Formal Verification of the WirelessHART Protocol - Verifying Old and Finding New Attacks

Martin Gunnarsson
martin.gunnarsson@ri.se
RISE Research Institutes of Sweden AB
Lund, Sweden

ABSTRACT

Connected industrial control systems open up many possibilities for increased efficiency and control but also bring drawbacks. Attacks against connected industrial control systems have highlighted the need for rigorous security analysis of every part of the systems.

In this paper, we report a careful scrutiny of the security of the WirelessHART protocol. A literature study points at the need for a formal verification of the protocol. We have modeled WirelessHART using the state-of-the-art formal verification tool Tamarin and tested the claimed security properties. Our analysis has verified the existence of all previously published attacks. In addition we have shown a new type of attack that enable an insider attacker to maliciously re-key devices in the system, effectively performing a denial of service attack on large parts of the system. Our analysis has demonstrated that WirelessHART is secure as long as no device in the network has been compromised. If an attacker can get a foothold in the network it can launch several types of attacks to compromise the network further.

KEYWORDS

WirelessHART, Tamarin, Formal Protocol Verification

1 INTRODUCTION

Industrial Control Systems (ICS) are becoming increasingly dependent on networking technologies and the interconnection of the system's parts to operate efficiently. Cheaper and more powerful networked devices enable data collection from industrial processes in ways that have not been previously possible. ICS-specific protocols have been developed to communicate with these new types of devices, one of them being WirelessHART [20]. Its precursor, Highway Addressable Remote Transducer Protocol (HART) [17], was developed in the 1980s for process automation. The HART protocol is wired and operates in point-to-point and multi-drop modes. The HART protocol can transmit both analog and digital signals in an ICS. In 2007, WirelessHART was standardized as IEC 62591. WirelessHART traffic is transmitted over IEEE 802.15.4 radio [8], like ordinary WiFi. Concepts like Smart Manufacturing and Industry 4.0 [28, 36] emphasize the inclusion of connected devices, sometimes called the Internet of Things (IoT), into manufacturing systems. Because WirelessHART interfaces with legacy HART technologies, it is a protocol suitable for wireless sensors connected to legacy ICS.

Security in ICS has become a pressing concern after a series of published attacks such as STUXNET [22], Black Energy [24], and Triton [12]. Studies have been published on attacks against cyber-physical systems and ICS [19, 37] attacks against oil and gas infrastructures have been studied [38]. Future challenges for ICS security have been identified [16] and in the realm of secure communications for ICS [33].

WirelessHART has been integrated into oil refineries [31] and ICS equipment produced by major manufacturers [23, 39]. When new technologies are integrated into ICS, they become new attack surfaces for malicious actors wanting access to the ICS. WirelessHART must be analyzed to verify the stated security goals of the protocol.

This paper will present an extensive *formal verification* of the WirelessHART security protocol. The WirelessHART standard, IEC 62591, is not openly available, making studies and security analyses of the standard more complex.

The security properties of WirelessHART have been studied since 2009 [35], and several works have been published [1, 5]. Published attacks include Sybil-Attacks, denial-of-service attacks, and device impersonation attacks. The security of the physical and medium access control layer of WirelessHART has been studied. A jamming attack was published in 2021 [9]. In 2021 the first attempt at a formal analysis of WirelessHART was published [25]. The authors modeled the end-to-end security protocol of WirelessHART and claimed to have found an attack. But no paper has managed to do a complete analysis of the entire WirelessHART protocol and all its details.

Previous security analyses of WirelessHART mainly used *semi-formal analysis*. The shortcoming of this approach is that it only covers some eventualities and possible executions. However, formal protocol verification techniques using *automated* proofs and the state-of-the-art prover Tamarin [27] can efficiently search for possible attacks over possible executions. Tamarin and other formal verification tools have demonstrated their utility by verifying known attacks on protocols and discovering new attacks. Analysis using Tamarin has been done on 5G-AKA [10], TLS1.3 [11], and recently the EMV standard [2], to only name a few.

The contributions of this paper are the following:

- We have cataloged and summarized all published security analyses of WirelessHART and attacks.
- We provide a detailed and comprehensive Tamarin model of the WirelessHART protocol. Our model covers the end-to-end security protocol, join sequence, network key change operation, and network advertisements. Our model allows us to find and verify all attacks already shown in previous works.
- During our analysis, we have found a novel, Malicious Rekeying attack. An insider attacker can send a malicious Network Key Change message impersonating the Security Manager. The Field Devices receiving these messages will change

their keys, and the legitimate Security Manager and Gateway cannot contact the Field Devices.

The rest of the paper will start with a description of WirelessHART in section 2. Next, we will outline previous work on WirelessHART security in Section 3. In section 4, we state the system model, assumptions, and threat model used in our analysis. We will then describe the Tamarin model used in our analysis and the methodology behind it in Section 5. We present our findings in Sections 6. Finally, we present related work in Section 7 and conclude.

2 THE WIRELESSHART PROTOCOL

In this section, we will present an overview of the WirelessHART specification. We present a shortened version focused on parts relevant to our security analysis. If the reader wishes to get all details of WirelessHART, we refer them to the WirelessHART specification[20]. The WirelessHART specification details the actors and procedures needed to implement the protocol. We obtained the WirelessHART standard from the International Electrotechnical Commission (IEC)[1].

In the next section, we use the following notation: data items such as keys or identities have been written with typewriter-font, e.g., `Join_key`. We write actors in italics, e.g., *Network Manager*. We denote procedures and states defined in the WirelessHART specification with quotation marks, e.g. "Join Process".

2.1 Actors and roles

WirelessHART is intended to facilitate communication with *Field Devices* in a factory environment. The *Plant Automation Host* will send *Commands* to the *Field Devices*. The commands can be requests to send a sensor value, move an actuator, or request information from the *Field Device*. The *Command* will be translated to WirelessHART in the *Gateway* and forwarded over WirelessHART.

An example of a small WirelessHART network can be seen in Figure 1. The figure shows all types of devices in a WirelessHART network. We will describe the types of devices below:

Field Devices join the network existing of a *Gateway* and a *Network manager*. The *Field Devices* have a physical port, called the *Maintenance Port*, where a *Handheld Devices* can be connected for management and diagnostics. It is used to start the "Join Process", described later.

At least one *Gateway* connects the WirelessHART network to the rest of the factory network. WirelessHART terminates at the *Gateway*, and messages are translated and forwarded to the *Plant Automation Network*. We have limited the scope of our analysis to include the *Gateway* and left the *Plant Automation Network* out of this work since the WirelessHART standard does not cover it.

Every network must have one *Network Manager* that manages the *Field Devices* by assigning transmission network information, keys, and more required by the protocol.

Each WirelessHART network also needs a *Security Manager*. The *Security Manager* is responsible for the keys in the system. One *Security Manager* can manage more than one WirelessHART network, but each network has one, *Security Manager*. The *Security Manager* and the *Network Manager* can be co-hosted on the same machine. We illustrated such a scenario in Figure 1.

[1]https://webstore.iec.ch/publication/24433

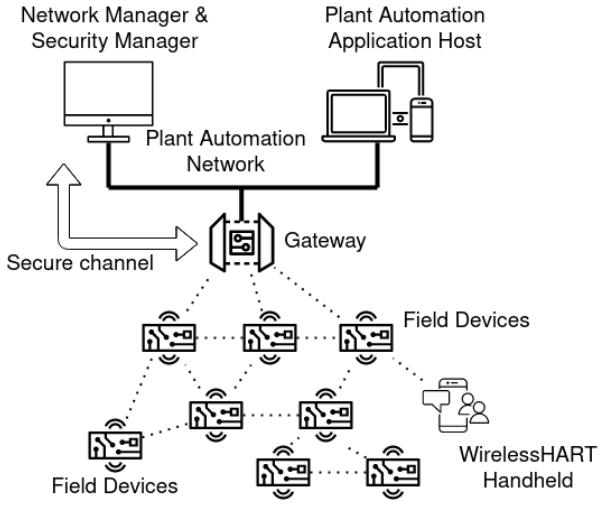

Figure 1: WirelessHART Automation Network.

2.2 WirelessHART protocol architecture

WirelessHART defines both network architecture and the radio protocols used for communication. In addition, the specification includes several sub-protocols that form layers in a protocol stack. The WirelessHART protocol uses an altered ISO/OSI 7-layer model[15] to separate the features and functions of the protocol into layers. This section will give a short overview of this.

The physical radio protocol used is IEEE 802.15.4-2006 [30]. Then the Physical Layer that has Medium Access Control (MAC)[2]. WirelessHART uses a mesh topology, with multiple routes for redundancy, for the *Field Devices* as indicated by the dotted lines in Figure 1.

The protocol abstraction layers are the physical Layer, the data-link Layer, the network layer, the transport layer, and the application layer. WirelessHART provides end-to-end encryption and single-hop integrity protection of messages. The security measures are handled at the network and data-link layers, and we will only focus on the relevant parts of the protocol stack. We will not go into further detail about the transport and application layers.

2.3 WirelessHART security

WirelessHART specifies several mechanisms to secure the network: single-hop message integrity protection, end-to-end encryption of data, and the management of keys.

The WirelessHART specification does not specify what adversary model the document's authors intended, and the security requirements of the protocol are not outright stated.

The specification mentions that the single-hop MIC layer is intended to protect against attackers that do not know any keys used in the network. The end-to-end transmission security protocol protects against attackers that are in the network.

[2]Usually, MAC is the acronym for Message Authentication Code. In the WirelessHART specification, MAC stands for Medium Access Control. To avoid confusion, we use Message Integrity Code (MIC) in this work.

To aid in reasoning about the different attacker models, we will introduce the notion of *Insider Attacker* and *Outsider Attacker* in Section 4. We will give a detailed description of what the different attackers know later, and until now, work with the assumption that *Insider Attacker* knows systems secrets and that *Outsider Attacker* does not know any secrets used in the network.

The WirelesHART specification states that *Field Devices* shall only be connected to the network after they receive the `Join_Key`. A MIC ensures hop-by-hop integrity. The key is the `Network_key` and protects against an *Outsider Attacker*. End-to-end confidentiality is achieved by encrypting messages. Here protection against *Insider Attackers* is managed. *Field Devices* use shared symmetric keys to secure communication with each other and provide authentication.

The WirelessHART standard also reserves itself against attacks on cryptographic primitives and the implementation of cryptographic functions.

2.4 Keys and Key Distribution

WirelessHART only uses symmetric key cryptography. The *Security Manager* is responsible for provisioning keys to all *Field Devices* in the network. When a new device joins the network, the `Join_key` is used. The Join_key *can* be different for each device according to the standard. The `Network_Key` is shared with all devices. It is used to compute and verify the DLPDU MIC. WirelessHART sends data through sessions between a *Field Device* and either the *Network Manager* or the *Gateway*. The session can be either unicast or broadcast. A `Unicast_Session_Key` is unique to a *Field Device*. `Broadcast_Session_Key`, are shared with all *Field Devices*. Each *Field Device* has a unicast session with the *Network Manager*. All *Field Devices* also share a broadcast channel with the *Network Manager*.

2.5 Error detection and the End-to-End security Protocol

Two parts of the protocol stack are used to ensure end-to-end message confidentiality and tampering resistance on messages.

The cryptographic primitives in WirelessHART is an Authenticated Encryption with Associated Data (AEAD), specifically AES-CCM*[30].

The Data-Link layer prevents the *Outsider Attacker* from tampering with messages as the Network_Key is used to compute a MIC over the contents of the DLPDU. The MIC is computed using AES-CCM* with the contents of the DLPDU as associated data and an empty plaintext field. The Network_Key is used to compute the DLPDU MIC. Each device that forwards the message can verify the MIC, preventing tampering with the message.

In Table 1, we show a DLPDU and the fields. The payload and all fields except the CRC are authenticated, and integrity protected using the `Network_key`. The nonce used for calculating the MIC included in the DLPDU is formed by concatenating the ASN, a network-wide counter value, with the source address.

Table 1: A WirelessHART Data Link Layer Protocol Data Unit (DLPDU)

Data Mode	Address Specifier	Sequence Number	Network_ID	Destination Address	Source Address	DLPDU Specifier	DLL Payload	MIC	CRC

Messages are encrypted to prevent information disclosure to either an *Outsider Attacker* or an *Insider Attacker*. Encryption is done at the Network Layer by encrypting the NPDU contents. The same AES-CCM* algorithm provides both encryption and a MIC. In Table 2, we show the fields in an NPDU. The NPDU Payload is encrypted and authenticated, and the rest of the NPDU is authenticated except the Hop-to-live, Counter, and MIC fields.

Table 2: A WirelessHART Network Layer Protocol Data Unit (NPDU)

NL Control	Hop-to-live	Sequence Number	Graph_ID	Destination Address	Source Address	(Proxy route)	0-N (source routes)	Security Control	Counter	MIC	Payload

A `Session_Key` or `Join_Key` is used to encrypt the NPDU. The nonce for the NPDU MIC is constructed by concatenating a counter with the source device's ID. Encrypted NPDU sent between two parties is a session and provides confidentiality, integrity protection, source authentication (provided it is a unicast session), and replay protection for the transmitted data. Replay protection is achieved by using incremental sequence numbers. A device will not accept messages with a lower sequence number than previously received and decrypted.

Two types of sessions exist, unicast and broadcast sessions. Sessions terminating in the *Network Manager* are used for managing the network and *Field Devices*. The sessions terminating in the *Gateway* are used for actual process data to and from the *Gateway*. During the "Join Sequence", a *Field Device* is provisioned with one unicast session and one broadcast session with both the *Gateway* and *Network Manager*, for a total of four sessions.

2.6 WirelessHART Join Sequence

The WirelessHART "Join Sequences" defines the messages transmitted and the steps taken when a new *Field Device* is connected to a WirelessHART network. The protocol can be seen in Figure 2. The process is described step by step below:

(0) Before enrolling a device the *Security Manager* generate a `Join_Key` for the new device. The `Join_Key` is transferred to the hand-held Maintenance Tool.
(1) The *Field Device* being connected is connected to a hand-held Maintenance Tool with a cable to the Maintenance Interface that is a physical port on the *Field Device*. The Maintenance Tool can then transfer a `Join_key` and a `Network_name` to the *Field Device*. The `Network_Name` allows the *Field Device* to start scanning for a network to join.
(2) When a network advertisement with the correct name has been received, a Join Request is sent to the *Network Manager*. The Join Request is forwarded over the mesh network to the *Network Manager*. The Join Request comprises the joining *Field Device's* identity and information about the neighboring *Field Devices*. All this is encrypted with the joining *Field Device's* `Join_Key`. The `Well-Known-Key` is used to compute the DLPDU MIC. The *Network Manager* decrypts and verifies the message using the `Join_key`.
The *Network Manager* responds by sending a *Field Device* Nickname, the `Network_Key` and one or more Session keys, generated by the *Network Manager*, encrypted with the `Join_Key`.

The Joining *Field Device* will acknowledge this message using the received *Network Manager* unicast session key and the `Network_Key`.

(3) The *Network Manager* now sends network information to the joining *Field Device*, such as time sources and routes, and links.
(4) The *Field Device* is left in a "quarantined" state. It can communicate with the rest of the *Field Devices* and the *Network Manager* but does not have a session to communicate with the *Gateway*.
(5) The *Network Manager* can leave the joining *Field Device* in "quarantine" for some time or directly provision a *Gateway* session to the joining *Field Device*. Once the joining *Field Device* has a session with the *Gateway*, it can communicate with the process network. The joining *Field Device* is now a part of the network and considered operational.

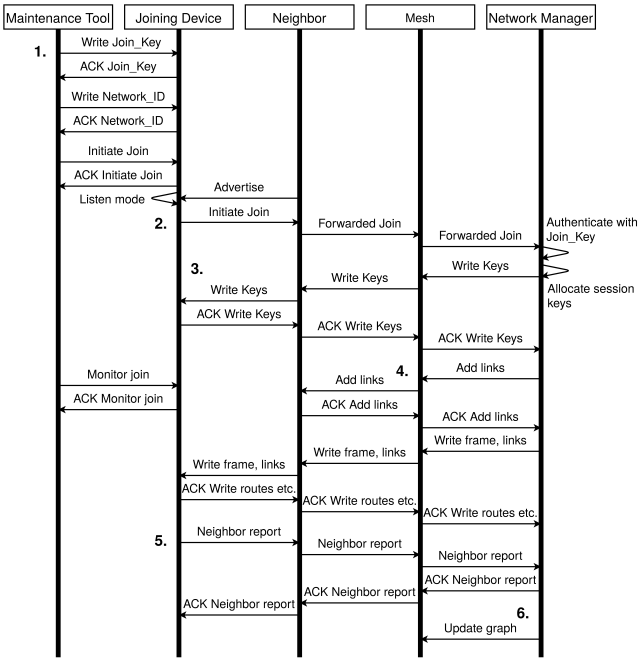

Figure 2: WirelessHART Join Sequence.

2.7 Network Key Change Operation

WirelessHART has a special sub-protocol defined for changing keys for nodes in the network. All keys used by WirelessHART can be changed with this protocol, including `Network_Key`, `Broadcast_Session_Key`, and `Unicast_Session_Key`. The protocol is executed between a *Field Device* and the *Network Manager*, with neighbors and mesh devices forwarding messages from the source to the destination. We show a diagram of the protocol in Figure 3.

The steps taken to change a key are:

(1) The *Network Manager* transmits a Broadcast message that is forwarded to all *Field Devices*. The *Network Manager* Broadcast key protects the Command.
(2) Each *Field Device* sends a response, and the response is encrypted with the *Network Manager* Unicast key preventing spoofing attacks.
(3) The *Network Manager* can verify that all *Field Devices* have acknowledged the new key. All *Field Devices* will change to the new key at a later time slot.

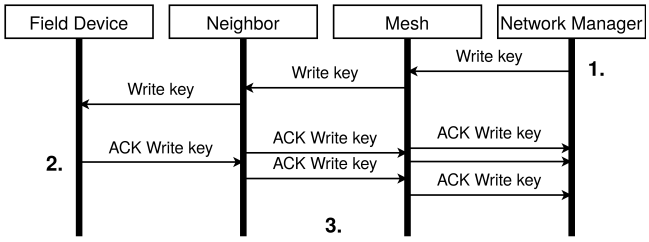

Figure 3: Network Key change operation.

The specification allows a key to be sent out before an anticipated change of keys, and sending the keys in advance allows retransmission if the message is lost. Since each *Field Device* acknowledges the received key, the *Network Manager* can verify that the key was received.

3 PREVIOUS SECURITY ANALYSIS OF WIRELESSHART

In [34, 35], the authors note the lack of security requirements in the WirelessHART specification and claim that they have done the first security analysis of the protocol. Furthermore, the authors argue that the following attacks need more attention: message corruption attacks, jamming attacks, and traffic analysis. Denial of Service attacks, De-synchronization attacks, and Wormhole attacks unless source routing is used. The authors claim that tampering with messages can be done if an adversary knows the `Network_Key`. Therefore, the authors recommend regular changing of the `Network_Key`. Further, an adversary can spoof advertisement messages with the `Well-Known-Key`.

In [3], Bayou et al. present a Disconnect Sybil attack against WirelessHART. In the Disconnect Sybil attack, the attacker A with knowledge of the `Network_Key` spoofs a DLPDU with the source address of the victim V. Knowing the Network_Key A can trick the neighbors of V into removing V from their tables. The neighbors of V will also forward the message to the *Network Manager* that V has left the network. V will be cleared from the *Network Managers* routing tables.

In [5], Bayou et al. present more attacks. This new attack allows an attacker to inject malicious Commands into a WirelessHART network. Additionally, the attack allows an *Insider Attacker* to leverage the knowledge of `Network_Key` to inject commands into the network.

The attacker will use broadcast session keys to enable this attack. Three attacks are presented: direct command injection bounced command injection and on-the-fly command injection. The three classes use the knowledge of a broadcast session key to spoof the sender of commands, tricking a *Field Device* into executing a malicious command.

In [32], the authors claim to have found a potential DOS attack similar to an attack briefly described in [35]. In this *exhaustion attack*, the attacker uses the `Well-Known-Key` to send Advertisements to a joining device. The joining device will respond to this (false) advertisement and be unable to join the legitimate network. An attacker can combine this attack with a de-authentication attack such as the one presented in [3].

We have summarized published security analyses of WirelessHART. Most of these works present attacks against the protocol in some form or another. We will look closer at the published attacks and establish the root cause. Table 3 present the previously published attacks, the type of vulnerability, and their root causes. We have used the threat terminology from the STRIDE model [18] to describe the root causes of the attacks. We have chosen the STRIDE model because it is complete and covers more threat classes than the Confidentiality Integrity Authentication (CIA) model.

The STRIDE model defines attacks of the following types: spoofing, tampering, replay attacks, information disclosure, denial of service, and elevation of privilege.

Table 3: A tabulation of published attacks against the WirelessHART protocol. (* Discussion, not a full attack)

Attack Name	Root Cause	Attacker	Published
Jamming Attacks	DoS	Outsider	[35]*
Wormhole Attacks	Spoofing	Insider	[35]*
Message Corruption	Tampering	Outsider	[35]*
Blackhole Attacks	Spoofing	Insider	[35]*
Disconnect Sybil Attack	Spoofing	Insider	[3]
Pre-Join exhaustion	DoS	Outsider	[35]*[32]
Direct CI	Spoofing	Insider	[5]
Bounced CI	Spoofing	Insider	[5]
On-the-fly CI	Tampering	Insider	[5]

Table 3 shows that most of the published attacks against WirelessHART are Spoofing attacks where the adversary knows the `Network_Key`. One final remark we want to make here is that in several papers, the authors have claimed that a specific attack is not possible, only for the attack to be published later. This is a further argument for formal protocol verification. A rigorous analysis of a protocol has a greater chance of discovering all possible attacks and preventing erroneous claims about the security of a protocol.

4 THREAT MODELS AND SECURITY ASSUMPTIONS

Since the WirelessHART specification does not properly specify the capabilities of a potential attacker, except for being unable to break cryptographic primitives or implementations, we have specified two potential attackers for our security analysis. We already introduced them briefly in Section 2.3, and here we provide the complete definitions.

The *Outside Attacker* is an active attacker, assumed to know no secrets from the system. The `Well-Known-Key` is explicitly known by all WirelessHART devices. The *Outside Attacker* can inject messages, replay messages, and discard messages. This is the same capabilities as a Dolev-Yao adversary [13].

The *Insider Attacker* is also an active attacker who knows one device's keys. We leave out of scope what specific device and how the attacker gained this knowledge. Apart from this, the *Insider Attacker* has the same capabilities as an *Outsider Attacker*, functioning as a Dolev-Yao adversary. The difference is that *Insider Attacker* can decrypt messages, spoof identities, and tamper with messages without being detected. This can be used for information disclosure and denial of service. In Table 4, we present all considered keys in the system and detail what keys the different Attacker knows.

	Outsider	Insider
`Join_Key`	U	K*
`Network_Key`	U	K
Network Manager Broadcast Session Key	U	K
Network Manager Unicast Session Key	U	K*
Gateway Unicast Session Key	U	K*
`Well-Known-Key`	K	K

Table 4: Keys known by Inside Attacker and Outsider Attacker. (K - system-wide key known, U - Unknown, K* - One key is known)

The WirelessHART standard state that the `Join_key` *can* be unique. Using the same, `Join_key` for multiple *Field Devices* in the network opens possibilities of nonce-reuses and replay attacks on other "Join Sequences". A `Join_Key` known to the adversary provides no security for the "Join Sequence". An attacker could intercept all messages transmitted during the "Join Sequence" and learn any new keys sent to a *Field Device*. The security of the "Join Sequence" hinges on the secrecy and uniqueness of the `Join_key`.

5 ANALYZING WIRELESSHART WITH TAMARIN

The Tamarin Prover [27] is a tool developed to model and verify the properties of security protocols. Protocols are modeled in a domain-specific language based on multi-set rewriting rules. The domain-specific languages include constructions such as Diffie-Hellman, symmetric and asymmetric encryption, and hash functions. Tamarin users can specify custom constructions, like Hash-based Message Authentication Codes (HMAC).

The rules defined for both the protocol and adversary can generate facts. In this context, a *fact* can be understood as the knowledge of an entity at a specific time. The rule $[Fr(\sim n)] \longrightarrow [Out(\sim n), A(\sim n)]$ generates a random number n. The number is then sent out on the communications network. The result is the *fact* $A(\sim n)$ that can be used in further rules, and that $\sim n$ is sent out on the network, letting the adversary know n. A full description of Tamarin's modeling language can be found in the Tamarin manual[3].

In Tamarin, the adversary is also defined using rules and facts. These rules and facts represent the adversary's knowledge at different states of protocol execution. The adversary in Tamarin is modeled as a Dolev-Yao adversary [13]. A Dolev-Yao adversary can intercept all messages, forge messages, impersonate entities, and decrypt messages using information known to the adversary.

[3] https://tamarin-prover.github.io/manual/index.html

Cryptographic primitives are assumed to be ideal and modeled algebraically [26].

The execution of rules produces an ordered sequence of facts, called a *trace*. A trace can be seen as one possible execution of the rules in the protocol. *Lemmas* are the properties of a trace that Tamarin tries to prove or find a counterexample. A lemma can specify that for all protocol executions, there must be no point where the adversary knows a secret message m. Tamarin will either prove that a property stated in a lemma holds or find a counterexample. Tamarin might not terminate since finding a proof or a counterexample is undecidable. The verification of properties always terminates in our analysis.

5.1 WirelessHART model

To model the WirelessHART protocol, we used Tamarin's built-in formulas. Tamarin does not have built-in support for AEAD operations but allows the user to specify formulas. We have added aead_encrypt/4, aead_decrypt/4, aead_verify/4, true/0, mic/2 and verify_mic/3. In this notation formula/n, n is the number of arguments to the formula. In the following equations, k denotes the key, n the nonce, aad the additional authenticated data, and p the plaintext. The formulas are specified as follows:

$$aead\_decrypt(k, n, aad, aead\_enc(k, n, aad, p)) = p$$

$$aead\_verify(k, n, aad, aead\_enc(k, n, aad, p)) = true$$

$$verify\_mic(mic(k, p), k, p) = true$$

We used the formulas for AEAD operations in the NPDU layer of the protocol, where both the encryption and authentication properties are used. We chose to use the MIC construction for the DLPDU, where an AEAD is used with the plaintext input set to NULL, providing integrity only.

We have also used the restriction Eq_testing() to force testing of equality in traces. We used this to force validation of verify_mic and aead_verify.

$$Eq\_testing : "All\ x\ y\ \#i.\ Eq(x, y)\ @\ i\ ==>\ x = y"$$

We focused on modeling the 'end-to-end security protocol', 'Join procedure', and 'Network key change operation'. These parts of WirelessHART are responsible for most security-critical operations. We chose to omit the replay protection scheme from our model since it uses sound constructions. For the 'Join procedure', we omitted messages after the step where the *Field Device* has accepted the new key since later steps of the procedure handles network management. When modeling the 'Network key change operation', we omitted the acknowledgment sent by the *Field Device*. The NLDPU and DLPDU headers were also simplified to reduce the number of terms.

The 'Pre-join exhaustion attack' and 'Disconnect Sybil attack' target the authentication of DLPDU messages. We, therefore, modeled the DLPDU message layer.

To keep our analysis manageable, we split our model into four parts: the End-to-End Security Protocol, the Join Sequence, the "Network Key Change Operation", and the DLPDU commands.

We will detail each part of the protocol in the following sections. We provide our model for further study[4].

[4]https://github.com/Gunzter/Formal-Verification-of-the-WirelessHART-Protocol

5.2 Security Analysis of the WirelessHART End-to-End Security Protocol

The end-to-end security protocol transports data and transmits configurations to devices. We modeled the DLPDU and NPDU protocol layers and created rules for sending and receiving messages for both unicast and broadcast sessions.

5.2.1 Security Lemmas and results. We have tested the following security properties of the WirelessHART end-to-end security protocol: payload secrecy and impersonation of a gateway. In Section 4, we discussed the notion of *Insider Attacker* and *Outsider Attacker*. These are the notions we will use in this section.

We analyzed the end-to-end security protocol's security using unicast and broadcast keys. For brevity, we only provide the lemmas for the broadcast case here.

The first lemma we present verifies the security of the sent command. It states that the adversary shall not be able to learn any command sent by the *Gateway*. A command known by the adversary implies that the broadcast key has been compromised. In other words, an *Outsider Attacker* can use its broadcast session key and decrypt the transmitted message and learn the command.

A similar lemma holds for unicast sessions so long as the unicast session keys used to encrypt the command are not compromised.

$$\begin{aligned}&lemma\ command\_secrecy :\\ &"All\ GW\ D\ k\ \#i\ \#j.\\ &SendCommand\_BC(GW, D, k)\ @i\ \ \&\ \ K(k)\ @j\\ &\ ==>\\ &Ex\ \#k.\ InsiderAttacker()\ @k"\end{aligned}$$

The lemma below verifies the authentication and integrity of a received command encrypted with a broadcast session key. A command accepted by a *Field Device* means that it either was legitimately sent by the Gateway or that an *Outsider Attacker* has compromised the broadcast session key and impersonated the *Gateway*.

A command sent over a unicast session is authentic as long as the unicast session key is not compromised, requiring the compromise of either the *Gateway* or the *Field Device*.

$$\begin{aligned}&lemma\ command\_authentication :\\ &"All\ GW\ D\ k\ \#i.\ ReceivedCommand\_BC(GW, D, k)\ @i\\ &\ ==>\ (Ex\ \#j.\ SendCommand\_BC(GW, D, k)\ @j\ \&\ \ j < i)\\ &|\ (Ex\ \#k.\ InsiderAttacker()\ @k)"\end{aligned}$$

5.3 Security Analysis of the WirelessHART Join Sequence

We modeled the Join Sequence by writing rules for creating the *Security Manager*, Gateway, and *Field Devices*. To model the execution of the protocol, we created the following rules: a *Field Device* sending a Join request message, the *Security Manager* receiving the Join request message and responding with encrypted keys, and a *Field Device* receiving the encrypted keys. We have modeled the

protocol so that the joining *Field Device* is provisioned with the Join Key out-of-band. This is done with a direct connection to a handheld Maintenance tool.

5.3.1 Security Lemmas and results. To verify the security of the WirelessHART Join Sequence, we will test the following security properties: key secrecy, key integrity, and resistance to impersonation attacks. These lemmas are tested with a unique and random `Join_Key` since the protocol is trivially broken when using a known `Join_Key`.

The first lemma verifies the key authentication and integrity from the joining *Field Device's* perspective. The lemma state that a received encrypted key means that either the encrypted keys were legitimately sent by the *Security Manager* to the *Field Device* or that the `Join_Key` has been compromised.

$$
\begin{aligned}
&\textit{lemma key\_authentication}: \\
&\text{"}\forall a\ k\ \#m. \\
&\textit{KeysReceived}(a,k)\ @m \\
&\Longrightarrow (\exists\ \#l.\ \textit{KeysSent}(a,k)\ @l) \mid \\
&(\exists\ \#j.\ \textit{InsiderAttacker}()\ @j)\text{"}
\end{aligned}
$$

The second lemma verifies the secrecy of the received key. A received key known by the adversary implies that it was previously sent by the *Security Manager* and that the `Join_Key` was compromised.

$$
\begin{aligned}
&\textit{lemma key\_secrecy}: \\
&\text{"}\forall a\ k\ \#m\ \#n. \\
&\textit{KeysReceived}(a,k)\ @m\ \&\ K(k)\ @n \\
&\Longrightarrow (\exists\ \#l.\ \textit{KeysSent}(a,k)\ @l)\ \& \\
&(\exists\ \#m.\ \textit{InsiderAttacker}()\ @m)\text{"}
\end{aligned}
$$

5.4 Security Analysis of the WirelessHART Network Key Change Operation

When modeling the "Network Key Change Operation", we have written rules corresponding to the *Security Manager* sending the change key message and the *Field Device* receiving the key change message. We wrote rules for the "Network Key Change Operation" over unicast and broadcast sessions.

5.4.1 Security Lemmas. The following lemma was used to verify the security of the "Network Key Change Operation". The operation can be done over unicast and broadcast sessions. We have only stated the broadcast lemmas here for brevity. The first lemma verifies the secrecy of the new key. Provided no *Field Device* has been compromised and revealed the broadcast key, the new key is secret against an *Outsider Attacker*. An *Outsider Attacker* can decrypt the message containing the new key and use that to decrypt future traffic.

$$
\begin{aligned}
&\textit{lemma key\_secrecy}: \\
&\text{"}\forall NM\ D\ k\ \#i\ \#j. \\
&\textit{SendKey\_BC}(NM,D,k)\ @i\ \&\ K(k)\ @j \\
&\Longrightarrow \\
&\exists\ \#k.\ \textit{InsiderAttacker}()\ @k\text{"}
\end{aligned}
$$

The following lemma verifies the authentication and integrity properties of the "Network Key Change Operation". Like in the case of key secrecy, the protocol is secure against an *Outsider Attacker* but does not hold against an *Outsider Attacker*. An *Outsider Attacker* can use its broadcast session key and `Network_Key` and spoof a message containing a fake key to the *Device*.

$$
\begin{aligned}
&\textit{lemma key\_authentication}: \\
&\text{"}\forall NM\ D\ k\ \#i.\ \textit{NewKeyReceived\_BC}(NM,D,k)\ @i \\
&\Longrightarrow (\exists\ \#j.\ \textit{SendKey\_BC}(NM,D,k)\ @j\ \&\ j < i) \\
&\mid (\exists\ \#k.\ \textit{InsiderAttacker}()\ @k)\text{"}
\end{aligned}
$$

5.5 Verification of Previously Published Attacks

We had to model more parts of the WirelessHART protocol to verify the previously published attacks. The Command Injection attacks were confirmed with the lemmas used to verify the end-to-end security protocol. The Disconnect Sybil attack and the Pre-Join Exhaustion attack required us to model the DLPDU control messages. DLPDU messages do not use the NPDU layer and consist of a command sent between two neighboring devices. The message is only MIC:ed using the `Network_Key` or the `Well-Known-Key`.

5.5.1 Security Lemmas. To verify the Disconnect Sybil Attack, we constructed a lemma that tests the authenticity of received disconnect DLPDU messages. The lemma verifies the authenticity of received disconnect commands. A disconnect command will be verified as authentic and implies that it previously has been legitimately transmitted or that an adversary has compromised the `Network_Key` and sent a forged disconnect command.

$$
\begin{aligned}
&\textit{lemma disconnect\_authentication}: \\
&\text{"}\forall A\ B\ \#i.\ \textit{ReceivedDisconnectCommand}(A,B)\ @i \\
&\Longrightarrow (\exists\ \#j.\ \textit{SendDisconnectCommand}(A,B)\ @j\ \&\ j < i) \\
&\mid (\exists\ \#k.\ \textit{InsiderAttacker}()\ @k)
\end{aligned}
$$

To verify the Pre-Join Exhaustion attack, we created a lemma that verifies the authenticity of the received Advertisement message. This lemma was trivially falsified since the Advertisement message is MIC:ed using the `Well-Known-Key`. An adversary can use the `Well-Known-Key` to create malicious Advertisement messages that the joining *Field Device* will accept.

lemma advertisement_authentication :
"*All A B #l.*
ReceivedAdvertisement(*A*, *B*) *@l*
==> (*Ex #m.SendAdvertisement*(*A*, *B*) *@m*)"

6 RESULTS AND DISCUSSION

In this section, we will present the results of our formal protocol analysis. We will discuss our findings for WirelessHART as a protocol and formal protocol verification as a technique.

6.1 Previously Published Attacks

As described in the previous section, we have modeled and analyzed the WirelessHART protocol. The three main parts of the protocol, the Join Sequence, the end-to-end Security Protocol, and the "Network Key Change Operation", were then analyzed using the lemmas presented in Section 5. We show the previously published attacks in Table 5. Our Tamarin analysis found all previously published attacks or variants of them.

Table 5: An overview of the results of our analysis previously shown in Table 3. (✓- Attack could be verified)

Attack Name	Root Cause	Source	Verified?
Disconnect Sybil Attack	Impersonation	[3]	✓
Pre-Join exhaustion	DoS	[32, 35]	✓
Direct CI	Impersonation	[5]	✓
Bounced CI	Impersonation	[5]	✓
On-the-fly CI	Tampering	[5]	✓

6.2 Novel Malicious Re-keying Attack

Modeling the "Network Key Change Operation", we found a new, previously unpublished attack. An *Insider Attacker* can utilize the known Broadcast Keys and maliciously transmit re-key commands to *Devices*. These *Field Devices* will change their keys according to the received command, bringing them out of synchronization with the *Network Manager* and the *Gateway(s)*. This attack can change any key used in the system. Maliciously re-keying of *Devices* and preventing the *Security Manager* and *Gateway* from communicating with re-keyed *Devices* is, in effect, a DoS attack on the parts of the network. We show the attack in Figure 4. The malicious adversary has compromised one *Field Device* and is now an *Outsider Attacker*.

(1) The *Outsider Attacker* sends two "Command 963 - Write Session", with Session Key 1* and Session Key 2* to the *Field Devices*. Next, a "Command 961 - Write Network_Key", containing `Network_Key*` is transmitted. These commands are encrypted with the current Session Key 1 and MIC:ed with the current `Network_Key`. These commands will be received by the targeted *Field Devices* and authenticated as coming from the *Security Manager*.
(2) The *Field Devices* will change keys to the maliciously provided Session Key 1*, Session Key 2*, and `Network_Key*`. The *Security Manager* and Plant Automation Host do not know these keys, so the legitimate owners of the *Field Devices* cannot contact the *Field Devices*.

Recovering from this attack requires finding and recovering the compromised *Outsider Attacker Field Device* and a complete re-provisioning of all *Field Devices* in the network.

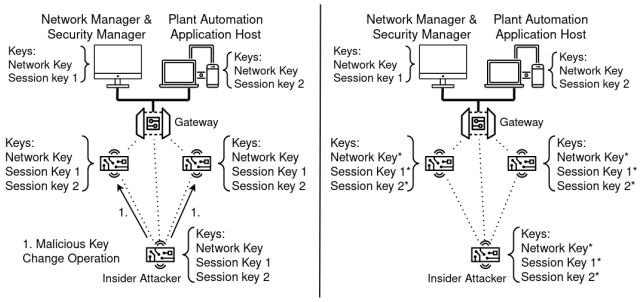

Figure 4: Malicous re-keying attack.

6.3 Security implications for WirelessHART

We can see that the root cause of many attacks is compromised broadcast keys. Symmetric key encryption needs key secrecy, and a compromised *Field Device* breaks this requirement, as in the case with *Insider Attacker*. As we show in Table 4, not only `Network_Key` is revealed to an *Insider Attacker*, but also the *Network Manager* unicast and broadcast keys and the session keys. The authors of the WirelessHART specification have failed to account for this fact. Given the above insight, we conclude that the "Network Key Change Operation" when Network Manager Broadcast keys are compromised is insecure. An *Outsider Attacker* can decrypt the messages in the Key Exchange messages with the Network Manager Broadcast key. The WirelessHART specification recommends periodic key changes [20] (Appendix A.3) to ensure the system's secure operation. This would not help the network's security since new keys are transmitted encrypted using a key known to the attacker, and the new key would be revealed to the attacker if the old key was compromised.

A potential remedy would be to use Network Manager Unicast keys to update the keys and limit communication to unicast sessions only. Individual key changes with unicast keys will increase the network overhead. Still, the owner and operator of a network can avoid revealing a key to a compromised device, i.e., a *Field Device* controlled by *Insider Attacker*.

Furthermore, there is a severe issue of spoofing attacks. Broadcast Session Keys are used to encrypt and protect the integrity of messages using a symmetric algorithm, which is not secure. It is difficult to imagine how these attacks could be mitigated without considerable changes to the WirelessHART specification.

Lastly, the Pre-Join exhaustion attacks can be prevented by provisioning the current `Network_Key` to the joining *Field Device*. This will prevent any attacker in the proximity from abusing the well-known key to spoof messages.

The Disconnect Sybil attack is more difficult to remedy. A *Field Device* does not share a unicast key with its neighbors, which also may change over time, so only the `Network_Key` can be used to authenticate messages from a neighbor device.

7 RELATED WORK

We have divided the preliminary research for this paper into two categories. Prior work on WirelessHART security and Formal Protocol Verification.

7.1 WirelessHART Security

In Section 3, we have already covered published attacks and security analyses of WirelessHART [3, 5, 32, 34, 35]. In this section, we will expand the discussion to cover attacks that target the lower layer of the protocol stack, such as jamming attacks.

Several other papers have studied the security of WirelessHART, summarizing published attacks but not presenting any new attacks. In [1], the authors analyze several protocols for IoT and ICS, among them WirelessHART. NetSIM, a simulator for WirelessHART SCADA systems, is presented in [4]. The simulator is used to demonstrate the Disconnect Sybil attack published in [3] Intrusion Detection Systems (IDS) and network monitoring systems for WirelssHART have been studied [32]. A jamming attack against WirelessHART was published in 2021 [9].

7.2 Formal Protocol verification of ICS & IoT protocols

Formal Protocol verification has been used with great success to find attacks in other protocols for IoT and ICS.

In their paper[21], the authors write about their process of verifying IoT protocols. They use Tamarin to analyze CoAP over DTLS, SigFox, LoRa, and MQTT. LoRaWAN has been analyzed [7] and [40], where more vulnerabilities were found. The proposed key establishment protocol EDHOC has been modeled and formally verified [6, 29]

The Virtual Private Network protocol Wireguard has also been formally verified [14]. Tamarin has been used during the development process of TLS1.3 [11]. The rationale was to verify the protocol before it was finalized and deployed. Then the protocol designers could address any issues before the protocol was deployed.

8 CONCLUSION

WirelessHART has been studied in the literature, and multiple attacks have been found in several published works. Formal verification of WirelessHART has been done, albeit in a limited scope. To our knowledge, our work is the first to model and verify all parts of WirelessHART.

We have modeled the WirelessHART end-to-end security protocol, the WirelessHART Joining sequence, and the WirelessHART Network Key Change Operation and tested them against both an *Insider Attacker* and *Outsider Attacker*.

We have shown that the previously published attacks can be found using formal protocol verification. The attacks were of varying types and against different parts of the WirelessHART protocol.

Further, we have extended the discussion about the threat model of WirelessHART. We have introduced a distinction in adversary models with the *Insider Attacker* and the *Outsider Attacker*. These attackers' knowledge necessitates considering two separate cases when analyzing the protocol.

When formally verifying the Network Key Change Operation, we found a new attack on the WirelessHART protocol. An Insider Attacker can maliciously change the keys of non-compromised devices in the network, making them uncontactable by the rest of the network, including the Security Manager. This attack is severe since only a single device needs to be compromised for an Insider Attacker to put the entire network offline.

The WirelessHART protocol would benefit from an update to the standard. New mechanisms with integrity protection of all messages are needed to prevent tampering and stronger message authentication. A topic for further research would be implementing the improvements and making a formal verification of the new version of the protocol.

9 ACKNOWLEDGMENT

We would like to thank Simon Bouget for his valuable help with Tamarin and the anonymous reviewers for their helpful comments. This work has been supported by framework grant RIT17-0032 from the Swedish Foundation for Strategic Research.

REFERENCES

[1] Cristina Alcaraz and Javier Lopez. 2010. A security analysis for wireless sensor mesh networks in highly critical systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 40, 4 (2010), 419–428.

[2] David Basin, Ralf Sasse, and Jorge Toro-Pozo. 2021. The EMV Standard: Break, Fix, Verify. In *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, New York, NY, USA, 1766–1781. https://doi.org/10.1109/SP40001.2021.00037

[3] Lyes Bayou, David Espes, Nora Cuppens-Boulahia, and Frédéric Cuppens. 2015. Security issue of wirelesshart based SCADA systems. In *International Conference on Risks and Security of Internet and Systems*. Springer, Berlin, Germany, 225–241.

[4] Lyes Bayou, David Espes, Nora Cuppens-Boulahia, and Frédéric Cuppens. 2015. WirelessHART NetSIM: a WirelessHART SCADA-based wireless sensor networks simulator. In *Security of Industrial Control Systems and Cyber Physical Systems*. Springer, Berlin, Germany, 63–78.

[5] Lyes Bayou, David Espes, Nora Cuppens-Boulahia, and Frédéric Cuppens. 2016. Security analysis of WirelessHART communication scheme. In *International Symposium on Foundations and Practice of Security*. Springer, Berlin, Germany, 223–238.

[6] Alessandro Bruni, Thorvald Sahl Jørgensen, Theis Grønbech Petersen, and Carsten Schürmann. 2018. Formal verification of ephemeral Diffie-Hellman over COSE (EDHOC). In *International Conference on Research in Security Standardisation*. Springer, Berlin, Germany, 21–36.

[7] Ismail Butun, Nuno Pereira, and Mikael Gidlund. 2018. Analysis of LoRaWAN v1. 1 security. In *Proceedings of the 4th ACM MobiHoc Workshop on Experiences with the Design and Implementation of Smart Objects*. ACM, New York, NY, USA, 1–6.

[8] Deji Chen, Mark Nixon, Song Han, Aloysius K Mok, and Xiuming Zhu. 2014. WirelessHART and IEEE 802.15. 4e. In *2014 IEEE International conference on industrial technology (ICIT)*. IEEE, IEEE, New York, NY, USA, 760–765.

[9] Xia Cheng, Junyang Shi, Mo Sha, and Linke Guo. 2021. Launching smart selective jamming attacks in wirelesshart networks. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, New York, NY, USA, 1–10.

[10] Cas Cremers and Martin Dehnel-Wild. 2019. Component-based formal analysis of 5G-AKA: Channel assumptions and session confusion. In *Network and Distributed Systems Security (NDSS) Symposium 2019*. Internet Society, Internet Society, Reston, VA, USA, 1–15.

[11] Cas Cremers, Marko Horvat, Jonathan Hoyland, Sam Scott, and Thyla van der Merwe. 2017. A comprehensive symbolic analysis of TLS 1.3. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, New York, NY, USA, 1773–1788.

[12] Alessandro Di Pinto, Younes Dragoni, and Andrea Carcano. 2018. TRITON: The first ICS cyber attack on safety instrument systems. In *Proc. Black Hat USA*, Vol. 2018. Black Hat, San Francisco, CA, USA, 1–26.

[13] Danny Dolev and Andrew Yao. 1983. On the security of public key protocols. *IEEE Transactions on information theory* 29, 2 (1983), 198–208.

[14] Jason A. Donenfeld and Kevin Milner. 2017. *Formal Verification of the WireGuard Protocol*. Technical Report. www.wireguard.com. https://www.wireguard.com/papers/wireguard-formal-verification.pdf

[15] International Organization for Standardization. 1994. *ISO/IEC 7498-1:1994 - Information technology — Open Systems Interconnection — Basic Reference Model: The Basic Model*. Standard. International Organization for Standardization.

[16] Sagarika Ghosh and Srinivas Sampalli. 2019. A Survey of Security in SCADA Networks: Current Issues and Future Challenges. *IEEE Access* 7 (2019), 135812–135831. https://doi.org/10.1109/ACCESS.2019.2926441

[17] FieldComm Group. 2020. *HART Communication Protocol Specification.* Specification. FieldComm Group, Geneva, CH.

[18] Shawn Hernan, Scott Lambert, Tomasz Ostwald, and Adam Shostack. 2006. Threat modeling-uncover security design flaws using the stride approach. *MSDN Magazine-Louisville* 15, 1 (2006), 68–75.

[19] Abdulmalik Humayed, Jingqiang Lin, Fengjun Li, and Bo Luo. 2017. Cyber-Physical Systems Security—A Survey. *IEEE Internet of Things Journal* 4, 6 (2017), 1802–1831. https://doi.org/10.1109/JIOT.2017.2703172

[20] International Electrotechnical Commission (IEC). 2016. *Industrial networks – Wireless communication network and communication profiles – WirelessHART.* Standard. International Electrotechnical Commission (IEC), Geneva, CH.

[21] Jun Young Kim, Ralph Holz, Wen Hu, and Sanjay Jha. 2017. Automated analysis of secure internet of things protocols. In *Proceedings of the 33rd Annual Computer Security Applications Conference.* ACM, New York, NY, USA, 238–249.

[22] Ralph Langner. 2011. Stuxnet: Dissecting a cyberwarfare weapon. *IEEE Security & Privacy* 9, 3 (2011), 49–51.

[23] Tomas Lennvall, Stefan Svensson, and Fredrik Hekland. 2008. A comparison of WirelessHART and ZigBee for industrial applications. In *2008 IEEE international workshop on factory communication systems.* IEEE, New York, NY, USA, 85–88.

[24] Gaoqi Liang, Steven R Weller, Junhua Zhao, Fengji Luo, and Zhao Yang Dong. 2016. The 2015 Ukraine blackout: Implications for false data injection attacks. *IEEE Transactions on Power Systems* 32, 4 (2016), 3317–3318.

[25] Fuyuan Luo, Tao Feng, and Lu Zheng. 2021. Formal Security Evaluation and Improvement of Wireless HART Protocol in Industrial Wireless Network. *Security and Communication Networks* 2021 (2021), 1–15.

[26] Simon Meier. 2013. *Advancing automated security protocol verification.* Ph.D. Dissertation. ETH Zurich.

[27] Simon Meier, Benedikt Schmidt, Cas Cremers, and David Basin. 2013. The TAMARIN prover for the symbolic analysis of security protocols. In *International Conference on Computer Aided Verification.* Springer, Berlin, Germany, 696–701.

[28] Reimund Neugebauer, Sophie Hippmann, Miriam Leis, and Martin Landherr. 2016. Industrie 4.0-From the perspective of applied research.

[29] Karl Norrman., Vaishnavi Sundararajan., and Alessandro Bruni. 2021. Formal Analysis of EDHOC Key Establishment for Constrained IoT Devices. In *Proceedings of the 18th International Conference on Security and Cryptography - SECRYPT,.* INSTICC, SciTePress, Setúbal, Portugal, 210–221.

[30] Institute of Electrical and Electronics Engineers. 2006. *IEEE 802.15.4-2006 - Local and metropolitan area networks - Specifications for Low Rate Wireless Personal Area Networks (WPANs).* Specification. Institute of Electrical and Electronics Engineers.

[31] Boris Ramos, Stefano Savazzi, JM Winter, Verónica Ojeda, Marjorie Chalen, Edison Del Rosario, et al. 2018. A Perfomance Comparison of WirelessHART and ZigBee in Oil Refinery. In *2018 IEEE-APS Topical Conference on Antennas and Propagation in Wireless Communications (APWC).* IEEE, New York, NY, USA, 846–849.

[32] Duarte Raposo, André Rodrigues, Soraya Sinche, Jorge Sá Silva, and Fernando Boavida. 2018. Securing wirelessHART: Monitoring, exploring and detecting new vulnerabilities. In *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA).* IEEE, New York, NY, USA, 1–9.

[33] Apala Ray, Johan Åkerberg, Mats Björkman, and Mikael Gidlund. 2016. Future research challenges of secure heterogeneous industrial communication networks. In *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA),* Vol. 1. IEEE, New York, NY, USA, 1–6.

[34] Shahid Raza. 2010. Secure Communication in WirelessHART and its Integration with Legacy HART. *Technical Report* 1, 2010 (2010), 103.

[35] Shahid Raza, Adriaan Slabbert, Thiemo Voigt, and Krister Landernäs. 2009. Security considerations for the WirelessHART protocol. In *2009 IEEE Conference on Emerging Technologies & Factory Automation.* IEEE, New York, NY, USA, 1–8.

[36] Ulrich Sendler et al. 2013. *Industrie 4.0.* Springer, Berlin, Germany.

[37] Joseph Slowik. 2019. *Evolution of ICS attacks and the prospects for future disruptive events.* Technical Report. Threat Intelligence Centre Dragos Inc, Hanover, MD, USA.

[38] George Stergiopoulos, Dimitris A. Gritzalis, and Evangelos Limnaios. 2020. Cyber-Attacks on the Oil & Gas Sector: A Survey on Incident Assessment and Attack Patterns. *IEEE Access* 8 (2020), 128440–128475. https://doi.org/10.1109/ACCESS.2020.3007960

[39] Teerawat Thepmanee, Sawai Pongswatd, Farzin Asadi, and Prapart Ukakimaparn. 2022. Implementation of control and SCADA system: Case study of Allen Bradley PLC by using WirelessHART to temperature control and device diagnostic. *Energy Reports* 8 (2022), 934–941.

[40] Stephan Wesemeyer, Ioana Boureanu, Zach Smith, and Helen Treharne. 2020. Extensive Security Verification of the LoRaWAN Key-Establishment: Insecurities & Patches. In *2020 IEEE European Symposium on Security and Privacy (EuroS&P).* IEEE, New York, NY, USA, 425–444.