miller. Lessons learned from the maroochy water breach. In E. Goetz and S. Shenoi, editors, *Critical Infrastructure Protection, Post-Proceedings of the First Annual IFIP Working Group 11.10 International Conference on Critical Infrastructure Protection, USA*, volume 253 of *IFIP*, pages 73–82, 2007. Springer. 1, 17

[Song et al. 2008] J. Song, S. Han, A. Mok, D. Chen, M. Lucas, and M. Nixon. Wirelesshart: Applying wireless technology in real-time industrial process control. In *Real-Time and Embedded Technology and Applications Symposium, 2008. RTAS '08. IEEE*, pages 377–386, April 2008. 31

[Stouffer et al. 2015] National Institute of Standards & Technology. SP - 800-82 Rev 2. Guide to Industrial Control Systems (ICS) Security. Technical report, Gaithersburg, MD, United States, 2015. 6, 7, 10, 13, 16, 140, 141

[Stouffer et al. 2011] National Institute of Standards & Technology. Sp 800-82. guide to industrial control systems (ics) security. Technical report, Gaithersburg, MD, United States, 2011. 30

[Tabrizi and Pattabiraman 2014] F. M. Tabrizi and K. Pattabiraman. A model-based intrusion detection system for smart meters. In *15th International IEEE Symposium on High-Assurance Systems Engineering, HASE 2014, Miami Beach, FL, USA, January 9-11, 2014*, pages 17–24, 2014. IEEE Computer Society. 30

[Tarakanov 2010] D. Tarakanov. Black DDoS. 2010. [Online; accessed 13-March-2018]. 19

[The Council of the European Union 2008] The Council of the European Union. Council Directive 2008/114/EC of 8 December 2008 on the indentification and designation of European critical infrastructures and the assessment of the need to improve their protection. *Official Journal of the European Union*, pages 75–82, 2008. 12

[The National Institute of Standards and Technology 2013] The National Institute of Standards and Technology. Improving Critical Infrastructure Cybersecurity Executive Order 13636: Preliminary Cybersecurity Framework. 2013. 12

[The Telegraph 2009] The Telegraph. French fighter planes grounded by computer virus. `http://www.telegraph.co.uk/news/worldnews/europe/france/4547649/French-fighter-planes-grounded-by-computer-virus.html`, February 07,2009. 17

[Ussath et al. 2016] M. Ussath, D. Jaeger, F. Cheng, and C. Meinel. Advanced persistent threats: Behind the scenes. In *2016 Annual Conference on Information Science and Systems (CISS)*, pages 181–186, March 2016. 24

[Waidner and Kasper 2016] M. Waidner and M. Kasper. Security in industrie 4.0 - challenges and solutions for the fourth industrial revolution. *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1303–1308, 2016. 9

[Wang et al. 2006] Y. Wang, G. Attebury, and B. Ramamurthy. A survey of security issues in wireless sensor networks. *IEEE Communications Surveys and Tutorials*, 8(1-4):2–23, 2006. 43, 50, 51, 93

[Wireless System for Automation] Wireless System for Automation. ISA100. `http://www.isa.org`. 29, 135, 148

[Xiangyu et al. 2017] L. Xiangyu, L. Qiuyang, and S. Chandel. Social Engineering and Insider Threats. In *2017 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, pages 25–34, oct 2017. IEEE. 25

[Yu et al. 2013] J. Yu, N. Wang, G. Wang, and D. Yu. Connected dominating sets in wireless ad hoc and sensor networks - A comprehensive survey. *Computer Communications*, 36(2):121–134, 2013. 118, 119

[Zand et al. 2014] P. Zand, E. Mathews, P. Havinga, S. Stojanovski, E. Sisinni, and P. Ferrari. Implementation of wirelesshart in the ns-2 simulator and validation of its correctness. *Sensors*, 14(5):8633–8668, 2014. 60

[Zhu et al. 2011] B. Zhu, A. D. Joseph, and S. Sastry. A taxonomy of cyber attacks on SCADA systems. In *2011 IEEE International Conference on Internet of Things (iThings) & 4th IEEE International Conference on Cyber, Physical and Social Computing (CPSCom), Dalian, China, October 19-22, 2011*, pages 380–388, 2011. IEEE. 10, 140

[ZigBee Alliance] ZigBee Alliance. ZigBee Pro. `http://www.zigbee.org`. 29, 135, 148

List of Figures

List of Tables

UNIVERSITE BRETAGNE LOIRE / MATHSTIC

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Titre : Évaluation et mise en œuvre de la sécurité dans les systèmes SCADA à base de réseaux de capteurs sans fil

Mots clés : Systèmes industriels, Réseaux de capteurs sans fil, WirelessHART, Analyse de protocoles, Détection d'intrusions

Résumé : La sécurité des systèmes de contrôle industriel est une préoccupation majeure. En effet, ces systèmes gèrent des installations qui jouent un rôle économique important. En outre, attaquer ces systèmes peut non seulement entraîner des pertes économiques, mais aussi menacer des vies humaines.

Par conséquent, et comme ces systèmes dépendent des données collectées, il devient évident qu'en plus des exigences de temps réel, il est important de sécuriser les canaux de communication entre ces capteurs et les contrôleurs principaux. Ces problèmes sont plus difficiles à résoudre dans les réseaux de capteurs sans fil (WSN).
Cette thèse a pour but d'aborder les questions de sécurité des WSN. Tout d'abord, nous effectuons une étude de sécurité approfondie du protocole WirelessHART. Ce dernier est le protocole leader pour les réseaux de capteurs sans fil industriels (WISN).

Nous évaluons ses forces et soulignons ses faiblesses et ses limites. En particulier, nous décrivons deux vulnérabilités de sécurité dangereuses dans son schéma de communication et proposons des améliorations afin d'y remédier.

Ensuite, nous présentons wIDS, un système de détection d'intrusion (IDS) multicouches qui se base sur les spécifications, spécialement développé pour les réseaux de capteurs sans fil industriels. L'IDS proposé vérifie la conformité de chaque action effectuée par un nœud sans fil sur la base d'un modèle formel du comportement normal attendu.

Title : **Assessment and enforcement of Wireless Sensor Networks-based SCADA systems security**

Keywords : Industrial Control Systems, Wireless Sensor Networks, WirelessHART, Protocol analysis, Intrusion Detection

Abstract : The security in Industrial Control Systems is a major concern. Indeed, these systems manage installations that play an important economical role. Furthermore, targeting these systems can lead not only to economical losses but can also threaten human lives. Therefore, and as these systems depend on sensing data, it becomes obvious that additionally to real-time requirement, it is important to secure communication channels between these sensors and the main controllers. These issues are more challenging in Wireless Sensor Networks (WSN) as the use of wireless communications brings its own
security weaknesses.

This thesis aims to address WSN-based security issues. Firstly, we conduct an in-deep security study of the WirelessHART protocol. This latter is the leading protocol for Wireless Industrial Sensor Networks (WISN) and is the first international approved standard.

We assess its strengths and emphasize its weaknesses and limitations. In particular, we describe two harmful security vulnerabilities in the communication scheme of WirelessHART and propose improvement in order to mitigate them.

Secondly, we present wIDS, a multilayer specification-based Intrusion Detection System (IDS) specially tailored for Wireless Industrial Sensor Networks.

The proposed IDS checks the compliance of each action performed by a wireless node based on a formal model of the expected normal behavior.