Mälardalen University Licentiate Thesis
No.135

SECURING COMMUNICATION IN IP-CONNECTED INDUSTRIAL WIRELESS SENSOR NETWORKS

Shahid Raza

June 2011

Swedish Institute of Computer Science

SICS

MÄLARDALEN UNIVERSITY
SWEDEN

School of Innovation, Design, and Engineering

ISBN 978-91-7485-021-5
ISSN 1651-9256
Printed by Mälardalen University, Västerås, Sweden

Populärvetenskaplig sammanfattning

Med introduktionen av trådlösa sensornätverk (WSN) och framgången för trådlös kommunikation såsom Wi-Fi och Bluetooth i lokala och personliga nätverk, inleddes mer allvarliga ansträngningar för att tillämpa trådlös kommunikation i säkerhetskritiska industriella nätverk. Detta har bland annat resulterat i standardiseringen av WirelessHART. Andra standardiseringar inkluderar ISA 100.11a och ZigBee. Med tanke på karaktären av trådlös kommunikation och känsligheten i industriella miljöer, får säkerhet i dessa nätverk större betydelse än i andra nätverk.

I denna avhandling studeras säkerhetsfrågor i industriella WSN i allmänhet och i IP-anslutna WSN i synnerhet. För närvarande är WirelessHART den enda godkända standarden för säker trådlös kommunikation i industriella WSN. Vårt arbete inleds med analys av säkerhetsmekanismer i WirelessHART. Vi föreslår lösningar för brister i säkerheten i WirelessHART och design av de saknade säkerhetskomponenterna. Särskilt har vi designat, implementerat och utvärderat den första öppna säkerhetshanteraren för WirelessHART-nät.

Med standardisering av IP i WSN (6LoWPAN) och tillkomsten av Internet of Things, blir behovet av IP-kommunikation i industriella WSN accentuerat. Den nyligen föreslagna standarden ISA 100.11a är IP-baserad i grunden. Standardiseringsansträngningar pågår även för att använda IP i WirelessHART och Zigbee. Nyligen har WSN och traditionella IP-nät blivit mer tätt integrerade genom IPv6 och 6LoWPAN. Vi ser behov av att ha samverkande standardiserad säker IP-kommunikation i industriella WSN. IP Security (IPSec) är en obligatorisk säkerhetslösning för IPv6. Vi föreslår användningen av IPsec för 6LoWPAN i industriella WSN. Det är dock inte rimligt att använda IPsec i sin nuvarande form i resurssvaga WSN. Förutom att tillhandahålla säkerhetslösningar

för WirelessHART, innefattar denna avhandling också design, implementation och utvärdering av Lightweight IPsec för 6LoWPAN-aktiverade WSN. Våra resultat visar att Lightweight IPsec är en förnuftig och praktisk lösning för WSN.

Abstract

With the advent of Wireless Sensor Networks (WSN) and success of wireless communication in the local and personal area networks such as Wi-Fi and Bluetooth more serious efforts to apply standard wireless communication in sensitive industrial networks were initiated. This effort resulted in the standardization of WirelessHART. Other standardization efforts include ISA 100.11a and ZigBee. Keeping in mind the nature of wireless communication and sensitivity of industrial environments security of these network gets greater importance.

In this thesis we work on security issues in industrial WSN in general and IP-connected WSN in particular. Currently WirelessHART is the only approved standard for secure wireless communication in industrial WSNs. We start our work with the analysis of security mechanisms in WirelessHART. We propose solutions for the security shortcomings in WirelessHART, and design and implement the missing security components. Particularly, we specify, design, implement, and evaluate the first open security manager for WirelessHART networks.

With the standardization of IP in WSNs (6LoWPAN) and birth of Internet of Things the need for IP communication in industrial WSN is getting importance. The recently proposed ISA 100.11a standard is IP-based since its inception. Also standardization efforts are in progress to apply IP in WirelessHART and Zigbee. Recently, WSNs and traditional IP networks are more tightly integrated using IPv6 and 6LoWPAN. We realize the importance of having an interoperable standardized secure IP communication in industrial WSNs. IP Security (IPsec) is a mandatory security solution in IPv6. We propose to use IPsec for 6LoWPAN enabled industrial WSNs. However, it is not meaningful to use IPsec in its current form in resource constrained WSNs. In addition to providing security solutions for WirelessHART, in this thesis we also specify, design, implement, and extensively evaluate lightweight IPsec that enables end-to-end secure communication between a node in a 6LoWPAN and a device

in the traditional Internet. Our results show that lightweight IPsec is a sensible and practical solution for securing WSN.

Acknowledgments

I am very grateful to all the people in SICS, MdH, and ABB who were associated with this work and guided me throughout the thesis period, but it is worth mentioning some of the people who were really kind and helpful.

Firstly, I like to express my gratitude to my supervisors Prof. Mats Björkman, Dr. Thiemo Voigt, and Dr. Christian Gehrmann for guiding and helping me during my studies. I specifically thank Dr. Thiemo Voigt for supporting me all the way from day first. I am also grateful to Sverker Janson and Thiemo Voigt for giving me chance to work in the Swedish Institute of Computer Science (SICS). Secondly, I am very thankful to my co-workers Simon Duquennoy, Dogan Yazar, and Adriaan Slabbert from SICS; Utz Roedig and Tony Chung from Lancaster University; Krister Landernäs and Mikael Gidlund for ABB, and Prof. Gianluca Dini from University of Pisa. I am thankful to their full support, encouragement, and the effort they put to help, correct, comment, and clarify my work. Last but not the least, I am really grateful to the people in CSL and CNS group at SICS for their help and support whenever I was in need of it; especially I am thankful to Joakim Eriksson, Niclas Finne, and Nicolas Tsiftes for supporting my work and giving valuable inputs.

I would like to dedicate this work to my parents and family. They are really special to me and I am thankful to them for their love and support.

Shahid Raza
Stockholm, June, 2011

This work has been performed within the SICS Center for Networked Systems funded by VINNOVA, SSF, KKS, ABB, Ericsson, Saab SDS, TeliaSonera, T2Data, Vendolocus, and Peerialism. This work has been partially supported by the European Commission with contract FP7-2007-2-224053 (CONET) and 224282 (GINSENG).
The SICS is sponsored by TeliaSonera, Ericsson, Saab SDS, FMV (Defence Materiel Administration), Green Cargo (Swedish freight railway operator), ABB, and Bombardier Transportation.

List of Publications

Papers Included in the Licentiate Thesis[1]

Paper A *Security Considerations for the WirelessHART Protocol*.
Shahid Raza, Adriaan Slabbert, Thiemo Voigt, Krister Landernäs. In 14th IEEE International Conference on Emerging Techonologies and Factory (ETFA'09), September 2009, Mallorca, Spain.

Paper B *Design and Implementation of a Security Manager for WirelessHART Networks*.
Shahid Raza, Thiemo Voigt, Adriaan Slabbert, Krister Landernäs. In 5th IEEE International Workshop on Wireless and Sensor Networks Security (WSN' dS 2009), in conjunction with MASS'2009, 12-15 Oct 2009, Macau SAR, P.R.C..

Paper C *Securing Communication in 6LoWPAN with Compressed IPsec*.
Shahid Raza, Simon Duquennoy, Tony Chung, Dogan Yazar, Thiemo Voigt, Utz Roedig. In 7th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS '11), 27-29 June 2011, Barcelona, Spain.

[1] The included articles have been reformatted to comply with the licentiate layout

Additional Papers, not Included in the Licentiate Thesis

Conferences and Workshops

- **Shahid Raza**, Gianluca Dini, Thiemo Voigt, and Mikael Gidlund. *Secure Key Renewal in WirelessHART*. In: Real-time Wireless for Industrial Applications (RealWin'11), CPS Week, 11-16 April 2011, Chicago, Illinois, USA.

- **Shahid Raza**, Thiemo Voigt, and Utz Roedig. *6LoWPAN Extension for IPsec*. In: Interconnecting Smart Objects with the Internet Workshop, 25 March 2011, Prague, Czech Republic.

- **Shahid Raza** and Thiemo Voigt. *Interconnecting WirelessHART and Legacy HART Networks*. In: 1st International Workshop on Interconnecting Wireless Sensor Network in conjunction with DCOSS'10., 21-23 June 2010, UC Santa Barbara, California, USA.

- Joakim Eriksson, Fredrik Österlind, Thiemo Voigt, Niclas Finne, **Shahid Raza**, Nicolas Tsiftes, and Adam Dunkels. *Demo abstract: accurate power profiling of sensornets with the COOJA/MSPSim simulator*. In: Sixth IEEE International Conference on Mobile Ad-hoc and Sensor Systems (IEEE MASS 2009), 12-15 Oct 2009, Macau SAR, P.R.C..

SICS Technical Reports

- **Shahid Raza**, Tony Chung, Simon Duquennoy, Dogan Yazar, Thiemo Voigt, Utz Roedig. *Securing Internet of Things with Lightweight IPsec*. ISSN No. 1100-3154, SICS Technical Report (T2010:08), 2010.

- **Shahid Raza**. *Secure Communication in WirelessHART and its Integration with Legacy HART*. ISSN No. 1100-3154, SICS Technical Report (T2011:01), 2011.

Contents

I

Thesis

Chapter 1

Introduction

A typical Wireless Sensor Networks (WSN) is a network of resource constrained sensor nodes and a base station usually connected together through lossy wireless links. Industrial WSN, though resource constrained, is a bidirectional network of relatively powerful devices with fairly stable wireless links and usually has a central network controller. In a bidirectional industrial WSN sensor nodes receive control messages from the central controller. WirelessHART [1], currently the only WSN standard designed primarily for industrial process automation and control, consists of a central network and a security manager on a wired network, wireless field devices and access points, and a gateway between wired and wireless networks. IPv6 over Low-power Wireless Personal Area Network (6LoWPAN) [2] that introduces IP in WSN has been standardized. The industrial community has also realized the importance of IP communication. This is apparent from the fact that the proposed industrial WSN standard ISA 100.11a is IP based. Also, efforts are being carried on to apply IP communication in WirelessHART, formally named HART IP, and in Zigbee named ZigBee IP.

Real world WSN deployments require secure communication as the wireless signal can easily be intercepted by an intruder and the contents can be revealed and modified. Due to the sensitive nature of industrial environments security is particularly important for industrial WSNs as a breach of security may result in catastrophic results. In this thesis we work on secure communication in industrial WSN in general and IP-connected WSN in particular. We target WirelessHART as it is currently the only WSN standard for industrial networks. Realizing the trend towards all IP networks, we design and imple-

ment lightweight IP Security (IPsec) for IP-connected WSNs.

Security specifications in the WirelessHART standard are incomplete and not well organized. The provided security is spread throughout the WirelessHART specifications [3] and the standard lacks a comprehensive document that explains and specifies the security. The network designers and device vendors encounter ambiguities regarding the complete security architecture of the WirelessHART, the strength of the provided security, the security keys needed, etc. The WirelessHART standard has been recently released and we are the first to analyze and clarify its security features. We discuss the strengths and weaknesses of the provided security mechanisms in the form of a threat analysis: we analyze the WirelessHART security against well known threats in the wireless medium and propose recommendations to mitigate the impact of these threats. We also explain the security keys and their usage as the standard does not illustrate them clearly.

The WirelessHART standard specifies the need of a Security Manager (SM) to provide key management; but the standard does not explicate a Key Management System (KMS). Also, the standard lacks the specifications and design of the SM. The standard emphasizes that the connections between the SM and the Network Manager (NM), the Gateway and the NM, and the Gateway and host applications must be secured; but it does not specify the ways to secure these connections. We design and implement the first open SM for WirelessHART that provides a complete KMS, authentication of wireless devices, and solutions to secure the wired part of the network. We specify how the SM interacts with the NM and what parameters are exchanged during these interactions. We experimentally evaluate the performance of our SM against different cryptographic algorithms. Our results show that our SM meets all related timing requirements of the standard.

With the inception of IP in HART, ZigBee-IP [4], and ISA 100.11a [5] it is evident that industry wants IP in industrial WSNs. However, these standards for industrial WSNs propose to use different security mechanisms that are not interoperable. ISA 100.11a specifies new symmetric and asymmetric security solutions rather than using standardized IPsec. HART IP currently only specifies IP communication in the wired part of the WirelessHART network and does not define security specifications yet. ZigBee IP secures communication with heavyweight state-of-art security protocols such as TLS, EAP, etc. [4] without caring for limited MTU size of 127 bytes and resource constrained nature of WSN. Currently, ZigBee IP does not use IPsec.

We believe that all these protocols can operate over IP. 6LoWPAN enables IP communication in WSNs. Available IPv6 protocol stacks can use IPsec to

secure data exchanges. It is beneficial to use IPsec because the existing endpoints on the Internet do not need to be modified to communicate securely with the WSN. Moreover, using IPsec, true end-to-end security is implemented and the need for a trustworthy gateway is removed. Thus, it is desirable to extend 6LoWPAN such that IPsec communication with IPv6 nodes is possible. We provide the first compressed lightweight design, implementation, and evaluation of 6LoWPAN enabled IPsec.

1.1 Contributions

We provide secure communication in industrial WSNs particularly in WirelessHART networks. We also develop a compressed version of IPsec for the IP-connected WSNs. The main contributions of this thesis are as follows.

1. **WirelessHART Security Analysis**
   We provide comprehensive security specifications for WirelessHART. We perform threat analysis of the WirelessHART security where we analyze the provided security mechanisms against well known threats in the wireless medium and identify the loopholes. We recommend solutions to overcome these shortcomings. This work has been published in ETFA'09.

2. **Design and Implementation of a WirelessHART Security Manager**
   We specify, design, implement, and evaluate the first open SM for the WirelessHART networks. Our evaluation shows that the provided SM is capable of securing both the wireless and the wired parts of the WirelessHART network. We have published this work in a very targeted WSN Security Workshop, WSNS'09.

3. **Lightweight IPsec for IP-connected WSN**
   We give specifications of IPsec for 6LoWPAN including definitions for AH and ESP extension headers. Prior to this work no specification for IPsec in the context of 6LoWPAN existed. We present the first implementation of IPsec for 6LoWPAN networks. We extensively evaluate and show that it is practical and feasible to secure WSN communication using IPsec. We have published this work in one of the top sensor networking conference, DCOSS'11.

4. **Standardization of the Proposed Solutions**
   In order to better understand the current status of standardization efforts

and make people aware of our work I have attended a WirelessHART working group meeting, the Internet Architecture Board (IAB) official workshop and tutorial, and IETF 80th meeting. This helped a lot to make people understand the importance of our work and later inclusion of our security solutions in standard specifications. I have also published my IPsec work in an IAB workshop [6].

1.2 Thesis Outline

The outline of the thesis is as follows. In Chapter 2 we present the background of the technologies used in this thesis. We describe wireless network for industrial communication particularly WirelessHART, IPv6 and its usage in WSN, and secure communication in such networks. We elaborate our work on secure communication in WSNs in Chapter 3 where we give an overview of our WirelessHART threat analysis, WirelessHART security manager, and lightweight IPsec for IP-connected WSNs. In Chapter 4 we present conclusions and future work. We present technical overviews of the papers that we include in this thesis in Chapter 5. We include these papers in Chapters 6 - 8.

Chapter 2

Background

In this chapter we give an introduction of the technologies used in this thesis. We provide overview of industrial WSNs particularly WirelessHART networks. We also discuss IP-connected WSN enabling technologies such as IPv6, 6LoWPAN, IPsec, etc. This background is needed to help understand our solutions for secure industrial WSNs that we present in Chapter 3.

2.1 Wireless Sensor Networks

A Wireless Sensor Networks (WSN) is a network of resource constrained sensor nodes and a base station that connects them with a traditional computer network. A typical WSN is a unidirectional network where sensor nodes collect sensor readings and send them to the base station through lossy wireless links. We consider such networks as first generation WSNs that primarily target environmental monitoring and deployed in volcanoes, forests, deserts, seas, etc. No standardized addressing and routing schemes exit for the first generation WSN.

Current WSNs are more successful than the first generation WSNs primarily because these are deployed within human environments. Applications of such WSNs include industrial automation, building and bridge monitoring, urban sensing, human sensing, etc. We consider these networks as second generation WSNs. Some standard protocols exist for the second generation WSNs primarily in the industrial realm, see Section 2.2. Future WSNs are tending towards IP enabled WSN. 6LoWPAN enables routing of IP packets between

WSN and traditional IP networks, see Section 2.3.

2.2 Wireless in Industrial Sensor Networks

Wiring in industrial networks can be a mess. The success of wireless technologies such as Wi-Fi and Bluetooth intrigued the need for wireless communication in industrial environments. However, due the resource constrained nature of WSN and specific real time requirements of industrial automation networks protocols such as Bluetooth, WiFi, etc. are not applicable in such environments [1].

Applications of industrial WSN include oil and gas production wells, tank farms, separator column monitoring, boiler and furnace monitoring, valve position monitoring, environment and energy monitoring, asset management, advanced diagnostics, etc. In this thesis we primarily focus on WirelessHART as it is currently the only approved standard for industrial WSNs.

2.2.1 WirelessHART

WirelessHART [1] is the first approved open standard for WSNs designed primarily for industrial process automation and control systems. The WirelessHART network is a collection of wired entities: Network Manager (NM), Gateway, Security Manager (SM), and Plant Automation Hosts (PAH); and wireless devices: Field devices, Adapters, Routers, Access Points, and handheld devices. A sample WirelessHART network is shown in Figure 2.1

The NM provides overall management, network initialization functions, network scheduling and monitoring, and resource management. The NM collaborates with the SM for the management and distribution of security keys. The wireless devices are connected using a mesh network where each device acts as a router and must be in one-hop range with at least two neighboring devices to provide path diversity. The protocol stack is based on the seven layer OSI stack with additional security and MAC sub layers. WirelessHART is a self healing and self organizing wireless protocol, in that the devices are able to find neighbors and establish paths with them, and detect network outrages and reroute.

[1] In this thesis we only focus on standardized solutions and skip relevant proprietary workouts.

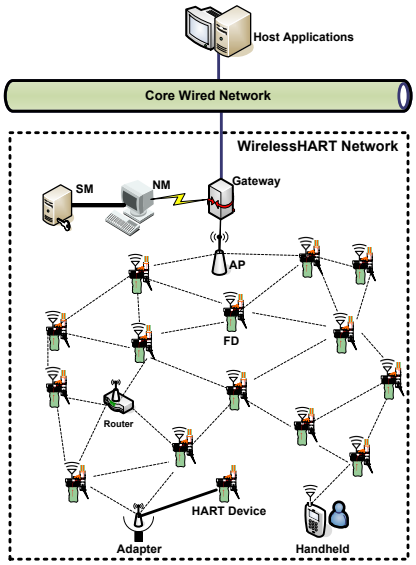


Figure 2.1: Complete WirelessHART network with wireless and wired parts

2.2.2 Other Industrial WSN standards

Other proposed global standards for industrial WSNs are ISA 100.11a and ZigBee. ISA 100.11a [5] is mainly designed for industrial automation and control and is very much similar to its competitor WirelessHART. Like WirelessHART it has a central system manager for the overall management of network, a security manager for device authentication and key management, field devices equipped with sensors, wireless routing devices, handheld devices, and a gateway. However, ISA 100.11a supports stronger security. Section 2.6.1 highlights security in ISA 100.11a.

ZigBee, considered not appropriate for industrial settings[7], has some inherited features that can be exploited to make it fit for industrial WSNs. WirelessHART and ISA 100.11a are more appropriate for the industrial automation as they follow strict real time requirements and are more robust against interference than ZigBee as they use frequency hopping. ZigBee is appropriate for home and building automation, smart metering, smart grids, etc.

2.3 IP in Wireless Sensor Network

IPv6 over Low-power wireless Personal Area Networks (6LoWPAN) enables routing of IPv6 packets over IEEE 802.15.4 networks. This endows end-to-end IP communication between a device in a 6LoWPAN and a device on the traditional Internet. All industrial WSN protocols discussed in Section 2.2 share the same IEEE 802.15.4 physical layer. Though it makes less sense to have multiple standards (particularly true for WirelessHART and ISA 100.11a) for more or less the same purpose the standards should at least be interoperable. These protocols can operate securely over IP. However, due to packet size limitations at the 802.15.4 link-layer [2] and the resource constrained nature of wireless sensor devices, traditional IP is too heavy for such networks. Recently, WSNs and traditional IP networks are more tightly integrated using IPv6 and 6LoWPAN [2].

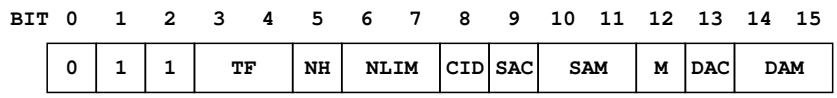
We believe that 6LoWPAN enabled IP communication is also possible in *industrial* WSN. Standardization bodies also realize the need and advantages of IP in industrial WSN. The proposed standard ISA 100.11a [5] is IP based, though not taking advantage of 6LoWPAN compression and IP security. IP is also introduced in WirelessHART's current specifications and formally named HART IP. The ZigBee alliance also proposed an IP solution for ZigBee and named it ZigBee IP [4]. Dust Network has already launched an $SmartMesh^{®}$ IP WSN evaluation kits that uses 6LoWPAN [8]. Also, CISCO and Emerson are jointly providing solutions to integrate an industrial WSN with Plants switched ethernet and IP network [9]; they propose to use 6LoWPAN.

In our solution for secure IP communication in WSN we make use of following standardized technologies.

2.3.1 IPv6

With the vision of all-IP networks all kind of physical devices such as wireless sensors are expected to be connected to the Internet [10]. These surely include industrial WSNs as well. This requires the use of IPv6 [11], a new version of the Internet Protocol that increases the address size from 32 bits to 128 bits. Besides the increased address space IPv6 provides in comparison to IPv4 a simplified header format, improved support for extensions and options, flow labeling capability and authentication and privacy capabilities.

[2] Maximum MTU size is 127 bytes. IP header only consumes 48 bytes.



TF: Traffic Class
NH: Next Header
HLIM: Hop Limit
CID: Context Identifier
SAC: Source Address Compression
SAM: Source Address Mode
M: Multicast Compression
DAC: Destination Address Compression
DAM: Destination Address Mode

(a) LOWPAN_IPHC encodings for basic IP header

(b) General format of LOWPAN_NHC encodings for the Next Header

Figure 2.2: 6LoWPAN Context-aware Compression Mechanisms

2.3.2 6LoWPAN

IPv6 over Low-power Personal Area Network (6LoWPAN) [12] is used to tightly interconnect the existing Internet and WSNs by specifying how IPv6 packets are to be transmitted over an IEEE 802.15.4 network. 6LoWPAN is an enabling technology towards the *Internet of Things* [13]. 6LoWPAN acts as a layer between the IP-layer and the link-layer that compresses IP and transport protocol headers and performs fragmentation when necessary. The maximum physical layer frame size of 802.15.4 packets is 127 bytes. If 802.15.4 security is enabled the maximum payload is reduced to 81 bytes. The IPv6 header alone would consume 40 bytes of the available 81 bytes. It is obvious that header compression mechanisms are an essential requirement of 6LoWPAN. When data cannot fit in a single packet 6LoWPAN performs fragmentation.

HC15[14] proposes context aware header compression mechanisms: the LOWPAN_IPHC (referred to as IPHC in the following) encoding for IPv6 header compression and Next Header Compression (NHC) for next header encoding that include IP extension headers and UDP.

For efficient IPv6 header compression, IPHC removes safely IPv6 header fields that are implicitly known to all nodes in the 6LoWPAN network. The

IPHC has a length of 2 byte of which 13 bits are used for header compression. Uncompressed IPv6 header fields follow directly the IPHC encoding in the same order as they would appear in the normal IPv6 header. In a multihop scenario IPHC can compress the IPv6 header to 7 bytes. The NH field in the IPHC indicates whether the next header following the basic IPv6 header is encoded. If NH is 1, NHC is used to compress the next header. 6LoWPAN specifies that the size of NHC should be multiple of octets, usually 1 byte where the first variable length bits represents a NHC ID and the remaining bits are used to encode/compress headers. 6LoWPAN already defines NHC for UDP and IP Extension Header [15]. The IPHC header is shown in Figure 2.2a. The NH field in the Figure 2.2a when set to 1 indicates that the next header following the compressed IPv6 header is NHC encoded. The general format of NHC is shown in Figure 2.2b.

2.4 Security in Wireless Sensor Networks

Due to the resource constrained nature of WSN providing secure communication is a challenge. There is always a trade off between security and performance. Unlike industrial WSN protocols such as WirelessHART where there is an always available central manager, typical WSNs protocols do not assume such a stable central entity. Based on this assumption many security architectures are proposed for WSNs such as SPINS [16], MiniSec [17], TinySec [18], TinySA [19]. Key management is one of the important problems in the WSN security. Many probabilistic [20, 21, 22, 23], deterministic [24, 25, 26], and hybrid [27, 28] key management schemes exist. Yang et al. [29] and more recently Simplício et al. [30] give very comprehensive overviews of these schemes. Though many schemes have been proposed none of them really seems to work for different WSN applications. Using pre-shared keys is still a state-of-art in WSN and security is still considered as an unsolved area in WSN.

The central manager in industrial WSNs such as the WirelessHART Network Manager can act as a key distribution center (KDC). This simplifies key management. However, the lack of Public Key Infrastructure (PKI) still makes the secure key management job difficult. This is particularly true for WirelessHART networks. Unlike typical WSN where security is usually ignored in real deployments security in industrial WSNs is necessary as the breach of security may produce catastrophic results. Security is built-in in all proposed standard protocols for industrial WSNs discussed in Section 2.2. In this thesis we mainly target security in WirelessHART.

2.5 WirelessHART Security

WirelessHART is a secure and reliable protocol for industrial automation. The field devices collect data about processes and securely send it, as an input, to other field devices. The routing information, security keys, and the timing information are sent to the devices in a secure way. In short, all data in a WirelessHART network travel in the form of WirelessHART commands and the confidentiality, integrity, and the authenticity of the commands are ensured. We can divide the provided security in the WirelessHART standard into three levels [3]: End-to-End, Per-hop, and Peer-to-Peer.

2.5.1 End-to-End Security

End-to-end security is enforced to secure the communication between the source and destination from malevolent insiders.The network layer is used to provide end-to-end security; any data that is passed from the network layer to the data-link layer is enciphered (except for the NPDU header) and only the destination device is able to decipher it. All field devices in the WirelessHART network have unicast and broadcast sessions with the Gateway and NM. Two field devices always communicate via the Gateway. It is possible to create peer-to-peer sessions between two field devices but the WirelessHART standard prohibits such direct connections due to security reasons.

The WirelessHART Network Protocol Data Unit (NPDU) is shown in the Figure 2.3. The NPDU payload is a Transport Layer PDU (TPDU) that is always encrypted using the Advanced Encryption Standard (AES) with a 128 bit key. The AES in CCM mode is used for calculating the MIC to provide authentication and data integrity, and encrypting the NPDU payload to provide confidentiality. The same key is used for both encryption and MIC calculation. The CCM mode is the combination of *Cipher Block Chaining-Message Authentication Code* (CBC-MAC) and *Counter* modes [31].

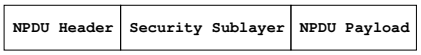


Figure 2.3: WirelessHART Network Layer PDU

For enabling security WirelessHART defines a security sub-layer beneath

[3]This is one of our contribution to specify security specifications in a clear way. The WirelessHART standard is very vague in explaining these services

network layer that comprises of Message Integrity Code (MIC), the Counter, and the Security Control Byte.

The network layer in the WirelessHART protocol stack provides three security services: confidentiality, integrity, and authentication. The type of key used at the network layer depends on the type of message; these keys are discussed in Section 3.2.2.

2.5.2 Per-Hop Security

Per-hop security is a defense against outsiders, i.e. devices that are not part of the network. The Data-Link Layer (DLL) is used to provide per-hop security between two neighboring wireless devices using the network key that is known to all authenticated devices in the WirelessHART network. Using the network key in the AES-CCM mode a keyed MIC is calculated on the entire Data Link-layer PDU (DLPDU). The MIC at the DDL ensures source integrity (authentication) of the messages between the two neighboring devices. The DLL also offers data integrity using Cyclic Redundancy Check (CRC) [4] .

2.5.3 Peer-to-Peer Security

All traffic in a WirelessHART network flows through the gateway, but a handheld device can create a direct one-to-one session with the field devices using the handheld key. In order to establish such connections, the handheld device first joins the WirelessHART network using its Join key; after successful joining, the handheld device requests the handheld key from the NM. The received handheld key is used to create a peer-to-peer session with the field device that also receives handheld key.

2.6 Security in IP-connected Industrial WSNs

The introduction of IP in ISA 100.11a and proposals for HART IP and ZigBee IP shows that industry wants IP in industrial WSNs. However, all these protocols rely on different security solutions. In this section we highlight security in IP enabled industrial WSNs.

[4] CRC is not a cryptographic way to enforce integrity; rather it is a way to check communication errors.

2.6.1 ISA 100.11a Security

Unlike WirelessHART, ISA 100.11a takes advantage of both symmetric and asymmetric security solutions. ISA 100.11a enables the use of PKI during device join operations and for key management. ISA 100.11a employs security at the link layer, transport layer, and application layer. The link layer security is borrowed from the IEEE 802.15.4 standard that secures communication between two neighboring devices. In addition to authentication and integrity services ISA 100.11a also provides confidentiality service at the link layer. The transport layer security or the UDP level security in ISA 100.11a ensures secure end-to-end communication. ISA 100.11a makes use of a Management Object (MO) at the application layer to provide secure key management. The standard restricts the device's MO communication with the ISA 100.11a security manager only. ISA 100.11a is not approved yet and in this thesis we do not focus on it. We intend to investigate ISA 100.11a security in future.

2.6.2 ZigBee IP Security

ZigBee IP is still in its inception stage. It proposes the use of state-of-start security mechanisms and develops a fully unified protocol stack. ZigBeeIP uses IEEE 802.15.4-2006 MAC/Physical layer, IETF 6LoWPAN layer for the header compression and neighbor discovery, IPv6 at the network layer, TCP/UDP at the transport layer, etc. For security processing, ZigBee IP proposes to use the Protocol for Authentication and Network Access (PANA) [32], Extensible Authentication Protocol (EAP) [33], EAP-TTLSv0 [34], EAP-TLS [35], and TLS [36]. All these security mechanisms provide strong security but ZigBee ignores the resource constrained nature of WSNs while proposing these protocols. All these protocols cannot take advantage of 6LoWPAN compression as 6LoWPAN currently does not provide compression mechanisms for the layers above UDP.

2.6.3 HART IP Security

HART IP is recently proposed but its current specifications use IP only in the wired part of the network. IPsec or Transport Layer Security (TLS), in current standardized forms, are the obvious choices to secure communication in HART IP as the wired part of the WirelessHART network has no resource scarce devices. However, the intended HART IP extension in the wireless part requires the use of lightweight versions of security protocols preferably IPsec. We plan

to apply IPsec in the intended WirelessHART IP protocol.

2.6.4 Security in 6LoWPANs

We believe that industrial WSN protocols, discussed in this chapter, can operate over IP. In IPv6, IPsec is mandatory meaning that every IPv6 enabled device must be able to process IPsec. Currently 6LoWPAN specifications do not specify the use of security protocols such as IPsec in conjunction with 6LoWPAN compression and fragmentation. We propose a lightweight and 6LoWPAN compressed version of IPsec for future industrial WSNs. This is one of our core contributions in this thesis that we highlight in Section 3.4. Before presenting our solutions for IP-connected WSNs based on IPsec we provide a comprehensive background of IPsec and current state-of-the-art security solution in 6LoWPANs.

2.6.5 IPsec

IPsec defines a set of protocols for securing IP communication: the security protocols Authentication Header (AH) [37] and Encapsulating Security Payload (ESP) [38], the algorithms for authentication and encryption, key exchange mechanisms and so called security associations (SA) [39]. An SA specifies how a particular IP flow should be treated in terms of security. To establish SAs, the IPSec standard specifies both pre-shared keys and the Internet Key Exchange (IKE) protocol. This means that every node on an IPv6 enabled conventional Internet supports pre-shared keys. In other words an implementation with pre-shared based SA establishment works with any IPv6 node on Internet. Also, IKE uses asymmetric cryptography that is assumed to be heavy weight for small sensor nodes. However, it would be worth investigating IKE with ECC for 6LoWPANs; we intend to do this in the future.

The task of the AH is to provide connectionless integrity and data origin authentication for IP datagrams and protection against replays. A keyed Message Authentication Code (MAC) is used to produce authentication data. The MAC is applied to the IP header, AH header, and IP payload.

ESP [38] provides origin authenticity, integrity, and confidentiality protection of IP packets. ESP is used to encrypt the payload of an IP packet but in contrast to AH it does not secure the IP header. If ESP is applied the IP header is followed by the ESP IP extension header which contains the encrypted payload. ESP includes an SPI that identifies the SA used, a sequence number to prevent replay attacks, the encrypted payload, padding which may be required

by some block ciphers, a reference to the next header and optional authentication data. Encryption in ESP includes Payload Data, Padding, Pad Length and Next Header;Authentication, if selected, includes all header fields in the ESP.

The protocols AH and ESP support two different modes: transport mode and tunnel mode. In transport mode IP header and payload are directly secured as previously described. In tunnel mode, a new IP header is placed in front of the original IP packet and security functions are applied to the encapsulated (tunneled) IP packet. In the context of 6LoWPAN tunnel mode seems not practical as the additional headers would further increase the packet size.

2.6.6 IEEE 802.15.4 Security

Currently, 6LoWPAN relies on 802.15.4 [40] security to protect the communication between neighboring nodes that are one hop apart. The security modes supported by the 802.15.4 standard include AES-CTR for encryption only, AES-CBC-MAC for message authentication only and AES-CCM which combines encryption and message authentication. For the MAC-modes the included authentication code is either 4, 8 or 16 byte. AES-CCM is the only mode mandated by the standard, which must be available on all standard compliant devices.The IEEE 802.15.4 standard currently uses pre-shared keys for encryption and integrity verification.

Chapter 3

Our Security Solutions for Industrial WSNs

In this chapter we present our solutions for the secure wireless communication in industrial WSNs. Our solutions comprise our work in both non IP WSNs and IP-connected WSN. We only present the core contributions and the related details can be found in Chapter 6, 7, 8. Before delving deeper in our core contributions presented in Section 3.2, 3.3, 3.4 we give an overview of our research method in the following section.

3.1 Research Method

In this thesis we adapt a research methodology that is hybrid of analytical and experimental research. *Analytical* research mainly deals with the testing of a concept that is not yet verified and specifying and inferring relationships by examining the concepts and information already available. *Experimental* research that often starts with a concrete problem is designed to evaluate the impact of one peculiar variable of a phenomena by keeping the other variables controlled. We apply the analytical research methodology to investigate the security issues in WirelessHART. Based on the knowledge we gain after literature review we perform a threat analysis of the WirelessHART network. We use the already known concepts about WirelessHART and facts about security threats in the wireless medium and examine how the provided security mechanisms in WirelessHART guard against these threats. This research results in

a clear understanding of the strengths and weaknesses in the WirelessHART security. We then provides the mechanisms to clarify, mitigate, or overcome the shortcomings in the WirelessHART security.

Most of the shortcomings that we identify in our analytical or investigative research are subjected to the unavailability of proposed but unspecified Security Manager (SM). We develop first SM for WirelessHART where we mainly adapt experimental research methodology as we have a concrete problem to solve. In order to build a SM we first develop hypotheses or ideas about the architecture of the security manager. We then formate a formal design based on our hypothesis. In order to validate our hypothesis we implement and evaluate the SM. In order to examine the impact of our designed and implemented SM on the WirelessHART network we perform the evaluation of the SM in a controlled setup where we only perform interaction with the network manager.

Realizing the need for the secure IP communication in WSN we develop the first lightweight compressed IPsec for WSNs as we find security loopholes in the current security solutions for the IP-connected WSN, see Section 8.3. The research method we adapt here is experimental too. The first step towards solving this problem is to formulate a hypothesis i.e. whether an IPsec is a feasible choice for the IP-connected WSN requirements. The next step is to develop a design of IPsec that suits WSNs. To this end we provide compressed version of IPsec based on 6LoWPAN specifications. To validate our hypothesis and design we implement IPsec and perform extensive experiments. In the next step we analyze our experimental results that clearly shows that IPsec is a feasible, practical, and a secure way of performing communication between a node in a 6LoWPAN and a device on conventional Internet.

3.2 WirelessHART Security Analysis

The WirelessHART security specifications are spread throughout the standard and it is hard to understand the provided security without studying the full WirelessHART standard. One of our contributions is to comprehend the security specifications in WirelessHART. We highlight some of core contributions here; details can be found in Chapter 6.

3.2.1 Threat Analysis

One of the core contributions of our work on the WirelessHART security is the formal threat analysis of the standard. After a detailed study of the Wire-

lessHART standard and the WSN security we perform threat analysis of the WirelessHART network. We analyze the WirelessHART network against interference, jamming, sybil attack [41], de-synchronization, traffic analysis, DOS attacks, wormhole [42], tampering, eavesdropping, selective forwarding attack, exhaustion, spoofing, and collision. Our analysis shows that the WirelessHART standard is secure enough to provide defense against most of the attacks. However, wormhole, de-synchronization, jamming, traffic analysis, spoofing, and exhaustion attacks need more attention.

Also, the physical protection of the WirelessHART devices is very important. If the device is captured by an attacker it should self destruct because otherwise it can be cloned and the secret contents can be revealed.

3.2.2 Security Keys in WirelessHART

WirelessHART is a complex standard and uses multiple security keys. However, the standard is very vague in specifying the security keys used and the functions of each key. One of our important contribution is to provide a comprehensive list of all security keys used in WirelessHART. These keys are explained in Chapter 6.

3.2.3 WirelesHART Security Limitations

After careful analysis of the WirelessHART standard we have found the following major security limitations:

- Secure multicast communication among the field devices is not supported.

- No security mechanisms are provide for physical security of the device and protection of data inside the device.

- WirelessHART does not support PKI. Hence security services such as non-repudiation are not supported.

- No mechanisms have been specified to provide authorization and accounting security services. We need accounting when the cost of a WirelessHART device is attached to its usage.

- The complete key management system is not specified; however, the commands for distribution of keys have been specified.

- Security in the wired part of the network is neither specified nor enforced.

- Secure integration of WirelessHART with the legacy HART is not specified in the standard.

- The specification, design, and architecture of the SM and the interface between the SM and the Network Manager is not specified in the standard.

3.3 WirelessHART Security Manager

One of the shortcomings we have found after the WirelessHART security analysis is the lack of WirelessHART SM specifications and design. In this section we highlight design, implementation, and evaluation of WirelessHART SM; details of these can be found in Chapter 7.

3.3.1 Design

Our SM provides key management system for WirelessHART networks, authentication of wireless devices, and security mechanisms to interconnect wired devices in the WirelessHART network. Our design consists of a *certification authority* (CA) and a *Key Manager*. The CA part secures the wired part of the WirelessHART network using PKI. The key manager provides a complete key management system for the wireless part of the WirelessHART network. The key manager includes key initialization, key renewal, key revocation, secure generation of keys, and key storage architecture. We also elaborate the protocol steps performed between SM and network manager for security processing; the WirelessHART standard does not provide these interactions.

3.3.2 Implementation and Evaluation

We implement our SM using the Metro web services architecture [43] that makes it possible to work with network managers developed in any language and for any platform. Our implementation consists of a CA and a key manager. We also evaluate our SM in term of average response time while interacting with network manager. We achieve an average response time of $71ms$ which

is far less than different reply time requirements in the WirelessHART standard such as maxReplyTime (30s), JoinReplyTimeout (default is Keep-Alive), BcastReplyTime (60s), etc. [44].

Our SM is complete in all aspects expect secure key renewal. This is because WirelessHART does not support PKI. However, we provide secure key renewal for WirelessHART networks in a separate publication [45].

3.4 Compressed IPsec for IP-connected WSN

The third main contribution of this thesis is the design, implementation, and detailed evaluation of IPsec. In this section we highlight some of the core components of this work.

3.4.1 6LoWPAN Extension for IPsec

IPsec requires header compression to keep packet sizes reasonable in 6LoWPAN. Unfortunately, there are no header encodings specified for AH and ESP extension headers. We highlight the hooks in the 6LoWPAN specification where we can link our proposed IPsec extensions.

LOWPAN_NHC Extension Header Encoding

As discussed in the background section the 6LoWPAN draft defines the general format of NHC that can be used to encode IP next header. We define NHC encodings for the two IP extension headers namely AH and ESP. 6LoWPAN already defines NHC encodings for IP extension headers (NHC_EH) that can be used to link AH and ESP extension headers. NHC_EH consist of a NHC octet where three bits (bits 4,5,6) are used to encode the IPv6 Extension Header ID (EID). Out of eight possible values for the EID, six are specified by the HC15 draft. The remaining two slots (101 and 110) are currently reserved. We propose to use the two free slots to encode AH and ESP. We present 6LoWPAN encodings LOWPAN_NHC_AH and LOWPAN_NHC_ESP for the AH and ESP in Chapter 8. Figure 3.1 shows a compressed IPv6/UDP packet secured with AH.

3.4.2 Implementation

We implement IPsec AH and ESP for the Contiki operating system [46]. The implementation required the modification of the existing Contiki μIP stack that

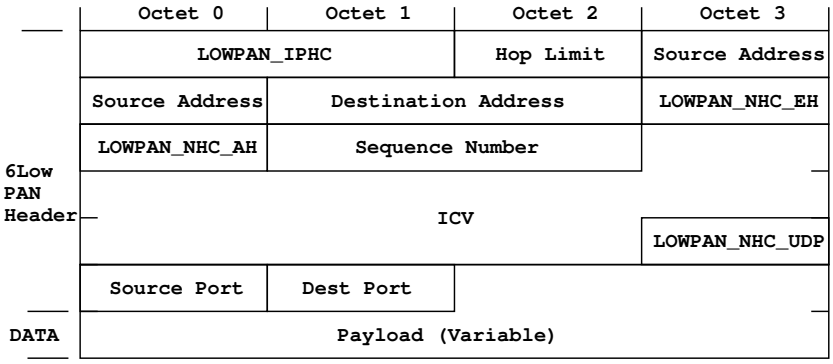


Figure 3.1: Example of a compressed IPv6/UDP packet using AH

already provides 6LoWPAN functionality. The Contiki μIP stack is used on the sensor nodes and on a so called soft bridge connecting WSN and the Internet. In addition to the IPsec protocol, we implement the IPsec/6LoWPAN compression mechanisms as outlined in the previous section. We support the NHC_EH, NHC_AH, and NHC_ESP encodings at the SICSLoWPAN layer, the 6LoWPAN component of the μIP stack.

We use the SHA1 and AES implementations from MIRACL [47], an open source library, and implement all cryptographic modes of operation needed for authentication and encryption in IPsec. For AH, we implement the mandatory HMAC-SHA1-96 and AES-XCBC-MAC-96. For ESP, we implement the mandatory AES-CBC for encryption and HMAC-SHA1-96 for authentication. Additionally, in ESP, we implement the optional AES-CTR for encryption and AES-XCBC-MAC-96 for authentication.

3.4.3 Evaluation

We evaluate the impact of IPsec in terms of memory footprint, packet size, performance of cryptography, energy consumption, and system-wide response time under different configurations. Our evaluation setup consists of four Tmote Sky [48] sensor nodes, a 6LoWPAN soft bridge (implemented by a fifth Tmote), and a Linux machine running Ubuntu OS with IPsec enabled.

Our evaluation results in Chapter 8 show that the IPsec AH and ESP fit in a tiny sensor node (e.g. Tmote Sky) with still room available for applications.

Our cryptographic algorithms analysis show that our implementations for AES-CBC and AES-XBC-MAC-96 – the IPsec standard recommended algorithms for future Internet – are faster in terms of processing time and efficient regarding energy consumption and could be definitely used in 6LoWPAN realm. The energy overhead involved is not significant when compared to the consumption of typical radio chips. The system-wide response time comparisons with and without IPsec show that the slowdown of IPsec is acceptable; hundreds of milliseconds for 512 bytes of data. Moreover, we show that these overheads can be significantly reduced with the help of hardware encryption.

3.5 Standardization of Proposed Solutions

Our contributions presented in this chapter mainly target two standards: HCF WirelessHART and IETF 6LoWPAN. During this thesis period I attended meetings of both the standardization Working Groups. This helped me a lot to know the current status of the standardization efforts and make people aware of our work. I have attended the WirelessHART Working Group meeting in Florence, the Internet Architecture Board (IAB) official workshop and tutorial in Prague, and the IETF 80th meeting. I tried to make people understand our current work in both the standards and raise the importance of security in industrial WSNs. The ultimate aim is the inclusion of these solutions in the standard specifications. I have also published my IPsec work in an IAB workshop [6].

Chapter 4

Conclusions

In this thesis we study the security issues in industrial WSNs. We have also highlighted the increasing interest in IP communication in industrial WSN and how these networks can be secured using our solutions. Seeing that the IPsec is mandatory in IPv6 and our evaluations showed very encouraging results we are confident that our lightweight IPsec solution will be a plausible choice for securing the 6LoWPAN enabled WSNs or so called the Internet of Things. In this chapter we give the summary of the work we have done throughout this thesis. We end this chapter with the future work that we intend to do.

4.1 Summary

We have discussed the security features in the WirelessHART standard and analyzed the specified security features against the available threats in the wireless medium. We have also identified some security limitations in the standard. However, the provided security in the wireless medium, although subjected to some threats due to its wireless nature, is strong enough to be used in the industrial process control environment. The physical protection of the WirelessHART devices is very important to avoid device cloning and stealing security secrets which can lead to other security attacks.

To overcome most of the limitations we identified in our previous work, we have developed a Security Manager (SM) for WirelessHART. We have elucidated comprehensive specifications of the SM. We have converted the specifications into an architectural design that models both the internals of the SM

and its interaction with the other network devices. We have developed a SM that can interact with other network devices developed for different platforms and with different programming languages.

Understanding the trend that WSNs will be an integral part of Internet we have proposed IPsec for such networks. IPv6 and 6LoWPAN are the protocol standards that are expected to be used in this context. IPsec is the standard method to secure Internet communication and we investigate if IPsec can be extended to sensor networks. Towards this end, we have presented the first IPsec specification and implementation for 6LoWPAN. We have extensively evaluated our implementation and demonstrated that it is feasible and practical to use compressed IPsec to secure communication in IP-connected WSNs.

4.2 Future Work

In the future we plan to work further on IPsec and its applicability in industrial WSNs. In our current IPsec implementation we rely on pre-shared keys. We are working on lightweight IKEv2 that is a de facto automatic key exchange protocol for IPsec.

We intend to apply our full fledged IPsec-IKE solution in securing industrial WSNs particularly HART IP and ISA 100.11a and try to make them interoperable at the network layer. We plan to deploy and test it in a real setup.

Last but not the least we plan to deploy and evaluate our IPsec solution in a real deployment. To this end we target building automation and its interconnection with a smart metering system.

Chapter 5

Overview of Papers

5.1 Paper A

Security Considerations for the WirelessHART Protocol. **Shahid Raza**, Adriaan Slabbert, Thiemo Voigt, Krister Landernäs. In 14th IEEE International Conference on Emerging Techonologies and Factory (ETFA'09), September 2009, Mallorca, Spain.

Summary WirelessHART is a secure and reliable communication standard for industrial process automation. The WirelessHART specifications are well organized in all aspects except security: there are no separate specifications of security requirements or features. Rather, security mechanisms are described throughout the documentation. This impedes implementation of the standard and development of applications since it requires profound knowledge of all the core specifications on the part of the developer. In this paper we provide a comprehensive overview of WirelessHART security: we analyze the provided security mechanisms against well known threats in the wireless medium, and propose recommendations to mitigate shortcomings. Furthermore, we elucidate the specifications of the Security Manager, its placement in the network, and its interaction with the Network Manager.

My contribution Thiemo Voigt suggested the study area of this paper. The basic idea of this paper was suggested by me and I was the main analyst and driver in conducting research, writing, and finalization of the paper.

5.2 Paper B

Design and Implementation of a Security Manager for WirelessHART Networks. **Shahid Raza**, Thiemo Voigt, Adriaan Slabbert, Krister Landernäs. In 5th IEEE International Workshop on Wireless and Sensor Networks Security (WSN'S 2009), in conjunction with MASS'2009, 12-15 Oct 2009, Macau SAR, P.R.C..

Summary WirelessHART is the first open standard for Wireless Sensor Networks designed specifically for industrial process automation and control systems. WirelessHART is a secure protocol; however, it relies on a Security Manager for the management of the security keys and the authentication of new devices. The WirelessHART standard does not provide the specification and design of the Security Manager. Also, the security specifications in the standard are not well organized and are dispersed throughout the standard which makes an implementation of the standard more difficult. In this paper we provide the detailed specification and design as well as an implementation of the Security Manager for the WirelessHART standard. We evaluate our Security Manager against different cryptographic algorithms and measure the latency between the Network Manager and the Security Manager. Our evaluation shows that the proposed Security Manager meets the WirelessHART requirements. Our analysis shows that the provided Security Manager is capable of securing both the wireless and wired part of the WirelessHART network.

My contribution The idea of this paper was suggested by me. I was the main designer of security manager. I am the main driver in writing the paper and was responsible for implementation and evaluation of the security manager proposed in the paper.

5.3 Paper C

Securing Communication in 6LoWPAN with Compressed IPsec. **Shahid Raza**, Simon Duquennoy, Tony Chung, Dogan Yazar, Thiemo Voigt, Utz Roedig. In 7th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS '11), 27-29 June 2011, Barcelona, Spain. (To Appear)

Summary Real-world deployments of wireless sensor networks (WSNs) require secure communication. It is important that a receiver is able to verify

that sensor data was generated by trusted nodes. It may also be necessary to encrypt sensor data in transit. Recently, WSNs and traditional IP networks are more tightly integrated using IPv6 and 6LoWPAN. Available IPv6 protocol stacks can use IPsec to secure data exchange. Thus, it is desirable to extend 6LoWPAN such that IPsec communication with IPv6 nodes is possible. It is beneficial to use IPsec because the existing end-points on the Internet do not need to be modified to communicate securely with the WSN. Moreover, using IPsec, true end-to-end security is implemented and the need for a trustworthy gateway is removed. In this paper we provide End-to-End (E2E) secure communication between IP enabled sensor networks and the traditional Internet. This is the first compressed lightweight design, implementation, and evaluation of 6LoWPAN extension for IPsec. Our extension supports both IPsec's Authentication Header (AH) and Encapsulation Security Payload (ESP). Thus, communication endpoints are able to authenticate, encrypt and check the integrity of messages using standardized and established IPv6 mechanisms.

My contribution I have designed the compression schemes for IPsec and linked it with existing 6LoWPAN specifications. I have implemented the security algorithms for IPsec and contributed in implementing IPsec AH and ESP support in the Contiki uIP and SICSLoWPAN stacks. I am the the main driver in writing the paper and I designed and conducted most of the evaluation.

Bibliography

[1] Anna N. Kim, Fredrik Hekland, Stig Petersen, and Paula Doyle. When hart goes wireless: Understanding and implementing the wirelesshart standard. *IEEE International Conference on Emerging Technologies and Factory Automation*, pages 899–907, September 2008.

[2] N. Kushalnagar, G. Montenegro, and C. Schumacher. IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals. RFC 4919, August 2007.

[3] *WirelessHART Specifications*. HART Communication Foundation, May 2008. http://www.hartcomm2.org.

[4] Don Sturek. *ZigBee IP Stack Overview*. http://www.zigbee.org/imwp/idms/popups/pop_download.asp?contentID=16560.

[5] ISA. *ISA-100.11a-2009: Wireless systems for industrial automation: Process control and related applications*. ISA Standards, 67 Alexander Drive Research Triangle Park, NC 27709 USA, May 2009.

[6] Shahid Raza, Gianluca Dini, Thiemo Voigt, and Mikael Gidlund. Secure key renewal in wirelesshart. *Real-time Wireless for Industrial Applications (RealWin'11) - CPS Week*, April 2011.

[7] Tomas Lennvall, Stefan Svensson, and Fredrik Heklan. A comparison of wirelesshart and zigbee for industrial applications. *IEEE International Workshop on Factory Communication Systems*, pages 85–88, May 2008.

[8] Meg Godfrey. Dust Networks Expands WSN Product Line with 6LoWPAN Offering. Technical report, 2009.

[9] CISCO White Paper C11-503160-01. Integrating an Industrial Wireless Sensor Network with Your Plants Switched Ethernet and IP Network. Technical report, 2009.

[10] J. Vasseur and A. Dunkels. *Interconnecting Smart Objects with IP - The Next Internet*. Morgan Kaufmann, 2010.

[11] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. RFC 2460, December 1998.

[12] G. Deloche, N. Kushalnagar, J. Hui, and D. Culler. Transmission of IPv6 Packets over IEEE 802.15.4 Networks. RFC 4944, September 2007.

[13] N. Gershenfeld, R. Krikorian, and D. Cohen. The internet of things. *Scientific American*, 291(4):76–81, 2004.

[14] J. Hui and P. Thubert. Compression Format for IPv6 Datagrams in Low Power and Lossy Networks. draft-ietf-6lowpan-hc-15, 2010.

[15] J. Hui and P. Thubert. Compression Format for IPv6 Datagrams in 6LoWPAN Networks. draft-ietf-6lowpan-hc-13, September 2010.

[16] A. Perrig, R. Szewczyk, JD Tygar, V. Wen, and D.E. Culler. Spins: Security protocols for sensor networks. *Wireless networks*, 8(5):521–534, 2002.

[17] M. Luk, G. Mezzour, A. Perrig, and V. Gligor. Minisec: a secure sensor network communication architecture. In *Proceedings of the 6th international conference on Information processing in sensor networks*, pages 479–488. ACM, 2007.

[18] C. Karlof, N. Sastry, and D. Wagner. Tinysec: a link layer security architecture for wireless sensor networks. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 162–175. ACM, 2004.

[19] J. Grosschadl. Tinysa: A security architecture for wireless sensor networks. In *Proceedings of the 2006 ACM CoNEXT conference*, page 55. ACM, 2006.

[20] H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. 2003.

[21] L. Eschenauer and V.D. Gligor. A key-management scheme for distributed sensor networks. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, pages 41–47. ACM, 2002.

[22] S. Zhu, S. Xu, S. Setia, and S. Jajodia. Establishing pairwise keys for secure communication in ad hoc networks: A probabilistic approach. 2003.

[23] T. Ito, H. Ohta, N. Matsuda, and T. Yoneda. A key pre-distribution scheme for secure sensor networks using probability density function of node deployment. In *Proceedings of the 3rd ACM workshop on Security of ad hoc and sensor networks*, pages 69–75. ACM, 2005.

[24] J. Lee and D. Stinson. Deterministic key predistribution schemes for distributed sensor networks. In *Selected Areas in Cryptography*, pages 294–307. Springer, 2005.

[25] D.S. Sanchez and H. Baldus. A deterministic pairwise key predistribution scheme for mobile sensor networks. In *Security and Privacy for Emerging Areas in Communications Networks, 2005. SecureComm 2005. First International Conference on*, pages 277–288. IEEE, 2005.

[26] S.A. Camtepe and B. Yener. Combinatorial design of key distribution mechanisms for wireless sensor networks. *Computer Security–ESORICS 2004*, pages 293–308, 2004.

[27] J. Deng. A pairwise key pre-distribution scheme for wireless sensor networks. 2005.

[28] D. Liu, P. Ning, and R. Li. Establishing pairwise keys in distributed sensor networks. *ACM Transactions on Information and System Security (TISSEC)*, 8(1):41–77, 2005.

[29] Y. Xiao, V.K. Rayi, B. Sun, X. Du, F. Hu, and M. Galloway. A survey of key management schemes in wireless sensor networks. *Computer communications*, 30(11-12):2314–2341, 2007.

[30] M.A. Simplício Jr, P.S.L.M. Barreto, C.B. Margi, and T.C.M.B. Carvalho. A survey on key management mechanisms for distributed wireless sensor networks. *Computer Networks*, 54(15):2591–2612, 2010.

[31] Shahid Raza, Adriaan Slabbert, Thiemo Voigt, and Krister Landernas. Security considerations for the wirelesshart protocol. In *Proc. ETFA 2009*, September 2009.

[32] D. Forsberg, Y. Ohba, B. Patil, H. Tschofenig, and A. Yegin. Protocol for Carrying Authentication for Network Access (PANA). RFC 5191 (Proposed Standard), May 2008. Updated by RFC 5872.

[33] B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, and H. Levkowetz. Extensible Authentication Protocol (EAP). RFC 3748 (Proposed Standard), June 2004. Updated by RFC 5247.

[34] P. Funk and S. Blake-Wilson. Extensible Authentication Protocol Tunneled Transport Layer Security Authenticated Protocol Version 0 (EAP-TTLSv0). RFC 5281 (Informational), August 2008.

[35] D. Simon, B. Aboba, and R. Hurst. The EAP-TLS Authentication Protocol. RFC 5216 (Proposed Standard), March 2008.

[36] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Proposed Standard), August 2008. Updated by RFCs 5746, 5878, 6176.

[37] Stephen Kent. IP Authentication Header. RFC 4302, 2005.

[38] S. Kent. IP Encapsulating Security Payload. RFC 4303, 2005.

[39] S. Kent and K. Seo. Security architecture for the internet protocol. RFC 4301, 2005.

[40] IEEE Computer Society. Ieee std. 802.15.4-2006, 2006.

[41] John R. Douceur. The sybil attack. *1st International workshop on Peer-To-Peer Systems (IPTPS)*, March 2002.

[42] Levente Buttyan and Jean-Pierre Hubaux. *Security and Cooperation in Wireless Network*. Cambridge University Press, 2007.

[43] GlassFish Community. Metro Users Guide. Technical report, 2009.

[44] *Network Management Specification,HCF_SPEC-085, Revision 1.1*. HART Communication Foundation, May 2008.

[45] Shahid Raza, Gianluca Dini, Thiemo Voigt, and Mikael Gidlund. Secure key renewal in wirelesshart. In *Real-time Wireless for Industrial Applications (RealWin'11) - CPS Week*, April 2011.

[46] A. Dunkels, B. Grönvall, and T. Voigt. Contiki - a lightweight and flexible operating system for tiny networked sensors. In *EMNets'04*, Tampa, USA, November 2004.

[47] Shamus Software. Multiprecision Integer and Rational Arithmetic C/C++ Library. Web page. Visited 2010-04-17.

[48] J. Polastre, R. Szewczyk, and D. Culler. Telos: enabling ultra-low power wireless research. In *IPSN'05*, apr. 2005.

II

Included Papers

Chapter 6

Paper A: Security Considerations for the WirelessHART Protocol

Shahid Raza, Adriaan Slabbert, Thiemo Voigt, Krister Landernäs
In 14th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA'2009), 22 Sept 2009, Mallorca, Spain.

Abstract

WirelessHART is a secure and reliable communication standard for industrial process automation. The WirelessHART specifications are well organized in all aspects except security: there are no separate specifications of security requirements or features. Rather, security mechanisms are described throughout the documentation. This impedes implementation of the standard and development of applications since it requires profound knowledge of all the core specifications on the part of the developer.

In this paper we provide a comprehensive overview of WirelessHART security: we analyze the provided security mechanisms against well known threats in the wireless medium, and propose recommendations to mitigate shortcomings. Furthermore, we elucidate the specifications of the Security Manager, its placement in the network, and interaction with the Network Manager.

6.1 Introduction

WirelessHART [1] is the first IEC approved [2] open standard for Wireless Sensor Networks (WSNs) designed primarily for industrial process automation and control systems. The applications of WirelessHART include process and equipment monitoring, environment and energy monitoring, asset management, and advanced diagnostics. The WirelessHART network is a collection of wired entities: Network Manager, Gateway, Security Manager, and Plant Automation Hosts (PAH); and wireless devices: Field devices, Adapters, Routers, Access Points, and Handheld devices [3]. The Network Manager provides overall management, network initialization functions, network scheduling and monitoring, and resource management. The Network Manager collaborates with the Security Manager for the management and distribution of security keys. The wireless devices are connected using a mesh network where each device acts as a router and must be directly connected with at least two neighboring devices to provide path diversity. The protocol stack is based on a seven layer OSI stack with additional Security and MAC sub layers. WirelessHART is a self healing and self organizing wireless protocol, in that the devices are able to find neighbors and establish paths with them, and detect network outrages and reroute.

The WirelessHART standard is developed by the HART Communication Foundation (HCF) [4] consisting of authorities in process automation and control. The WirelessHART specifications are very well designed and almost complete in all aspects except security. The provided security is spread throughout the WirelessHART specifications and the standard lacks a comprehensive document that explains and specifies the security. The network designers and device vendors encounter ambiguities regarding the complete security architecture of the WirelessHART, the strength of the provided security, the security keys needed, and the functionalities and placement of Security Manager.

The WirelessHART standard has been recently released and we are the first to analyze and clarify its security features. Our main contribution is to provide a thorough understanding of the security features in WirelessHART. We discuss the strengths and weaknesses of the provided security mechanisms in the form of threat analysis: we analyze the WirelessHART security against the well known threats in the wireless medium and propose recommendations to mitigate the impact of these threats. We also explain the security keys and their usage as the standard does not illustrate them clearly. Finally, we elaborate the functions of the Security Manager, its placement in the network, and its interaction with the Network Manager.

6.2 WirelessHART Security

The legacy HART protocol (HART 6 and earlier) uses only single parity check coding schemes [5] to detect communication errors. However, WirelessHART (HART 7) is a secure and reliable protocol for industrial automation. The field devices collect data about processes and securely send it, as an input, to other field devices. The routing information, security keys, and the timing information are sent to the devices in a secure way. In short, all data in the WirelessHART network travel in the form of WirelessHART commands and the confidentiality, integrity, and the authenticity of the commands are ensured. We can divide the provided security in the WirelessHART standard into three levels: End-to-End, Per-hop, and Peer-to-Peer.

6.2.1 End-to-End Security

End-to-end security is enforced to secure the communication between the source and destination from malevolent insiders.The Network Layer is used to provide end-to-end security; any data that is passed from the network layer to the data-link layer is enciphered (except for the NPDU header) and only the destination device is able to decipher it. All field devices in the WirelessHART network have unicast and broadcast sessions with the Gateway and Network Manager. Two field devices always communicate via the Gateway [1]. The Network Protocol Data Unit (NPDU) is shown in the Table 6.1.

NPDU Header	Security Sublayer	NPDU Payload

Table 6.1: WirelessHART Network Layer PDU

The NPDU payload in Table 6.1 is a Transport Layer PDU (TPDU) that is always encrypted using the Advanced Encryption Standard (AES) with a 128 bit key. The Security Sub-layer consists of the Message Integrity Code (MIC), the Counter, and the Security Control Byte. The NPDU header is needed for routing of data; its details can be found in the specifications [6]. The three fields in the Security Sub-layer are used as follows:

i. Security Control Byte: It is used to define the type of the security employed. The first four bits are reserved for future security enhancement

[1] It is possible to create peer-to-peer session between the two field devices but the WirelessHART standard prohibits such direct connections due to security reasons.

and the next four bits define the security types. In HART 7.1, only three types are identified, see Figure 6.1 for details.

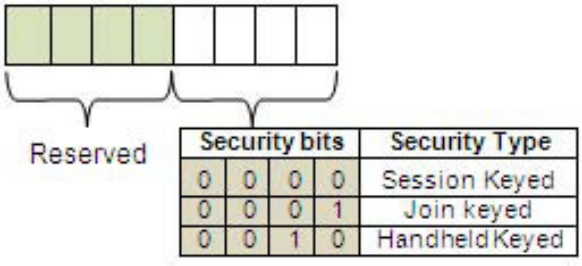


Figure 6.1: Security Control Byte

ii. Counter: A four-byte counter that is used to create the nonce.

iii. MIC: Keyed MIC is used for data integrity and source integrity (authentication) between source and destination. The MIC is calculated on the whole NPDU by setting the Time To Live (TTL), Counter, and MIC to zero. Four byte-strings are needed to calculate the MIC, including: NPDU header (a) - from control byte to MIC, NPDU payload (m) - the encrypted TPDU, *Nonce* - 13 byte long and provides defense against reply attacks, *AES key* - a 128 bit key needed for calculating the MIC. The same key is used for encrypting NPDU payload.

The Network Layer in the WirelessHART protocol stack provides three security services: confidentiality, integrity, and authentication. The AES in Counter with CBC-MAC (CCM) mode [7] is used for calculating the MIC to provide authentication and data integrity, and encrypting the NPDU payload to provide confidentiality. The same key is used for both encryption and MIC calculation. The CCM mode is the combination of *Cipher Block Chaining-Message Authentication Code* (CBC-MAC) and *Counter* modes. The two methods are highlighted below:

i. AES-CCM in CBC-MAC mode

In CBC-MAC, the message is enciphered using a block cipher algorithm in CBC mode and the last cipher block called MAC/MIC is constructed. In WirelessHART, the CBC-MAC mode is used to calculate the MIC at

the network and the data-link layers. CBC-MAC can be used for both plain text and cipher text. This mode needs the exact number of blocks and padding is used to equalize the last block. Only Encryption is used for calculating and verifying the MIC. A formatting function is applied on the unencrypted NPUD header, the encrypted NDPU payload, and the Nonce to produce the blocks B0, B1, B2...Bi; for details about this formatting function and block formation please refer to [8]. Figure 6.2 shows the operations to calculate MIC using CBC-MAC mode.

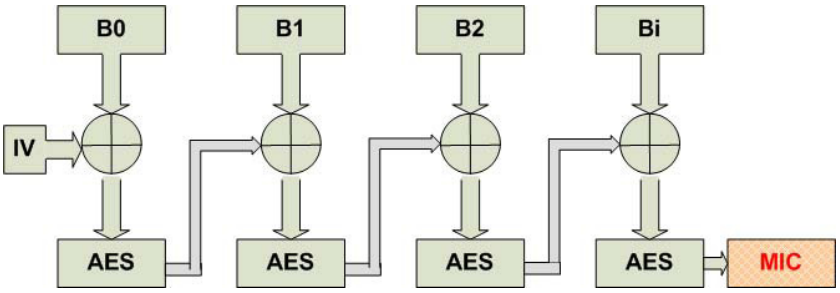


Figure 6.2: CBC-MAC mode for calculating MIC

ii. AES-CCM in Counter mode

The Counter mode is used for the encryption/decryption of the WirelessHART NPDU payload. Here too, the message blocks are created in the same fashion as above, but no padding is required and blocks can be manipulated in parallel. The cipher text C0, C1, C2,... will form an encrypted NPDU payload. The counter mode is shown in the Figure 6.3.

6.2.2 Per-Hop Security

The Data-Link Layer (DLL) is used to provide per-hop security between the two neighboring wireless devices using the Network key. Per-hop security is a defense against outsiders, i.e. devices that are not part of WirelessHART network. The Network key is known to all authenticated devices in the WirelessHART network. The keyed MIC is calculated on the entire Data Link-layer PDU (DLPDU) using the AES-CCM mode as discussed above. The four parameters for the AES CCM mode are:

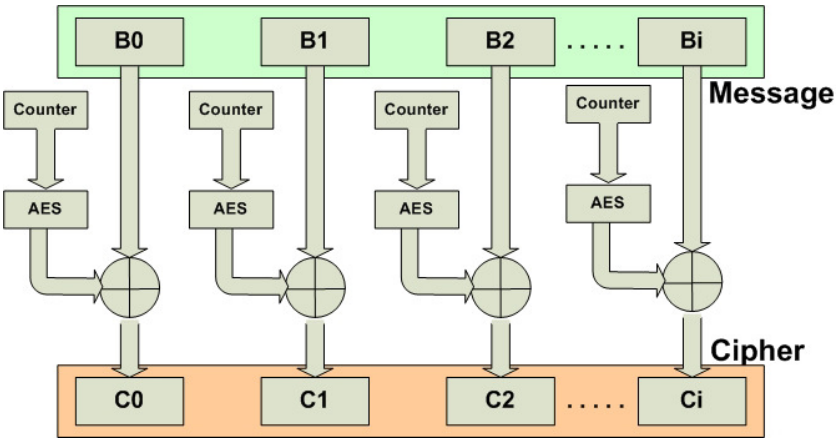


Figure 6.3: Counter mode for enciphering NPDU payload

- *m*: the encrypted message; but as the DLPDU is not encrypted the length of this byte-string is zero in WirelessHART.
- *a*: the DLPDU from 0x41 to DLPDU payload [9].
- *N*: a 13 bytes byte-string that is formed by concatenating the Absolute Slot Number (ASN) and source address [9].
- *K*: the 128 bit Network Key.

The DLL ensures source integrity (authentication) of the message between the two neighboring devices. The DLL also offers data integrity using Cyclic Redundancy Check (CRC) [2] . The WirelessHART standard uses the 16-bits ITU-T polynomial [10] to calculate the CRC.

6.2.3 Peer-to-Peer Security

All traffic in the WirelessHART network flows through the gateway, but a Handheld device can create a direct one-to-one session with the field devices using the Handheld key [3]. In order to establish such connections, the Handheld device first joins the WirelessHART network using its Join key; after

[2]CRC is not a cryptographic way to enforce integrity; rather it is a way to check communication errors.

successful joining, the Handheld device requests the Handheld key from the Network Manager. The received Handheld key is used to create a peer-to-peer session with the field device that also receives Handheld key.

Summary

The WirelessHART standard provides data confidentiality, data integrity, authentication (source integrity), and availability (using FHSS [11] and time slotting [6]) but the standard does not enforce authorization, non-repudiation, and accounting services.

6.3 Threat Analysis

A threat is an indication of a potential undesirable event [12]. The use of the wireless interface makes WirelessHART more vulnerable than legacy HART. We list possible threats against a WirelessHART network and discuss which threats are addressed by WirelessHART and which threats must be addressed. We propose recommendations to reduce the impact of the threat. The threat analysis will help developers, manufactures, and protocol designers to mitigate the impact of the threat in the design solutions.

6.3.1 Interference

Interference is an unintentional disruption of a radio signal; a signal with the same frequency and modulation technique can override the actual signal at the receiver. WirelessHART operates at the 2450 (2400-2483.5) MHz frequency band spectrum and has 16 channels; this spectrum can be shared with e.g. Wi-Fi, Bluetooth, WibRee (Bluetooth Low Energy Technology), ZigBee, and ISA100.11.a.

The WirelessHART standard uses Frequency-Hopping Spread Spectrum (FHSS) [11], uniquely assigned time slots using Time Division Multiple Access (TDMA), and path diversity which reduces the chances that interference causes actual harm to the operation of the network. With the reliability greater than 3-sigma (99.7300204%) [6] WirelessHART is the most reliable protocol among the current available solutions for industrial process automation especially if we compare it with ZigBee [13]. Nevertheless the strict and sensitive nature of a process automation system requires fail proof (100%) reliability

and failure may produce catastrophic results. The growing number of Wi-Fi, ZigBee, Bluetooth etc. devices can make the WirelessHART frequency band more vulnerable to interference in the future.

6.3.2 Jamming

Jamming is normally considered an intentional interruption of radio signal when purposely introducing noise or signal with same frequency and modulation technique as used in the target network. WirelessHART is more vulnerable to jamming attacks than interference; the attacker can deliberately introduce radio signals using commonly used Bluetooth devices like cell phones or laptops.

WirelessHART uses the concept of channel Blacklisting. If a certain frequency channel is jammed or is a continuous source of interference, then it can be blacklisted. Blacklisting enhances the reliability of the WirelessHART network but at the same time it limits the number of channels available to send/receive traffic. In spite of FHSS with 15 available channels, the active attacker can jam the WirelessHART network. The switching of channels in the FHSS is based on a pseudorandom sequence. Now if,

a. An attacker has knowledge of pseudorandom sequence (which is hard to find), he/she can calculate the actual channel. (ActualChannel = (ChannelOffset + ASN) % NumChannels) [9]

b. There are sufficient number of 2.4 GHz (Bluetooth, ZigBee, etc) devices in the range of the WirelessHART network

c. The manufacturing plant has legally deployed Wi-Fi networks in and around the WirelessHART network

d. The manufacturing plant produces sufficient amount of noise signals (which is very common there)

e. Some of the channels are already blacklisted,

then the active attacker can jam the WirelessHART network [14]. This jamming of the whole or a part of the WirelessHART network can block or even damage the machinery or plant assets.

6.3.3 Sybil

In a Sybil attack [15], an antagonist can hold multiple identities by introducing an adverse entity such as a node or piece of software into a network. The

lack of a trusted central authority in the traditional wireless ad hoc and sensor networks make it possible for the adversary to own multiple identities.

The Network Manager in the WirelessHART network binds an entity with a unique identity. The Network Manager assigns a unique Nickname to all the connected devices. Also, every device has a globally unique ID where the ID is a combination of Device Type and Device ID. The WirelessHART Gateway maintains the list of the Unique IDs and the Network Manager maintains the list of the Nicknames; the wireless devices use these Unique IDs and Nicknames along with the session keys to maintain sessions with the Gateway and Network Manager respectively. This makes Sybil attacks almost impossible in WirelessHART networks.

6.3.4 Traffic Analysis

The broadcast nature of the wireless signals make them more prone to the traffic analysis than wired signals where the attacker should be physically connected to the network.

In WirelessHART networks, the NPDU header and the whole DLPDU are unencrypted and the adversary can easily analyze the WirelessHART traffic. The NPDU header fields e.g. source/Destination addresses, Security Control byte, Nonce counter, etc. are all sent in clear. These fields provide enough information to the rival to perform analysis of the network: finding new devices by analyzing join requests, work peak hours, device usage that can help to make other attacks more effective etc.

If the DLPDU payload were allowed to be encrypted with the Network key (which is also used to calculate the MIC over DLPDU) then the traffic analysis could be minimized, but then all the intermediate devices have to decrypt the NPDU at the DLL to find the destination address, routing information, etc; this will make it difficult to meet the timing requirement of 10ms which is already hard as pointed out by Song [16]. This trade off between the security and system performance makes traffic analysis attack relatively easy.

6.3.5 DOS

Denial-of-Service (DOS) is a common attack on all networked systems; it is against the *Availability* security service. The wireless nature of WirelessHART makes it more prone to the DOS attack than legacy HART. DOS attacks against a WirelessHART network can be launched by:

- Flooding the network with join requests as the join message is encrypted with the Well-known key at the DLL.

- Sending the fake Advertisements to the neighbors (also encrypted with the Well-known key).

- Continuously modifying the DLPDU and re-computing the CRC: Now the receiving device has to verify the message integrity by calculating the MIC (as the CRC is verified); the WirelessHART protocol uses AES in CCM for calculating MIC which is an expensive operation and requires strict timing (TsTxAckDelay =1ms) requirements [16] to verify the MIC. The unverified packet will be discarded, which results in the retransmission of the packet and consumption of network resources.

- Launching a jamming attack (see section 3.2).

6.3.6 De-synchronization

The attacker can disrupt the communication between two nodes by introducing false timing information in the network and engaging the devices to waste their resources in time synchronization.

The WirelessHART standard has strict timing requirements, and the Timer [1] is one of the primary modules in the network. The Timer module has to meet the timing requirements and keep the time slots (10ms) in synchronization. The MAC sub-layer is responsible for time slotting. Each time a node receives an ACK from its time source, it adjusts its clock. The timing source for a node can be a sender [16], and if the sender is compromised it can disrupt the timing between the two nodes. Hence the participating nodes waste their resources in time synchronization.

6.3.7 Wormhole

In a wormhole attack [17] the adversary creates a tunnel between two legitimate devices by connecting them through the stronger wireless (by inaugurating radio transceivers at both ends) or wired links.

The potential WirelessHART devices that the attacker can use to launch wormhole attack are HART devices (wired) connected to WirelessHART network through the Adapters; the adversary can create a tunnel by connecting two field devices using their maintenance port. A tunnel can also be created by a wireless connection if the Network or Session keys are compromised.

WirelessHART can be subjected to wormhole attack if it uses graph routing (that supports redundant paths). However, if source routing is used then the device must use device-by-device route from source to destination. Source routing provides defense against wormhole attacks but is not reliable, since if any of the intermediate links fail a packet will be lost. One of the recommended solutions to prevent wormhole attack is packet leashing [18]. The physical protection of devices can avoid wired connected wormholes.

6.3.8 Tampering

Tampering or modification attack is the changing of stored secrets or data in transit. If the message is protected with CRC or hash, the attacker usually modifies the data and recalculates the hash or CRC. The stored secrets can be tampered by physically capturing the device and changing the data.

The WirelessHART standard uses the keyed MIC at the Network and Data-link layer to enforce integrity and provide defense against a data tampering attack. Without the knowledge of this specific key the attacker is unable to perform this attack. It is easier to perform a modification attack in the DLL than in the Network layer as the Network key is shared among all the devices and hence easy to find while session keys are device specific. Knowing the Network key and the unencrypted DLPDU, an adversary can seriously damage the normal operations of the WirelessHART network by tampering with the DLPDU and re-calculating the MIC to make it authentic.

Regular changing of the Network key is highly recommended. The physical protection of the device provides defense against the tampering of stored secrets.

6.3.9 Eavesdropping

Eavesdropping refers to the surreptitious listening of private communication. The *Confidentiality* security service is used to protect data from eavesdroppers.

The actual WirelessHART message consists of aggregated commands. These Commands, the Transport Byte, and the Device Status collectively form a NPDU payload that is encrypted with an AES 128 algorithm using unicast session key. Although some attacks [19] [20] have been identified against AES, none of them are able to crack it and AES is still a NIST USA recommended standard. For an attacker it is very hard to find a session key as it is short lived and unique for each device; hence message eavesdropping is difficult in the

WirelessHART network. Also, the use of FHSS does not allow the eavesdropper to intercept the signal without having the pseudorandom sequence [11].

6.3.10 Selective Forwarding Attack

Here the compromised node selectively drops packets; the worst form is when the node does not forward any packet and creates a black-hole [21], but normally the node selectively discards packets so that it is considered as legitimate and cannot be detected by the recovering mechanisms. The Selective Forward attack is more effective if it is backed by traffic analysis.

The Network Manager in the WirelessHART network is responsible for general monitoring of the network; the Handheld device is used to monitor the specific device. They should collectively monitor the network on regular basis to detect and eliminate these attacks. The WirelessHART command 779 (Report Device Health) can be useful in detecting this attack.

6.3.11 Exhaustion

Any device that supports the WirelessHART protocol stack and has knowledge of network parameters (Network ID, Device ID, etc.) can send messages to the neighboring devices using the Well-known key. A fake device can use the Well-known key for calculating the MIC over the DLPDU and can use a fake Join key to encrypt and authenticate the NPDU. Although this message will be discarded when received by the Network Manager (as it uses a faked Join key) it consumes network resources along the route from the field device to the Network Manager. If a series of such join attempts are initiated by an active attacker then it can give rise to a serious DOS effect/risk.

In WirelessHART networks, the attacker can only send these messages using the join slot which will not affect the communication among other networked devices. The protection of non-cryptographic secrets (Network ID, Device ID, etc.) can also eliminate this attack.

6.3.12 Spoofing

Field devices in the WirelessHART network use the Well-known key not only for joining the network but also for advertisements [3]. The adversary can spoof the new joining device by sending fake advertisements and on receipt of the

[3] WirelessHART devices have Advertisement slots that are used to publish the device presence to the new potential devices who wish to join the network.

join request it can simply discard it. If the fake device has access to the valid Network key then the spoofing attack is more effective since the device can announce its presence to the other legitimate networked devices. Moreover, this can result in a serious blockage of network traffic.

The use of different devices while joining the network can overcome this attack. The regular monitoring and changing of the Network key by the Network Manager can minimize this attack as well.

6.3.13 Collision

Collisions can occur when two or more devices try to access the same frequency channel at exactly the same time; collision can be intentional or unintentional. An attacker can also introduce collision in small portion of the packet [21].

The combination of time diversity and frequency diversity is used to minimize the collision and CRC-16 is used to detect the collision in the WirelessHART network. To *minimize* the collision, the WirelessHART protocol provides scheduled data transmission based on time slotting; TDMA and channel hopping is used to control access to the network [9]. The CRC is used to *detect* the collision based on ITU-T polynomial (aka CRC-16) [10].

The CRC-16 might not be able to detect the insertion attack (see security consideration in [10]). This attack can be avoided by better implementation and active coordination between the Physical and Data-link layer especially when the physical layer connection state changes.

6.3.14 Summary

The WirelessHART standard is secure enough to provide defense against most of the attacks. However, wormhole, de-synchronization, jamming, traffic analysis, spoofing, and exhaustion attacks need more attention.

Other than these attacks, the physical protection of the WirelessHART devices is very important. If the device is captured by the attacker it should self destruct because otherwise it can be cloned and the secret contents can be revealed. When a device is disconnected from the network, it should wipe out its volatile memory.

6.4 WirelessHART Security Manager

The Security Manager is an integral wired device in the WirelessHART network. Some of the critical points about the WirelessHART Security Manager are:

- One Security Manager can serve more than one WirelessHART network but there is only one active Security Manager per network.

- The Security manager is an application that meets the security needs of the wireless network. It can reside in a standalone device; it can be a function in the PAH; and it can be integrated in the black box consisting of Gateway, Network Manager, and Security Manager.

- The Security Manager cannot create sessions with the wireless devices; also, it is completely hidden from the Gateway.

- The interface between the Security Manager and Network Manager is not defined by the standard.

- The Security Manager provides security keys to the Network Manager that distributes them to the respective wireless devices.

Based on these prerequisites, we propose that the Security Manager should be directly connected using a dedicated link with the Network Manager at one end and with the wired/core network at the other end. This way, the Security Manager is capable of serving both the wired and wireless networks. Also, the Security Manager can serve more than one Network Manager, but the other Network Managers should be connected to the core network at one end (the other end may be connected with the Gateway). Figure 6.4 shows the placement of the Security Manager (SM) in the WirelessHART network.

According to the WirelessHART standard, the core responsibility of the Security Manager is to manage security keys. However, as a key manager the Security Manager is responsible for generation, storage, revocation, and renewal of keys. The Security Manager is not responsible for the distribution of keys to the wireless devices; instead the Security Manager provides keys to the Network Manager that in turn distributes them to the devices. The commands [22] for the key distribution are listed the Table 6.2.

As of other security functionalities, the security keys are not clearly mentioned in the WirelessHART standard and therefore we elucidate them. In a

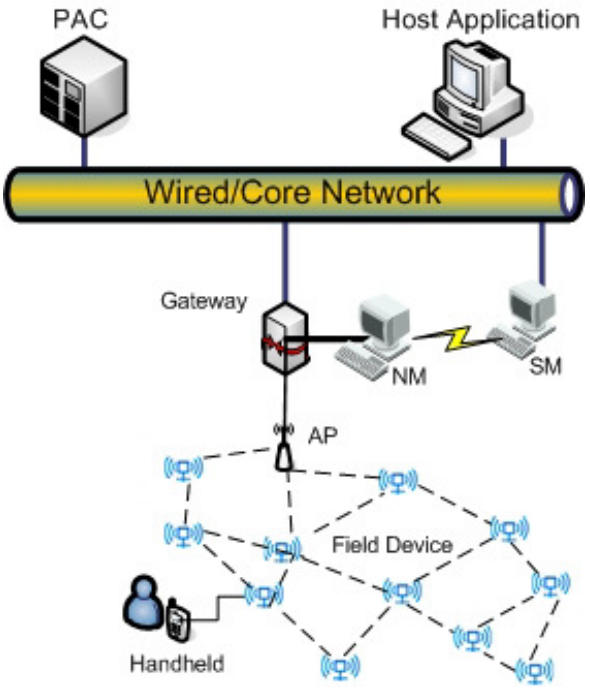


Figure 6.4: Our Proposed Placement of Security Manager in the network

WirelessHART network at maximum eight different keys can be used to encrypt/decrypt the NPDU payload and to calculate the MIC at the Network and the Data-link layer. These are:

1. *Network Key*: Used to calculate the MIC over the DLPDU. It is also used for changing the broadcast session keys.

2. *Join Key*: Used to secure the NPDU [4] during the joining process. It is also used when changing the unicast session keys both of the Network Manager and the Gateway.

3. *Unicast-NM*: Used to secure the NPDU during the communication between the Network Manager and a specific Field device. It is also used

[4]For encrypting/decrypting the NPDU payload and calculating the MIC over entire NPDU.

Keys	Commands
Session Keys	Command 963 (Write Session)
Network Key	Command 961 (Write Network Key)
Handheld Key	Command 823 (Request Session)
Join key	Command 768 (Write Join Key)

Table 6.2: Key Distribution Commands in WirelessHART

for changing the Join key.

4. *Unicast-Gateway*: Used to secure the NPDU during the communication between the Gateway and a specific Field device.

5. *Broadcast-NM*: Used to secure the NPDU during a Network Manager broadcast to all the field devices. It is also used for changing the Network key.

6. *Broadcast-Gateway*: Used to secure the NPDU during Gateway broadcast to all field devices.

7. *Handheld key*: Used to secure the NPDU during the communication between the Handheld device and the connected Field device.

8. *Well-known key*: Used to calculate the MIC over the DLPDU during the join process and while sending advertisements. The value of the Well-known key is always **772E 6861 7274 636F 6D6D 2E6F 7267.**

All wireless devices have a pre-shared Join key; the Security Manager stores all the Join keys as well. During the joining process the Network Manager asks the Security Manager for the Join key of a new joining device. This key is used to authenticate the NPDU payload and verify the MIC of the joining request. On successful authentication, all other keys are distributed to the devices.

Another important aspect the standard lacks is the interaction between the Security Manager and the Network Manager. The Security Manager manages the keys and the Network Manager uses or distributes them to the Field devices and the Gateway. The Network Manager can request a specific key from the Security Manager by providing the following parameter over a secure channel.

1. *Network ID*: As one Security Manager can serve more than one WirelessHART network each network is uniquely identified by the Network ID.

2. *Nickname*: The Network Manager maintains a list of 2-bytes Nicknames that are used to uniquely identify the WirelessHART devices. The Unique ID (UID) can be used but UIDs are maintained by the Gateway and the Security Manager cannot cannot communicate with the Gateway directly.

3. *Key Type*:The key type can be one of the seven key types listed above. The Well-known key is always the same and can be hardcoded in the Network Manager.

The WirelessHART standard does not specify the security in the wired part of the network. However, the capabilities of the Security Manager can be extended to secure the connection between the wired devices based on asymmetric or public key cryptography [23].

6.5 Security Limitations of WirelessHART

Although the WirelessHART standard is designed to be a secure and reliable protocol intended to be used for industrial process automation the current release of the standard has some security limitations. These include:

- The WirelessHART protocol does not support public key cryptography which makes it unable to provide certain security services such as non-repudiation. Strong authentication, i.e. authentication without sending the security secrets over the network is not possible either.

- No mechanisms have been specified to provide authorization and accounting security services. We need accounting when the cost of WirelessHART device is attached to its usage.

- The complete key management system is not specified; however, the commands for distribution of keys have been specified.

- Security in the wired part of the network is neither specified nor enforced.

- Secure multicast communication among the Field devices is not supported.

- Secure integration of wireless and legacy HART is not specified in the WirelessHART standard.

- The architecture of the Security Manager and the interface between the Security Manager and the Network Manager is not specified in the standard.

6.6 Conclusions and Future Work

We have thoroughly discussed the security features in the WirelessHART standard and analyzed the specified security features against the available threats in the wireless medium. We have also identified some security limitations in the standard. However, the provided security in the wireless medium, although subjected to some threats due to its wireless nature, is strong enough to be used in the industrial process control environment. The physical protection of the WirelessHART devices is very important to avoid device cloning and stealing security secrets which will lead to other security attacks. Also, the careful implementation of the Network Manager is very important. The WirelessHART standard does not enforce security in the core/wired network but the connections between the wired devices must be secured. The standard provides core security services including *Confidentiality*, *Integrity*, *Authentication*, and *Availability*; however, other necessary services such as *Non-repudiation*, *Authorization* or *Access Control*, and *Accounting* are yet to be provided.

The reserved security bits (see Security control byte [6]) can be used to enhance WirelessHART security with public key cryptography [24] [25]. Although PKI is avoided in embedded devices, having a central trusted authority (Network Manager/Security Manager) and relatively high processing power and energy resources makes WirelessHART devices different from traditional sensor devices. Research in implementing ECC and RSA on sensor nodes have shown the potential for PKI in WSNs [26]. The WirelessHART's counterpart ISA100.11.a:2008 [27] also uses public key cryptography. One way to enrich the standard with security features is to identify and specify ways to provide additional security services such as *accounting* and *access control/authorization*.

Acknowledgments

This work has been performed within the SICS Center for Networked Systems funded by VINNOVA, SSF, KKS, ABB, Ericsson, Saab Systems, TeliaSonera

and T2Data. This work has been partially supported by CONET, the Cooperating Objects Network of Excellence, funded by the European Commission under FP7 with contract number FP7-2007-2-224053.

Bibliography

[1] Anna N. Kim, Fredrik Hekland, Stig Petersen, and Paula Doyle. When hart goes wireless: Understanding and implementing the wirelesshart standard. *IEEE International Conference on Emerging Technologies and Factory Automation*, pages 899–907, September 2008.

[2] *IEC approves WirelessHART*. Control Engineering, Vol. 55 Issue 10 Pages 34-34, October 2008.

[3] *WirelessHART Device Specification,HCF_SPEC-290, Revision 1.1*. HART Communication Foundation, May 2008.

[4] *HART Communication Foundation (HCF)*. 9390 Research Blvd., Suit I-350 Austin TX 78759 USA. http://www.hartcomm2.org/index.html.

[5] Cyril Leung. Evaluation of the undetected error probability of single parity-check product codes. *IEEE Transactions on Communications*, 31(2):250–253, 1983.

[6] *Network Management Specification,HCF_SPEC-085, Revision 1.1*. HART Communication Foundation, May 2008.

[7] D. Whiting, R. Housley, and N. Ferguson. *Counter with CBC-MAC (CCM), RFC 3610*. IETF, Network Working Group, Fremont, California 94538 USA, September 2003.

[8] Morris Dworkin. *Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality*. NIST Special Publication 800-38C, May 2004.

[9] *TDMA Data Link Layer Specification, HCF_SPEC-075, Revision 1.1*. HART Communication Foundation, May 2008.

[10] W. Simpson. *PPP in HDLC Framing, RFC 1549*. IETF, Network Working Group, Fremont, California 94538 USA, December 1993.

[11] William Stallings. *Data and Computer Communications*, pages 277–282. Prentice Hall, eighth edition, 2006.

[12] Christopher Alberts and Audrey Dorofee. *Managing Information Security Risks: The OCTAVE Approach*. Addison Wesley, 09 July 2002.

[13] Tomas Lennvall, Stefan Svensson, and Fredrik Heklan. A comparison of wirelesshart and zigbee for industrial applications. *IEEE International Workshop on Factory Communication Systems*, pages 85–88, May 2008.

[14] Yee Wei Law, Marimuthu Palaniswami, Lodewijk Van Hoesel, Jeroen Doumen, Pieter Hartel, and Paul Havinga. Energy-efficient link-layer jamming attacks against wireless sensor network mac protocols. *ACM Transactions on Sensor Networks (TOSN)*, 1(5):71–80, February 2009.

[15] John R. Douceur. The sybil attack. *1st International workshop on Peer-To-Peer Systems (IPTPS)*, March 2002.

[16] Jianping Song, Song Han, Aloysius K. Mok, Deji Chen, Mike Lucas, and Mark Nixon. Wirelesshart: Applying wireless technology in real-time industrial process control. *Real-Time and Embedded Technology and Applications Symposium, 2008(RTAS-08)*, pages 377 – 386, April 2008.

[17] Levente Buttyan and Jean-Pierre Hubaux. *Security and Cooperation in Wireless Network*. Cambridge University Press, 2007.

[18] Yih-Chun Hu, Adrian Perrig, and David B. Johnson. Wormhole attacks in wireless networks. *IEEE Journal on Selected Areas in Communications*, 24(2):370– 380, February 2006.

[19] Andrey Bogdanoy. Multiple-differential side-channel collision attacks on aes, lecture notes in computer science. *10th international workshop on Cryptographic Hardware and Embedded Systems*, 5154(2):30–44, 2008.

[20] Raphael Chung-Wei Phan. Impossible differential cryptanalysis of 7-round aes. *Information Processing Letters*, 91(1):33–38, 2004.

[21] Hiran Kumar Deva Sarma and Avijit Kar. Security threats in wireless sensor networks. *IEEE A&E Systems Magazine*, June 2008.

[22] *Wireless Command Specification, HCF_SPEC-155, Revision 1.1*. HART Communication Foundation, May 2008.

[23] Harold F. Tipton and Micki Krause. *Information Security Management Handbook*, pages 1129–1135. Auerbach Publications, sixth edition, 2007.

[24] An Liu and Peng Ning. Tinyecc: A configurable library for elliptic curve cryptography in wireless sensor networks. *International Conference on Information Processing in Sensor Networks, 2008. IPSN '08*, pages 245–256, April 2008.

[25] Haodong Wang, Bo Sheng, and Qun Li. Elliptic curve cryptography-based access control in sensor networks. *International Journal of Security and Networks*, 1(3/4):127–137, 2006.

[26] Haodong Wang and Qun Li. Efficient implementation of public key cryptosystems on mote sensors. *In Proceedings of International Conference on Information and Communication Security (ICICS)*, pages 33–38, 2004.

[27] ISA. *ISA-100.11a-2009: Wireless systems for industrial automation: Process control and related applications*. ISA Standards, 67 Alexander Drive Research Triangle Park, NC 27709 USA, May 2009.

Chapter 7

Paper B: Design and Implementation of a Security Manager for WirelessHART Networks

Shahid Raza, Thiemo Voigt, Adriaan Slabbert, Krister Landernäs
5th IEEE International Workshop on Wireless and Sensor Networks Security (WSN'S 2009), in conjunction with MASS'2009, 12-15 October 2009, Macau SAR, P.R.C..

Abstract

WirelessHART is the first open standard for Wireless Sensor Networks designed specifically for industrial process automation and control systems. WirelessHART is a secure protocol; however, it relies on a Security Manager for the management of the security keys and the authentication of new devices. The WirelessHART standard does not provide the specification and design of the Security Manager. Also, the security specifications in the standard are not well organized and are dispersed throughout the standard which makes an implementation of the standard more difficult.

In this paper we provide the detailed specification and design as well as an implementation of the Security Manager for the WirelessHART standard. We evaluate our Security Manager against different cryptographic algorithms and measure the latency between the Network Manager and the Security Manager. Our evaluation shows that the proposed Security Manager meets the WirelessHART requirements. Our analysis shows that the provided Security Manager is capable of securing both the wireless and wired part of the WirelessHART network.

7.1 Introduction

WirelessHART is a secure and reliable wireless sensor/mesh network protocol developed by the HART Communication Foundation (HCF). The WirelessHART protocol is standardized in 2007 by IEC [1] and currently it is the only open standard for the Wireless Sensor Networks (WSN) designed primarily for industrial process automation. A wireless interface is introduced in the release 7 of the Highway Addressable Remote Transducer (HART) protocol and named as WirelessHART^{TM}. HART 6 and earlier are current looped wired protocols and are insecure. Only a single parity check coding schemes [2] is used to detect communication errors in legacy HART. In contrast, WirelessHART [3] is a secure and a reliable protocol.

The WirelessHART network initiation is started by the Network Manager (NM). The NM is a centralized entity responsible for the overall network management, scheduling, initiation, maintenance, monitoring, and resource management. The NM accepts joining requests from the Gateway, Access Points, Field devices, Adapters, Routers, and the Handheld devices. In addition, it allocates network resources, see the WirelessHART Device Specification [4]. The Field devices sense process data using equipped sensors and securely send it to the host applications or other Field devices through the Gateway. All wireless devices communicate using the 2.4 GHz frequency band. They create secure sessions with the Gateway and the NM.

WirelessHART is a hybrid network consisting of both wireless and wired devices. The standard provides the means to secure the wireless part but the security in the wired part is neither specified nor enforced. The standard specifies the need of a Security Manager (SM) to provide key management; but the standard does not elucidate a Key Management System (KMS). Also, the standard does not provide the complete specifications, design, and organization of the SM. The standard emphasizes that the connections between the SM and the NM, the Gateway and the NM, and the Gateway and host applications must be secured; but it does not specify the ways to secure these connections.

We design and implement the first open SM for WirelessHART. Our implementation that is based on standard technology, provides a complete KMS for WirelessHART networks. We also provide authentication of wireless devices and solutions to secure the wired part of the network. We specify how the SM interacts with the other devices in the network and what parameters are exchanged during these interactions. We implement our system using state-of-the-art technology. We experimentally evaluate our SM against different cryptographic algorithms and measure the latency for key generation and the

required communication between the NM and SM. Our results show that we meet all related timing requirements of the standard. Our experiments demonstrate that the latency can be further reduced by pre-generating keys.

This paper makes two main contributions. First, we design and implement the first open SM for WirelessHART that includes KMS, authentication of wireless devices and solutions to secure the wired part of the network. Second, we show that applying state-of-the-art technologies is sufficient for meeting the timing requirements of the security parts of WirelessHART.

This paper proceeds as follows. In Section 7.2 we give an overview of the provided security in the WirelessHART standard. In Section 7.3 we elaborate specifications of the SM. Section 7.4 and 7.5 provide the design and the implementation of the SM respectively. In Section 7.6 we provide an evaluation of the SM. Section 7.7 shows related work and the paper ends with the conclusion and future work.

7.2 Security in WirelessHART

In this section we give an overview of the provided security in the wireless part of the WirelessHART standard. We list all security keys needed and their role. Finally, we explain the functions and capabilities of the SM listed in the standard. We use these functions as foundations to specify, design, and evaluate our SM.

The WirelessHART standard provides communication security between two end devices i.e. the source and the destination at the Network layer and between two neighboring devices (one hop apart) at the Data-link layer. The standard does not provide mechanisms for device security and data storage security. The 128 bit AES block cipher in the Counter with CBC-MAC (CCM) [5] mode is used to secure the sessions between end devices. In CCM, the Counter mode is used for encryption of the Network Protocol Data Unit (NPDU) payload. The Cipher Block Chaining-Message Authentication Code (CBC-MAC) mode in the CCM is used to calculate the Message Integrity Code (MIC) over the entire NPDU. The same key is used for both the Counter and the CBC-MAC modes. The type of the key used at the Network layer depends on the type of message; these keys are discussed in next section. The Network layer provides both confidentiality by encrypting the NPDU payload and integrity by calculating the keyed MIC over the entire NPDU. Although two neighboring Field devices can create direct peer-to-peer session at the Network layer the standard prohibits such connections due to security reasons. The communication

between Field devices is always through the Gateway. The Gateway has unicast and broadcast sessions with all the Field devices. Handheld devices create direct peer-to-peer sessions with a Field device using the Handheld key.

The Data-link Layer (DLL) provides authentication services between two neighboring devices by calculating the MIC over the DLL Protocol Data Unit (DLPDU). Here too the AES block cipher in CCM mode is used for calculating the MIC. As the encryption is not used at the DLL, the encrypted message parameter (m) for the AES-CCM is set to zeros. The Network key or the Well-known key is used to calculate the MIC. In the next section we provide details about these keys.

7.2.1 Security Keys in WirelessHART

Before delving deeper in the KMS we first elaborate all the keys needed in WirelessHART as the standard does not specify them clearly. A total of eight keys can be used in WirelessHART networks. These are:

1. ***Join key***: All wireless devices must be equipped with the Join key before joining the network. The Security Administrator (SA) manually distributes this key to the devices. The device's maintenance port can be used to add the Join key to the device. The Join key acts as a password that the device uses to authenticate it to the NM. The Join key is used at the Network layer to encrypt the payload and to calculate the MIC. The NM uses the Join key to renew unicast session keys.

2. ***Unicast-Gateway key***: The Unicast-Gateway session key is used to provide secure communication between the Gateway and a Field device and hence between two Field devices. The Gateway has secure sessions with all Field devices and two Field devices should always communicate through the Gateway. The Unicast-Gateway and all other session keys are used for the NPDU payload encryption and MIC calculation at the Network layer.

3. ***Unicast-NM key***: The Unicast-NM session key provides secure messaging between the NM and wireless devices. The NM uses this session for device specific management such as asking for device health information, allocating time slots etc. The Unicast-NM key is also used for changing the Join key when the device is part of the WirelessHART network.

4. ***Broadcast-Gateway key***: The Broadcast-Gateway session key is used for sending secure broadcast messages from the Gateway to the Filed devices. These messages can include general notifications, timing information, etc.

5. ***Broadcast-NM key***: The Broadcast-NM is used for sending global secure messages to the wireless devices and the Gateway. These messages include routing information , network scheduling, etc. This key is also used for changing the Network key.

6. ***Handheld key***: After authenticating itself to the NM using the Join key a Handheld device can request a Handheld key. The NM provides this key to both the Handheld device and the Field device. The Handheld device uses this key to create a secure one-to-one session with the Field device. The Handheld key secures the NPDU by encrypting the payload and by calculating the MIC.

7. ***Network key***: The Network key provides defense against outside attacks. The Network key is used to calculate the keyed MIC to secure the DLPDU. Two neighboring devices authenticate each other by verifying the MIC. The Network key is shared amongst all authenticated devices. The NM uses the Network key for renewing the Broadcast session keys.

8. ***Well Known key***: All messages in the WirelessHART network must be encrypted. During the join process a device is not authenticated and hence does not have the Network key. A known network key called the Well-known key (777 772E 6861 7274 636F 6D6D 2E6F 7267) is used to calculate the MIC for the join request/response messages. The Well-known key is also used for sending join advertisements.

7.2.2 The Security Manager in the Standard

The WirelessHART standard very briefly specifies the functions of the SM. According to the standard the SM is responsible for managing the security keys for the wireless devices and the authentication of new devices. The standard does not specify the architecture of the SM and its organization in the network. However the following about the SM is mentioned in the standard:

- There is only one SM in a network but one SM can serve more than one WirelessHART networks.

- The SM can exist as a standalone entity, it can be a function in the host application, or it can reside within the NM.
- The SM is completely hidden from the Gateway and must not communicate directly with Field devices.
- The connection between the SM and the NM, the Gateway and the NM, the Gateway and host applications must be secured, but the standard does not specify the ways to secure these connections.

7.3 Security Manager Specifications

The Security Manager (SM) is an integral part of the WirelessHART network but unlike other devices such as the NM, Gateway, etc. the requirements and functions of the SM are not clearly defined in the standard. In this section we elaborate specifications for the SM that can secure the entire WirelessHART network. The broader capabilities of our proposed SM include:

7.3.1 SM as Key Manager

The SM we propose provides a KMS along with securing the wired part of the WirelessHART network. The SM is responsible for the management of all security keys (see Section 7.2.1) except the Well-known key. By key management we mean the generation, storage, distribution, renewal, and revocation of the security keys. The design of the KMS varies with the structure of the underlying WSN. Broadly speaking, the structure of a WSN can be distributed or hierarchical [6]. In a distributed structure there is no fixed infrastructure and the network topology is unknown before the deployment. The hierarchical WSN establishes a hierarchy among the devices based on their capabilities and normally comprises a base station and sensor nodes.

The WirelessHART network is hierarchical in nature consisting of a base station (Gateway), a central station (NM), and sensor nodes (Field devices). Each wireless device has a preshared symmetric key that the device uses to authenticate itself to the network. After successful authentication the central station distributes the session keys and the Network Key to the wireless devices. The devices use the session keys to secure end-to-end communication and use the Network key for secure per-hop communication.

The key management process starts with the generation of Join keys. The Join key is a device specific master key which is initially generated and stored

by the SA. A device tries to join the WirelessHART network using the Join key. On successful authentication the SM generates four session keys: the Unicast-Gateway, the Unicast-NM, the Broadcast-Gateway, and the Broadcast-NM. When the WirelessHART network is initialized the NM requests the Network key from the SM that creates a Network key, stores it locally, and sends it to the NM. A Handheld device can request the NM for the Handheld key; in turn, the NM requests and receives a Handheld key from the SM and forwards it to the Handheld device and the associated Field device.

The SM stores all the generated keys in a secure storage and the key related information such as Network ID, Nickname, Device Address, Device Identity, Key Type, Status, Generation Date, Expiry Data, etc. in a key database. The key storage is protected with a storage password and the individual keys in the storage can be protected with a key password. For simplicity the storage and the key passwords can be the same. The key related information in the key database is stored as plain text but the database can be protected with a password.

The NM can request any key from the SM. In response, the SM returns the appropriate key based on the type of parameters the NM passes in the key request. We propose the following list of necessary parameters. The NM can send a subset of parameters from this list.

1. ***Key Type***: The name of the key requested. It can be one of the seven key types listed in Section 7.2.1. The NM always sends this parameter in the key request.

2. ***Network ID***: Each WirelessHART network has a unique ID. A single SM can serve more than one WirelessHART networks and hence the Network ID is used to uniquely identify the specific network. The NM always sends this parameter too.

3. ***Nickname***: In the NM each authenticated WirelessHART device is identified by a Nickname. When a device successfully joins the network the NM assigns it a Nickname. All key requests except the Join key contain this parameter.

4. ***Device Address***: The joining device is not yet a part of the network and hence has no Nickname. In this case the NM passes the device address as device identifier.

5. ***Device Identity***: It is a device's identity, i.e. the response to WirelessHART Command 0 (Read Unique Identifier) or Command 20 (Read

Long Tag). It is used to authenticate the new joining device.

The NM uses the returned Join key to authenticate the device. The received session keys and the Network key are distributed to the appropriate device. The NM uses standard commands [7] to write these keys in the actual device. Command 963 (Write Session) is used to write all session keys in the device, Command 961 (Write Network Key) writes the Network key, the response to the Command 823 (Request Session) writes the Handheld key, and the Command 768 (Write Join Key) writes the new Join key in the device.

All keys in WirelessHART are renewable except the Well-known key. The NM sends key renewal requests to the SM with Key Type, Network ID, and Nickname as parameter. The SM verifies the parameters, i.e. it checks whether the key is present in the storage and if it finds one it creates a new key, then deletes the old key and updates the database and ultimately sends the new key to the NM. The NM writes it in the actual device using one of the above commands.

When a device leaves the network, the NM initiates a key revocation request to the SM. The SM sets the status of all session keys including the Handheld key inactive for that specific device. The session keys can be revoked immediately but in this case the device history will be deleted which can be useful for future network analysis. The SA can revoke the Join key through SM.

7.3.2 SM as Device Authenticator

Our SM authenticates new devices to the NM. A device tries to join the WirelessHART network using the Join key and the associated information including the Device's Identity extracted from the Command 0 or the Command 20 response. Using the Join key the new device encrypts the joining message NPDU payload containing the Device's Identity and sends it to the NM. Here we have a chicken and egg situation: for requesting the Join key to decrypt this message the NM needs to pass Device's Identity to the SM, but for extracting the Device's Identity the NM has to decrypt the joining request (the encrypted NPDU) with the Join key. To overcome this problem we have to rely on some unencrypted field in the NPDU. Our solution to this is to use source device address. The source address should be 8 byte Extended Unique Identifier (EUI). We cannot use the 2 bytes Nickname for device identification because the Nickname is allocated after the device joins the network. The same source address should be added in the SM during the device registration. The NM sends the

authentication request to the SM with Key Type (i.e. Join), Device Address, and Network ID as parameters. The SM verifies the Device Address and extracts the stored Join key and sends it back to the NM. The NM decrypts the joining request NPDU payload with the provided Join key and if successfully decrypted, it reads the Device Identity and sends it to the SM along with the Network ID, Device Address, and provided Join key. The SM authenticates the Device's identity against the Join key and returns either a success or a failure message.

7.3.3 SM as Certification Authority

The WirelessHART network is a hybrid network having both wired and wireless devices. The standard enforces security in the wireless part of the network but the security in the wired part is neither specified nor enforced. However, the standard asserts that the connection between the wired devices must be protected. The wired part contains core devices such as NM, Gateway, Host Application, and SM. Unlike wireless sensor devices, the wired devices are not resource scarce and hence public key cryptography is an obvious option to secure the wired part of the network. The Public Key Infrastructure (PKI) [8] is a secure way to mutually authenticate each other and exchange a symmetric key that is later used for normal message encryption/decryption.

In order to overcome man-in-the-middle (MITM) [9] attacks in the public key cryptography we develop a PKI. In a PKI, the best approach is to use digital certificates. In a digital certificate a public key is bound to an entity and is digitally signed by a trusted authority called Certification Authority (CA). A trusted certificate can be used to sign other certificates and hence a trust hierarchy is developed. To implement PKI in WirelessHART we can either get a signed certificate from some known CAs such as Verisign or we can develop our own CA. We propose the use of the SM as WirelessHART CA since this is a more secure and a cost effective solution, as the SM is a trusted entity that is internal to the network, can be easily controlled, and we already trust and rely on it for the symmetric security keys.

As a CA the SM generates its public-private key pair and creates a self signed certificate containing its public key. It also issues signed certificates and the corresponding private keys to the NM, Gateway, Plant Automation Hosts (PAHs), and other potential WirelessHART devices in all supported WirelessHART networks. All wired devices have trust stores that contain CA (SM) signed certificates of other devices and key stores that contain private keys and CA signed self certificates.

7.4 Security Manager Design

We propose the first open design of the SM for WirelessHART networks. Our design is fully compatible with the WirelessHART standard and covers all the features specified in Section 7.3. The complete functions of the SM and its interaction with other network devices is represented in the form of a *use-case diagram*. The Use-case diagram is a combination of actors (normally outside the system) and their interaction with the system functionalities called use cases; it also shows the collaboration among the use cases. Figure 7.1 shows a use-case diagram with actions, use-cases, and interaction among them.

The ***actors*** that can interact with our SM are:
SA: They are responsible for the administration of the SM which includes:

1. Generation of the Join keys and registration of the devices.
2. Renewal/revocation of Join Keys. The Join key can also be renewed when the NM sends a renewal request.
3. Creation of secure storage(s)
4. Performing back-ups
5. Creating, signing, revoking, and distributing security certificates and corresponding private keys.

NM: The NM can request:

1. Any of the predefined Join keys
2. The generation of Network and session keys
3. Any of the generated keys
4. Renewal of any key (including Join keys)
5. Revocation of any key

The ***use-cases*** of the SM are grouped into two packages:
KeyManager: It serves the NM and manages the symmetric keys needed to secure the communication among the wireless devices, the Gateway, and the NM. The functions of the KeyManager include: generation, renewal, revocation, distribution, and secure storage of symmetric keys.

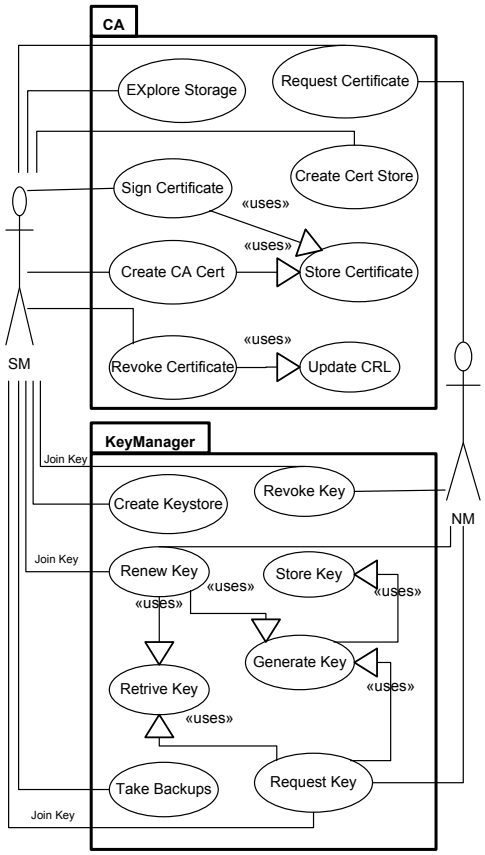


Figure 7.1: Our proposed use-case diagram of WirelessHART SM

Certification Authority (CA): The CA is responsible for securing the communication in the wired/core medium. This approach is based on asymmetric cryptography where the public keys (certificates) and the corresponding private keys are used to secure the communication. Every device in the wired/core network will trust on the security certificate signed with the SM's private key. As a CA, the SM will be responsible for:

1. Creating a self-signed certificate.
2. Issuing signed certificates to the other network devices.
3. Registering the requests for new certificates from the newly joined devices in the wired network.
4. Revocation of the SM signed certificates held by the devices which are no more part of the WirelessHART network.
5. Creation of a key store for storing private keys and and trust stores for storing certificates signed by the SM.
6. Exploration/Investigation of stored certificates (by the SA).

The following subsections show our proposed protocol steps needed to carryout key management and certification.

7.4.1 Key Request

The NM relies on the SM for all the keys except the Well-known key. On successful authentication, the NM requests the session keys from the SM and distributes the returned keys to the wireless devices. The NM needs Unicast-NM and Broadcast-NM keys to encrypt/decrypt the messages, and all other keys to distribute them to the wireless devices and the gateway. The NM gets these keys from the SM by sending a key request.

The NM initiates key requests by sending a subset of Key Type, Nickname, Network ID, Device Address, and/or Device Identity (see section 3.1.). The SM tries to retrieve the requested key from the secure storage. The key is returned if it is found in the storage. Otherwise the SM generates a key, stores it locally for later use, and returns a copy of it to the NM that either uses it locally for decrypting the NPDU or sends it to the actual requesting device using one of the commands specified in Section 7.3.1. Figure 7.2 shows the key request process.

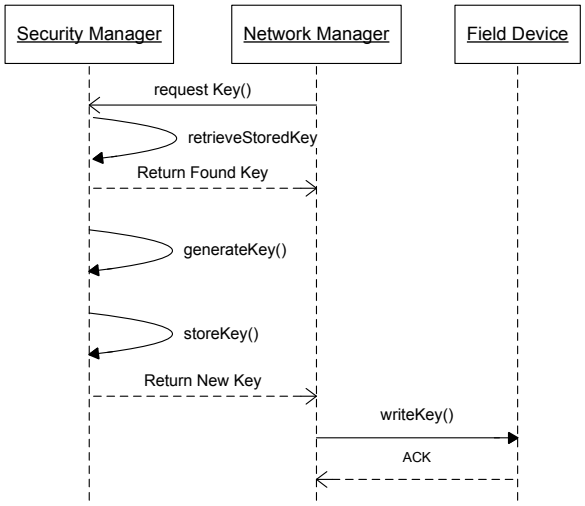


Figure 7.2: Our proposed Key Request Protocol steps

7.4.2 Key Renewal

All the keys in the WirelessHART network are renewable except the Well-known key. The key renewal request can be initiated by the SA (only for the Join key), the SM (when keys expire), or by the NM. If the request is generated by the SA or the SM the NM is notified with the key renewal request. The SM will not change the key until or unless it receives a renewal request from the NM; this is because the NM has to write the changed key into the actual device as the SM cannot make sessions with the Field devices and the Gateway. When the NM receives a key renewal notification or needs to change keys itself, it requests the SM to change the key. The SM verifies the request, changes the key, and returns the new key to the NM that sends it to the actual wireless device or the Gateway. Figure 7.3 shows the key renewal process.

7.4.3 Key Revocation

Key revocation is simply a deletion of keys from the secure storage and its related information from the key database. The key revocation request can be generated by the NM or the SA. The parameters in the key revocation request

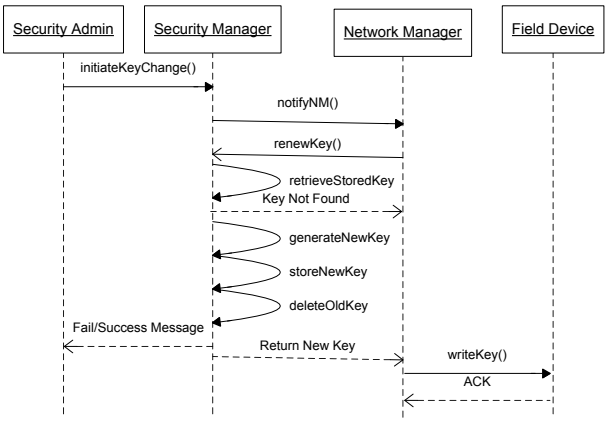


Figure 7.3: Our proposed Key Renewal Protocol steps

are Network ID, Nickname, and Key Type. On receiving a revocation request the SM deletes the corresponding key and responds with a success or a failure message. The key revocation is needed when the device leaves the network or the Handheld-to-Field device (peer-to-peer) session expires; in the former case all keys are revoked and in latter case only the Handheld key is revoked.

7.4.4 Key Generation

The data that flows through the WirelessHART network is secured using 128 bit AES that is recommended by the NIST USA and considered strong enough since it is hard in terms of time and cost to break it using brute force attacks within an effective time [10]. But if the generated keys are not random enough the statistical attacks can break the keys in less time and with few resources [11]. Hence the key generation mechanisms should be based on secure random/pseudorandom sources to create a random key.

For secure communication in a WirelessHART network, the first key the device should be provisioned with is the Join Key and the first key the NM needs is the Network Key. For generating the Join key, the real random sources such as thermal noise, gas discharge tubes, response time of hard disk sector reading [12], etc. can be used. For test purposes, the secure random source can be a key password (secure) and current system time in milliseconds (random).

The generated Join key is later combined with random source to generate session keys. For the Network key the real random source can be combined with the Administrator's password. The password and the output of the random source are Exclusive-ORed to get a secure random output. Sometimes the random sources may get biased and produce uneven output containing a series of ones or zeros. To overcome this, the output is hashed to get random distribution of ones and zeros.

In our implementation, the pseudorandom number generator is cryptographically strong as it complies with Section 4.9.1 of FIPS 140-2 (Security Requirements for Cryptographic Modules). Also, the final random number complies with the Randomness Recommendations for Security defined in the RFC 1750. Figure 7.4 shows the key generation process.

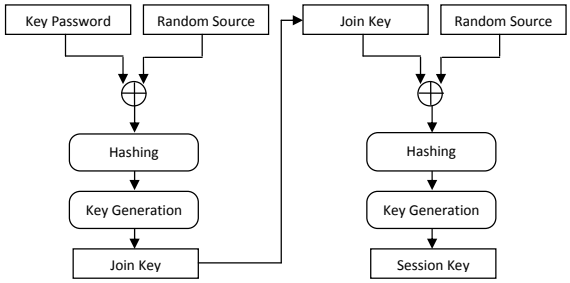


Figure 7.4: Our proposed Key Generation process

7.4.5 Key Storage

The SA registers the new device in the WirelessHART Network by generating and adding the Join key and related information in the secure storage and key database respectively. The actual key is stored in the key store in an encrypted form. The keys in the store are protected with the key password and the whole storage is protected with the storage password. The information associated with a specific key is stored in the key database protected with the database password. The Key database contains Key Store Aliases (alias for the key store where actual key is stored in a protected form), Network ID, Nickname, Device ID, Device Identity, Key Type, Generation Date, and Expiry Date. The storage password is shared between the SA and the NM/Administrator. The NM provides the storage password at the time of connectivity. If every stored

key is encrypted with a different password then the SM has to keep track of all the passwords; so in practice, all keys are encrypted with a single password that the SA enters at the time of launching SM application. Figure 7.5 shows our key storage model.

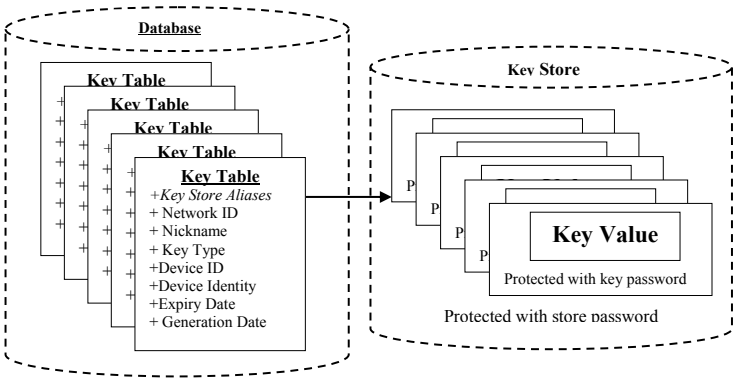


Figure 7.5: Our proposed Key Storage Model

7.4.6 Wired Network Security

The wired devices are secured using a Public Key Infrastructure (PKI). Our SM as CA develops this infrastructure. The CA generates public-private key pairs, composes certificates containing public keys, and signs certificates. The private key and corresponding signed certificates are manually distributed to the device. It is highly recommended that the private key should be stored in a smartcard as the smartcard provides a tamper resistant way to secure private keys [13]. The device can store its certificate and private key in a secure repository often called key store. The certificates of the other WirelessHART devices are stored in a trusted repository called the trust store. The SA can predistribute all required certificates to the devices or the devices can exchange certificates during the authentication phase. It all depends on the type of protocol we use, e.g. using the TLS/SSL protocol the certificates are exchanged during the session establishment [14].

A wired device in the WirelessHART network communicates with another device by digitally signing the authentication request with its private key and

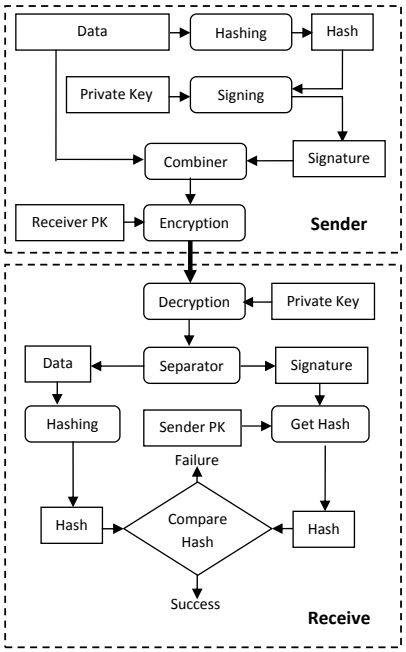


Figure 7.6: Generic PKI based authentication

encrypting it with the receiver's public key. Now only the intended receiver can decrypt these messages using its private key and can authenticate the message by verifying the sender's digital signature. This ensures sender and receiver authenticity, data confidentiality, data integrity, and non repudiation. The authentication request also contains a symmetric key that is later used for regular secure communication. Asymmetric cryptography is used only for authentication and symmetric key exchange and the actual communication is carried out using symmetric cryptography, because the symmetric algorithms are much more efficient compared to the asymmetric algorithms. Figure 7.6 shows general PKI based communication which is applicable in our proposed WirelessHART wired security specifications.

A device that leaves the network should not be able to communicate with the WirelessHART devices using its private key and a certificate. The SM

publishes a Certificate Revocation List (CRL) as a service for the other wired devices. This list contains the revoked certificates of the devices that are no longer the part of WirelessHART network. The CRL also contains the certificates that are expired and the certificates whose corresponding private keys are compromised.

Many PKI based protocols are available such as Secure Socket Layer (SSL) [14], Mutual Authentication [15], etc. In our implementation we use the Metro Web Services architecture [16] to implement the SM and use its security architecture to secure the connection between the SM and the other wired devices in the WirelessHART network. See Section 5 for details.

7.5 Security Manager Implementation

The SM and the NM are two separate entities that may be located at two different locations and the software and hardware used for both may vary. The WirelessHART standard does not specify the interface between the two. So the SM implementation should be platform independent and interoperable with the technologies used to implement the Network Managers and other network entities.

There are different technologies to achieve interoperability e.g. CORBA, RMI over IIOP, Web Services etc. Web service is a widely used technology for exporting the functionalities of an application to the other applications located on a local or remote machines. In the past security was a serious issue in web services because the two different technologies at two ends should follow the same security protocol; this was hard to achieve until Sun Microsystems and Microsoft worked closely to overcome interoperability issues. As a result of this coordination, Sun Microsystems (Glassfish community) developed a web services stack named Metro [16].

We develop our own CA as a web service using Metro Web services, Java Cryptographic Extension (JCE), and Bouncy Castle API [17]. Metro 1.4 provides built-in capabilities to use many asymmetric protocols such as Mutual Certificate Security, TLS/SSL, etc. We rely on JCE for key generation, hash calculation, and secure storage. All symmetric keys are stored in secure Java Cryptographic Extension Key Store (JCEKS) and all the certificates and private keys are secured in a Java Key Stores (JKS). We use Bouncy Castle APIs for generating and signing X509 certificates. For key database we use the Derby driver and host it in GlassFish server v.2. Metro built-in security services use our trust stores and key stores and create a secure session between the SM and

other devices.

Our second web service called KeyManager provides the KMS for the wireless part of the WirelessHART network. It provides key generation, retrieval, storage, renewal, and revocation mechanisms. We also develop a NM [1] and security administration applications; the latter is a complete web based GUI that provides interfaces for key store creation and exploration; certificate generation, signing, and revocation; Join key creation and device registration; and backup of security keys and key database.

7.6 Security Manager Evaluation

The WirelessHART standard does not provide a complete specification and design of the SM. However, the SM is a *mandatory* device in WirelessHART networks.In this section, we evaluate the implementation of our SM that we have designed and implemented from scratch. Our implementation in itself is a verification and evaluation of the design.

7.6.1 Performance Evaluation

The standard forbids that the SM directly communicates with the sensor devices but the SM interacts with the NM that in turn interacts with the devices. Hence the sensor nodes' low processing power, memory constraints, limited battery life, etc. do not affect the design and implementation of the SM. However, the response time or latency between the NM and SM impacts the performance of the rest of the WirelessHART network. We test different cryptographic algorithms for key generation and measure the latency between the SM and the NM. We start measuring the latency from the NM's key request function to the SM and back to the NM. We deploy SM and NM on two different machines and connect them through a direct link. We get an average response time of 71ms which is far less than different reply time requirements in the WirelessHART standard such as maxReplyTime (30s), JoinReplyTimeout (default is Keep-Alive), BcastReplyTime (60s), etc. [18]. We also calculate the standard deviation to explain the variation in the latencies. Figure 7.7 shows the average latencies and standard deviations of different hash algorithm and SHA1PRNS as Pseudo-Random Number Generator (PRNG). Figure 7.7 depicts that the SHA-1 and SHA-256 algorithms have a lower average latency

[1] Our NM is not a fully functional application rather we only defined the functions needed to interact with the SM.

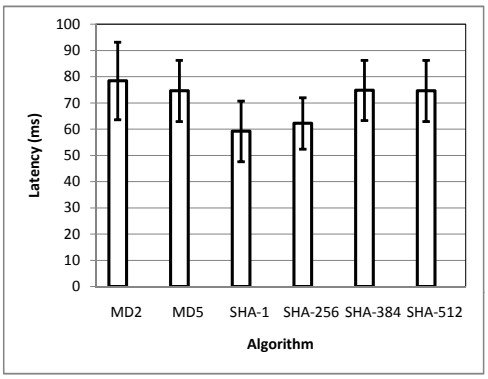


Figure 7.7: Latency between the NM and the SM with Key generation and Hashing

than MD2 and MD5. Moreover, there are no successful attacks against the SHA algorithms which implies that SHA algorithms are more secure than MD algorithms. In case of SHA-256, the deviation in latencies is less than the other algorithms. Based on these results our recommendation is to use the SHA-256 hash algorithm with the SHA1PRNG to generate secure random keys as it is fast, secure, and has relatively constant behavior.

When the NM requests the SM for the pre-generated keys to encrypt/decrypt normal messages the average latency decreases to 19ms. This is because the keys are already generated and stored in a secure storage. Figure 7.8 shows latencies for 10 executions when the NM requests a pre-generated key from the SM.

7.6.2 Security Analysis

Our SM completes the security requirements of the WirelessHART network by providing the security in both the wired and the wireless parts of the network. The wireless portion uses the secure and recommended AES algorithm to provide security against both insiders (end-to-end security at the Network layer) and outsiders (per-hop security at the DDL). Also, the reliability and the availability services are ensured in the wireless portion using the FHSS, path redundancy (using graph routing), and time diversity (using the Time Division Multiple Access).

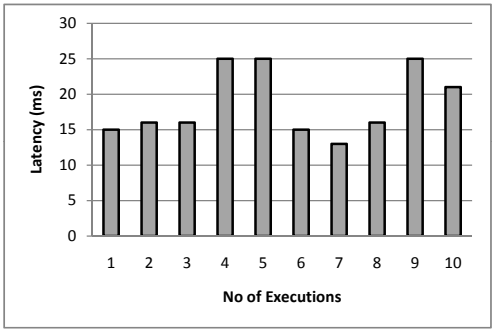


Figure 7.8: Latency between the NM and the SM when requesting pre-generated keys

In our implementation the KMS provides secure key storage, random and secure key generation, and reliable and secure key distribution, revocation, and renewal. The wired portion is secured using public key cryptography with digital certificates and local highly trusted CA.

However, the overall WirelessHART KMS does not provide defense-in-depth because of the standard's inherited limitations for key distribution. The security of some keys is interdependent and if one is compromised the others will be revealed as well. For example, the Join keyed session is used to renew unicast session keys and the Unicast-NM key is used to renew the Join key; if one of the keys is revealed the other will be compromised as well. The same is true for the Network key and the Broadcast-NM key.

In the wireless part, the WirelessHART standard only provides communication security; whereas the protection mechanisms for stored secrets are not specified in the standard. Also, the standard does not provide secure multicast communication among the Field devices. The authorization and accounting security services are also not specified in the standard. The current release (HART 7.1) of the WirelessHART standard only supports symmetric cryptography in the wireless medium. The lack of asymmetric cryptography makes the standard unable to provide certain security services such as strong authentication, non-repudiation, etc.

7.7 Related Work

WirelessHART is a recent standard. To the best of our knowledge we are the first to specify, design, and implement a SM for WirelessHART networks. However, WirelessHART is not the only secure standardized solution for the industrial process automation. Other wireless technologies such as Bluetooth, ZigBee, ISA100.11a, WibRee, etc may be used for industrial automation but with limitations.

Security is optional in Bluetooth and is based on weak E_o stream cipher algorithm, has improper key management, prone to impersonation attacks, no application level security, etc. [19]. However security is not the only reason which makes Bluetooth unsuitable for industrial applications. Other limitations such as limited battery life, maximum 8 devices in the network, star topology, etc. [20] also make Bluetooth impertinent for the WSN especially in an industrial realm. Wibree (Bluetooth Low Energy Technology) is more power efficient than standard Bluetooth but still has the other Bluetooth limitations. Industrial applications have strict security and reliability requirements. On one hand ZigBee is a better choice than Bluetooth and Wibree as it is secured using 128 bit AES algorithm, has user defined security at application layer [20], is energy efficient, based on mesh topology, and relatively fast. On the other hand, no frequency diversity, no path redundancy, and lack of robustness make ZigBee less reliable and make it inappropriate for the industrial process automation [21].

ISA100.11a [22] is another proposed standard for the industrial applications but it is not approved as a standard yet. However, the best of WirelessHART features and the additional claimed features such as asymmetric cryptography, object-based application layer security, security management data structures, etc. make ISA100.11a. a suitable standard for the industrial process automation and control systems [22]. But we cannot see the actual comparison unless or until ISA100.11a releases.

Among the available standardizes solution WirelessHART is the most suitable protocol for the industrial process automation. The usage of feature such as frequency diversity, path diversity, time diversity, etc. make WirelessHART a reliable industrial standard.

7.8 Conclusions and Future Work

The lack of SM specification in the WirelessHART standard gives rise to ambiguities about the capabilities and design of SM. We have developed our SM from scratch. After understanding the whole WirelessHART standard we have elucidated comprehensive specifications of the SM. We have converted the specifications into an architectural design that models both the internals of SM and its interaction with the other network devices. We have developed the SM keeping in mind that the SM is a standalone device that interacts with other network devices that may have been developed for different platforms and with different programming languages. Lastly we have evaluated our SM in term of efficiency to meet overall WirelessHART timing requirements. Our evaluation shows that the SM fulfills the timing requirements of WirelessHART.

Our SM fully complies with the WirelessHART standard and meets all security requirements mentioned in the standard. The proposed solutions to secure the wired part of the network are strong enough to provide all core security services including authentication, confidentiality, integrity, authorization, and non-repudiation. However, the inherited limitations of the WirelessHART standard such as lacking asymmetric cryptography do not allow us to provide some security services such as strong authentication, non-repudiation, etc. in the wireless part.

The WirelessHART standard can be extended with asymmetric cryptography [23] using the reserved security bits in the security sub-layer [18].

Acknowledgment

This work has been performed within the SICS Center for Networked Systems funded by VINNOVA, SSF, KKS, ABB, Ericsson, Saab Systems, TeliaSonera and T2Data. This work has been partially supported by CONET, the Cooperating Objects Network of Excellence.

Bibliography

[1] *IEC approves WirelessHART*. Control Engineering, Vol. 55 Issue 10 Pages 34-34, October 2008.

[2] C. Leung. Evaluation of the undetected error probability of single parity-check product codes. *Communications, IEEE Transactions on*, 31(2):250–253, 1983.

[3] Jianping Song, Song Han, Aloysius K. Mok, Deji Chen, Mike Lucas, and Mark Nixon. Wirelesshart: Applying wireless technology in real-time industrial process control. *Real-Time and Embedded Technology and Applications Symposium, 2008(RTAS-08)*, pages 377 – 386, April 2008.

[4] *WirelessHART Device Specification,HCF_SPEC-290, Revision 1.1*. HART Communication Foundation, May 2008.

[5] Morris Dworkin. *Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality*. NIST Special Publication 800-38C, May 2004.

[6] J. Lopez and J. Zhou. Wireless Sensor Network Security. Volume 1 of Cryptology and information security series, 2008.

[7] *Wireless Command Specification, HCF_SPEC-155, Revision 1.1*. HART Communication Foundation, May 2008.

[8] P. Resnick. Internet Message Format. RFC 2822 (Proposed Standard), April 2001. Obsoleted by RFC 5322, updated by RFCs 5335, 5336.

[9] GH Larsen. Software: Man in the middle. *Datamation*, 19(11):61–66, 1973.

[10] J. Nechvatal, E. Barker, L. Bassham, W. Burr, M. Dworkin, J. Foti, and E. Roback. Report on the development of the Advanced Encryption Standard (AES). *Journal of Research-National Institute of Standards and Technology*, 106(3):511–576, 2001.

[11] A. Bogdanov. Multiple-differential side-channel collision attacks on AES. *Cryptographic Hardware and Embedded Systems–CHES 2008*, pages 30–44, 2008.

[12] M. Jakobsson, E. Shriver, B.K. Hillyer, and A. Juels. A practical secure physical random bit generator. In *Proceedings of the 5th ACM Conference on Computer and Communications Security*, pages 103–111. ACM, 1998.

[13] M. Hendry. *Smart card security and applications*. Artech House Publishers, 2001.

[14] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.1. RFC 4346 (Proposed Standard), April 2006. Obsoleted by RFC 5246, updated by RFCs 4366, 4680, 4681, 5746, 6176.

[15] S. Wakid. Entity Authentication Using Public Key Cryptography. Technical report, NATIONAL INST OF STANDARDS AND TECHNOLOGY GAITHERSBURG MD, 1997.

[16] GlassFish Community. Metro Users Guide. Technical report, 2009.

[17] 675 N. 1st Street Suite 1200 San Jose CA 95112 USA. The Legion Of The Bouncy Castle. Technical report, 2009.

[18] *Network Management Specification,HCF_SPEC-085, Revision 1.1*. HART Communication Foundation, May 2008.

[19] C. Gehrmann. Bluetooth security. 2004.

[20] N. Baker. ZigBee and Bluetooth strengths and weaknesses for industrial applications. *computing & control engineering journal*, 16(2):20–25, 2005.

[21] Tomas Lennvall, Stefan Svensson, and Fredrik Heklan. A comparison of wirelesshart and zigbee for industrial applications. *IEEE International Workshop on Factory Communication Systems*, pages 85–88, May 2008.

[22] ISA. *ISA-100.11a-2009: Wireless systems for industrial automation: Process control and related applications*. ISA Standards, 67 Alexander Drive Research Triangle Park, NC 27709 USA, May 2009.

[23] W. Hu, P. Corke, W. Shih, and L. Overs. secfleck: A public key technology platform for wireless sensor networks. In *EWSN 2009*, Cork, Ireland, February 2009.

Chapter 8

Paper C: Securing Communication in 6LoWPAN with Compressed IPsec

Shahid Raza, Simon Duquennoy, Tony Chung, Dogan Yazar, Thiemo Voigt, Utz Roedig
7th IEEE International Conference on Distributed Computing in Sensor Systems (IEEE DCOSS '11), 27-29 June 2011, Barcelona, Spain.

Abstract

Real-world deployments of wireless sensor networks (WSNs) require secure communication. It is important that a receiver is able to verify that sensor data was generated by trusted nodes. It may also be necessary to encrypt sensor data in transit. Recently, WSNs and traditional IP networks are more tightly integrated using IPv6 and 6LoWPAN. Available IPv6 protocol stacks can use IPsec to secure data exchange. Thus, it is desirable to extend 6LoWPAN such that IPsec communication with IPv6 nodes is possible. It is beneficial to use IPsec because the existing end-points on the Internet do not need to be modified to communicate securely with the WSN. Moreover, using IPsec, true end-to-end security is implemented and the need for a trustworthy gateway is removed.

In this paper we provide End-to-End (E2E) secure communication between IP enabled sensor networks and the traditional Internet. This is the first compressed lightweight design, implementation, and evaluation of 6LoWPAN extension for IPsec. Our extension supports both IPsec's Authentication Header (AH) and Encapsulation Security Payload (ESP). Thus, communication endpoints are able to authenticate, encrypt and check the integrity of messages using standardized and established IPv6 mechanisms.

8.1 Introduction

Wireless Sensor Networks can be tightly integrated with existing IP based infrastructures using IPv6 over Low Power Wireless Personal Area Networks (6LoWPAN). Sensor nodes using 6LoWPAN can directly communicate with IPv6 enabled hosts and, for example, sensor data processing can be performed by standard servers. Thus, 6LoWPAN greatly simplifies operation and integration of WSNs in existing IT infrastructures.

Real-world deployments of wireless sensor networks (WSNs) require secure communication. For instance, in a smart meter application, the provider and the meters would need to authenticate one another and encryption would be desirable to ensure data confidentiality. IPv6 hosts in the Internet support by default IPsec for secure communication. Therefore, if data flows between IPv6 hosts and 6LoWPAN sensor nodes it is desirable to take advantage of existing capabilities and to secure traffic using IPsec. Thus, we propose to add IPsec support to 6LoWPAN as illustrated by Figure 8.1.

IPsec defines an Authentication Header (AH) and an Encapsulating Security Payload (ESP). The AH provides data integrity and authentication while ESP provides data confidentiality, integrity and authentication. Either AH, ESP or both can be used to secure IPv6 packets in transit. It is up to the application to specify which security services are required. 6LoWPAN uses header compression techniques to ensure that the large IPv6 and transport-layer headers (UDP/TCP) are reduced. By supporting IPsec's AH and ESP, additional IPv6 extension headers have to be included in each datagram. Thus, it is important to ensure that compression techniques are as well applied to these extension headers.

Independent of the achieved compression rates of AH and ESP it is obvious that IPsec support in 6LoWPAN will increase packet sizes as additional headers must be included. Note, however, that by using IPsec we do not need to use existing 802.15.4 link-layer security mechanisms which in turn frees some header space.

The main contributions of this paper are:

- *6LoWPAN-IPsec Specification:* We give a specification of IPsec for 6LoWPAN including definitions for AH and ESP extension headers. Prior to this work no specification for IPsec in the context of 6LoWPAN existed;
- *6LoWPAN-IPsec Implementation:* We present the first implementation of IPsec for 6LoWPAN networks. We show that it is practical and feasible to secure WSN communication using IPsec;

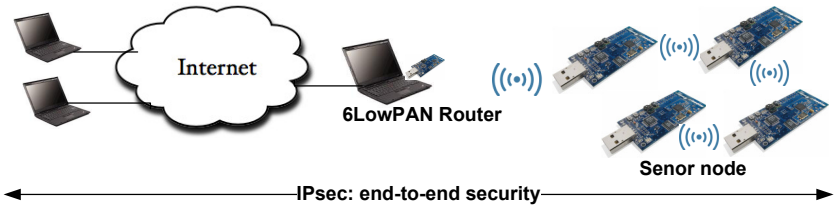


Figure 8.1: We propose to use IPsec to secure the communication between sensor nodes in 6LoWPANs and hosts in an IPv6-enabled Internet. IPsec provides E2E security using existing methods and infrastructures.

- *6LoWPAN-IPsec Evaluation:* We evaluate the performance of our IPsec 6LoWPAN implementation in terms of code size, packet overheads and communication performance. Our results show that overheads are comparable to overheads of generally employed 802.15.4 link-layer security while offering the benefit of true E2E security.

The paper proceeds by discussing related work followed by a further motivating of using of IPsec. Then we present background knowledge on IPv6, IPsec and 6LoWPAN. Section 8.5 describes our proposed integration of 6LoWPAN and IPsec. After a thorough experimental evaluation of the performance of our IPsec implementation, we conclude the paper.

8.2 Related Work

Message authentication and encryption in WSNs is generally performed using well known cryptographic mechanisms such as block ciphers as part of standards-based protocols such as IEEE 802.15.4. However, these mechanisms are difficult to implement on resource constrained sensor nodes as cryptographic mechanisms can be expensive in terms of code size and processing speed. Furthermore, it is necessary to distribute and maintain keys and it is difficult to implement efficient key distribution protocols for resource constrained sensor nodes. Thus, a lot of research work aims to reduce complexity of cryptographic mechanisms, for example, TinyEEC [1] and NanoEEC [2], or to simplify key distribution, for example, Liu and Ning's proposal for pairwise key predistribution [3] and DHB-KEY [4]. These improvements make cryptographic mechanisms in the context of WSNs more viable but an important

issue remains: a standardized way of implementing security services is missing and for each deployment unique customized solutions are created. Using the standardized 6LoWPAN as a vehicle to implement security services in form of the proven and standardized IPsec offers a solution to this problem. IPsec is currently available as part of some WSN products, but does not provide a full E2E security solution. One such example is the ArchRock PhyNET [5] that applies IPsec in tunnel mode between the gateway and Internet hosts, but still relies on link-layer security within the sensor network thus breaking true E2E assurance. We are not aware of a complete E2E implementation nor an evaluation of a working system which we present in this paper.

The IEEE 802.15.4 [6] standard defines Advanced Encryption Standard (AES) message encryption and authentication on the link-layer. The cryptographic algorithms could be executed by specialized hardware within the transceiver chip. However, link-layer security only protects messages while they travel from one hop to the next as we discuss in Section 8.3. Wood and Stankovic [7] as well as Hu et al. [8] have demonstrated performance gains when security operations are performed in hardware. We expect similar performance gains when IPsec operations are implemented in hardware. Granjal et al. argue that IPsec is generally feasible in the context of WSN [9]. In their study they analyze the execution times and memory requirements of cryptographic algorithms. Their work only discusses performance of cryptographic algorithms but does not describe how IPsec is actually integrated with 6LoWPAN. In our work, we implement 6LoWPAN with compressed IPsec and we analyze the performance of the overall system, not only the performance of the cryptographic algorithms.

8.3 Securing WSN Communications

Researchers have unanimous consensus that security is very important for the future IP based WSN and its integration with the traditional Internet. IPv6 with potentially unlimited address space is the obvious choice for these networks [10]. However, security support for IP-based low power networks is still an open issue, as mentioned in the 6LoWPAN specifications [11, 12]. Actually, security can be guaranteed at different layers of the IP protocol stack, resulting in solutions with various compromises..

6LoWPAN today relies on the IEEE 802.15.4 (referred to as 802.15.4 in the following) link-layer which provides data encryption and integrity check-

ing. This solution is appealing since it is independent of the network protocols and is currently supported by the hardware of 802.15.4 radio chips. However, such link-layer mechanism only ensures *hop-by-hop* security where every node in the communication path (including the 6LoWPAN gateway) has to be be trusted, and where neither host authentication nor key management is supported. Furthermore, messages leaving the sensor network and continuing to travel on an IP network are not protected by link-layer security mechanisms.

End-to-end security can be provided by the widely used Transport Layer Security (TLS) standard. By operating between the transport-layer and the application-layer, it guarantees security between applications, includes a key exchange mechanism and provides authentication between Internet hosts in addition to confidentiality and integrity. As a counterpart, TLS can only be used over TCP, which is rarely used in wireless sensor networks. An adaptation of TLS for UDP called DTLS is available, but it is not widely used.

The IPsec protocol suite, mandated by IPv6, provides end-to-end security for any IP communication [13]. Like TLS and unlike hop-by-hop solutions, it includes a key exchange mechanism and provides authentication in addition to confidentiality and integrity. By operating at the network-layer, it can be used with any transport protocols, including potential future ones. Furthermore, it ensures the confidentiality and integrity of the transport-layer headers (as well as the integrity of IP headers), which cannot be done with a higher-level solution like TLS. For these reasons, researchers [9, 14, 15] and 6LoWPAN standardizations groups [12] consider IPsec a potential security solution for IP based WSN.

In this paper we show that compressed IPsec is a sensible and viable choice for 6LoWPANs. The key advantage of using IPsec in WSN is that we achieve *end-to-end* IP based communication between a sensor device and Internet hosts. When using IPsec, the IEEE 802.15.4 security features can be disabled as security services are provided in the IP layer. We show later that when comparing link-layer security with IPsec, packet sizes are similar.

8.4 Background

In this section we briefly outline core functionality of IPv6, IPsec and 6LoWPAN that is relevant for the work presented in this paper. For more information we refer to the corresponding RFCs: RFC2460 [16], RFC4301 [17] and RFC4944 [12].

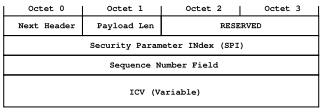


Figure 8.2: IPsec AH headers

8.4.1 IPv6 and IPsec

With the vision of the Internet of Things and Smart Objects all kind of physical devices such as wireless sensors are expected to be connected to the Internet via IP [10]. This requires the use of IPv6 [16], a new version of the Internet Protocol that increases the address size from 32 bits to 128 bits. Besides the increased address space IPv6 provides in comparison to IPv4 a simplified header format, improved support for extensions and options, flow labeling capability and authentication and privacy capabilities.

Authentication and privacy in IPv6 is provided by IPsec [17]. IPsec defines a set of protocols for securing IP communication: the security protocols Authentication Header (AH) [18] and Encapsulating Security Payload (ESP) [19], the algorithms for authentication and encryption, key exchange mechanisms and so called security associations (SA) [17]. An SA specifies how a particular IP flow should be treated in terms of security. To establish SAs, IPSec standard specifies both pre-shared key and Internet Key Exchange (IKE) protocol. This means that every node on IPv6 enabled conventional Internet supports pre-shared key. In other words an implementation with pre-shared based SA establishment works with any IPv6 node on Internet. Also, IKE uses asymmetric cryptography that is assumed to be heavy weight for small sensor nodes. However, it would be worth investigating IKE with ECC for 6LoWPANs; we intend to do it in future.

The task of the AH is to provide connectionless integrity and data origin authentication for IP datagrams and protection against replays. A keyed Message Authentication Code (MAC) is used to produce authentication data. The MAC is applied to the IP header, AH header and IP payload. The authentication header is shown in Figure 8.2. All hosts must support at least the hash-based message authentication code algorithm AES-XCBC-MAC-96 [20] to calculate authentication data that has a size of 12 bytes. Thus, as shown in Figure 8.2, a basic AH header has a size of 24 bytes.

ESP [19] provides origin authenticity, integrity, and confidentiality protection of IP packets. ESP is used to encrypt the payload of an IP packet but in

contrast to AH it does not secure the IP header. If ESP is applied the IP header is followed by the ESP IP extension header which contains the encrypted payload. ESP includes an SPI that identifies the SA used, a sequence number to prevent replay attacks, the encrypted payload, padding which may be required by some block ciphers, a reference to the next header and optional authentication data. Encryption in ESP includes Payload Data, Padding, Pad Length and Next Header;Authentication, if selected, includes all header fields in the ESP. If we assume mandatory AES-CBC as encryption algorithm an ESP with perfect block alignment will have an overhead of 18 bytes (10 bytes for ESP and 8 bytes for Initialization Vector). If additional authentication using AES-XCBC-MAC-96 is used the ESP overhead is 30 bytes, as the minimum length of AES-XCBC-MAC-96 is 12 bytes.

The protocols AH and ESP support two different modes: transport mode and tunnel mode. In transport mode IP header and payload are directly secured as previously described. In tunnel mode, a new IP header is placed in front of the original IP packet and security functions are applied to the encapsulated (tunneled) IP packet. In the context of 6LoWPAN tunnel mode seems not practical as the additional headers would further increase the packet size.

8.4.2 6LoWPAN

6LoWPAN [12] aims at integrating existing IP based infrastructures and sensor networks by specifying how IPv6 packets are to be transmitted over an IEEE 802.15.4 network. The maximum physical-layer packet size of 802.15.4 packet is 127 bytes and the maximum frame header size is 25 bytes. An IPv6 packet has therefore to fit in 102 bytes. Given that packet headers of a packet would already consume 48 bytes of the available 102 bytes it is obvious that header compression mechanisms are an essential component of the 6LoWPAN standard.

HC13[21] proposes context aware header compression mechanisms: the LOWPAN_IPHC (referred to as IPHC in the following) encoding for IPv6 header compression and the LOWPAN_NHC (referred to as NHC in the following) encoding for the next header compression. The IPHC header is shown in Figure 8.3.

For efficient IPv6 header compression, IPHC removes safely IPv6 header fields that are implicitly known to all nodes in the 6LoWPAN network. The IPHC has a length of 2 byte of which 13 bits are used for header compression as shown in Figure 8.3. Uncompressed IPv6 header fields follow directly the IPHC encoding in the same order as they would appear in the normal IPv6

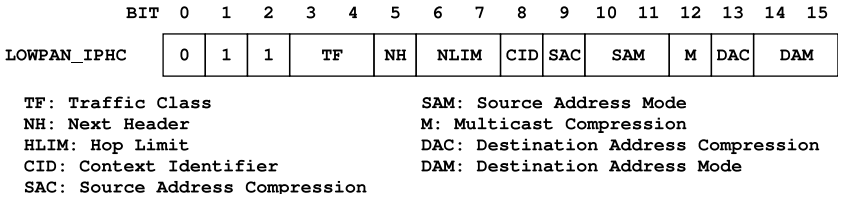


Figure 8.3: The LOWPAN_IPHC Header.

header. In a multihop scenario IPHC can compress the IPv6 header to 7 bytes The NH field in the IPHC indicates whether the next header following the basic IPv6 header is encoded. If NH is 1, NHC is used to compress the next header. 6LoWPAN specifies that the size of NHC should be multiple of octets, usually 1 byte where first variable length bits represents a NHC ID and the remaining bits are used to encode/compress headers. 6LoWPAN already defines NHC for UDP and IP Extension Header [21].

8.5 6LoWPAN and IPsec

IPsec requires header compression to keep packet sizes reasonable in 6LoWPAN. Unfortunately, there are no header encodings specified for AH and ESP extension headers. In this section we therefore propose these extension header encodings. We evaluate our savings in terms of packet size later in Section 8.6. At the end of this section, we also discuss further improvements that would be possible by small, standard-compliant modifications of the end hosts where there is need for cryptographic algorithms that could handle 6LoWPAN UDP compression.

8.5.1 LOWPAN_NHC Extension Header Encoding

As previously described, HC13 defines context aware header compression using IPHC for IP header compression and NHC for the next header compression. The already defined NHC encoding form for IP extension headers can be used to encode AH and ESP extension headers. NHC encodings for the IPv6 Extension Headers consist of a NHC octet where three bits (bits 4,5,6) are used to encode the IPv6 Extension Header ID (EID). This NHC_EH encoding for extension headers is shown in Figure 8.4.

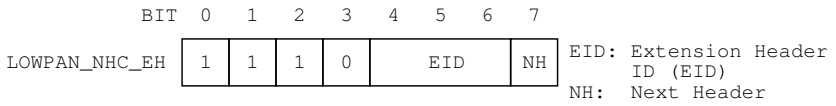


Figure 8.4: LOWPAN_NHC_EH: NHC encoding for IPv6 Extension Header

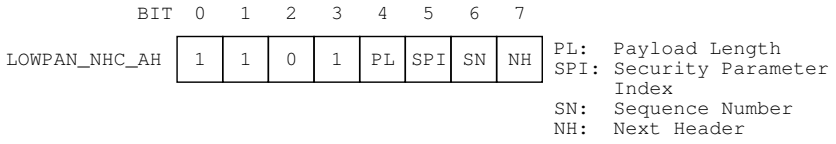


Figure 8.5: NHC_AH: NHC encoding for IPv6 Authentication Header

Out of eight possible values for the EID, six are specified by the HC13 draft. The remaining two slots (101 and 110) are currently reserved. We propose to use the two free slots to encode AH and ESP. Also, it is necessary to set the last bit in IPv6 extension header encoding to 1 to specify that the next header (AH or ESP) is encoded as well using NHC.

8.5.2 LOWPAN_NHC_AH Encoding

We define the NHC encoding for the AH. Our proposed NHC for AH is shown in Figure 8.5.

We describe the function of each header field:

- The first four bits in the NHC_AH represent the NHC ID we define for AH, and are set to 1101. These are needed to comply with 6loWPAN standard.
- PL: If 0, the payload lengths is omitted. This length can be obtained from the SPI value because the length of the authenticating data depend on the algorithm used and are fixed for any input size. If 1, the length is carried inline after the NHC_AH header
- SPI: If 0, the default SPI for the sensor network is used and the SPI field is omitted. We set the default SPI value to 1. This does not mean that all

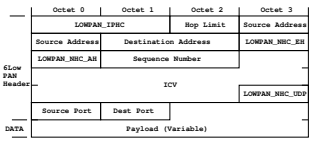


Figure 8.6: Example of a compressed IPv6/UDP packet using AH

nodes use the same security association (SA), but that every node has its own preferred SA, identified by SPI 1. If 1, the SPI is carried inline

- SN: If 0, a 16 bit sequence number is used and the left most 16 bits are assumed to be zero. If 1, all 32 bits of the sequence number are carried inline.

- NH: If 0, the next header field in AH will be used to specify the next header and it is carried inline. If 1, the next header field in AH is skipped. The next header will be encoded using NHC.

The minimum length of a standard AH supporting the mandatory HMAC-SHA1-96 is 24 bytes. After optimal compression we obtain a header size of 16 bytes. Figure 8.6 shows compressed IPv6/UDP packet secured with AH with HMAC-SHA1-96.

8.5.3 LOWPAN_NHC_ESP Encoding

Also the NHC encoding for ESP encodes the security parameter index, the sequence number, the next header fields and the NHC ID for ESP. In the case of ESP, we propose 1110 as NHC ID while we propose 1101 as NHC for AH as shown in Figure 8.6. Due to space limitation, we do not detail these encoding for ESP which are available in full details in a technical report [22].

Recall that the minimum ESP overhead without authentication, AES-CBC and perfect block alignment is 18 bytes. After optimal compression this header overhead is reduced to 12 bytes. ESP with authentication (HMAC-SHA1-96) has an overhead of 30 bytes which is reduced to 24 bytes using the outlined ESP compression.

8.5.4 Combined Usage of AH and ESP

It is possible to use AH and ESP in combination; obviously the defined AH and ESP compression headers can be used in succession. However, it is more efficient in terms of header sizes to use ESP with authentication option than to apply AH and ESP to a packet. As packet sizes are important in the context of WSNs we expect that this IPsec option will not be used in practice.

8.5.5 End Host Requirement

AH capable 6LoWPAN nodes can directly communicate with unmodified IPsec hosts on conventional Internet. When ESP is used 6LoWPAN nodes can as well communicate directly with unmodified IPsec hosts. However, if ESP is used it is not possible to compress upper layer headers such as UDP. A 6LoWPAN gateway between sensor network and IP network cannot access and expand the encrypted UDP header. To enable UDP compression with ESP we need to specify a new encryption algorithm for ESP which is able to perform UDP header compression and encryption. Again, if this optimization is used IPsec hosts must include and support this encryption protocol.

8.6 Evaluation and Results

In this section we quantify performance of the proposed IPSec extensions for 6LowPAN. After describing our implementation and experimental setup, we evaluate the impact of IPsec in terms of memory footprint, packet size, energy consumption and performances under different configurations.

8.6.1 Implementation and Experimental Setup

We implement IPsec AH and ESP for the Contiki operating system [23]. The implementation required the modification of the existing Contiki μIP stack which already provides 6LoWPAN functionality. The Contiki μIP stack is used on the sensor nodes and on a so called soft bridge connecting WSN and the Internet. In addition to the IPsec protocol, we implement the IPsec/6LoWPAN compression mechanisms as outlined in the previous section. We support the NHC_EH, NHC_AH, and NHC_ESP encodings (see Section 8.5) at the SICSLoWPAN layer, the 6LoWPAN component of the μIP stack.

We use the SHA1 and AES implementations from MIRACL [24], an open source library, and implement all cryptographic modes of operation needed for

System	ROM (kB)		RAM (kB)	
	overall	diff	overall	diff
Without IPsec	32.9	–	8.0	–
AH with HMAC-SHA1-96	36.8	3.9	9.1	1.1
AH with XCBC-MAC-96	38.4	5.5	8.5	0.5
ESP with AES-CBC	41.4	8.5	8.3	0.3
ESP with AES-CTR	39.8	6.9	9.1	0.3
ESP with AES-XCBC-MAC-96	39.8	6.9	8.3	0.3
ESP with AES-CBC + AES-XCBC-MAC-96	41.9	9.0	8.3	0.3
ESP with AES-CBC + AES-XCBC-MAC-96	41.9	9.0	8.3	0.3

Table 8.1: Memory footprints show that AH and ESP consumes just 3.9kB and 9kB for mandatory IPsec algorithms

Service	Uncompressed IPsec		Compressed IPsec		802.15.4	
	Mode	Bytes	Mode	Bytes	Mode	Bytes
AH Authentication	HMAC-SHA1-96	24	HMAC-SHA1-96	16	AES-CBC-MAC-96	12
ESP Encryption	AES-CBC	18	AES-CBC	12	AES-CTR	5
ESP Encryption and Authentication	AES-CBC and HMAC-SHA1-96	30	AES-CBC and HMAC-SHA1-96	24	AES-CCM-128	21

Table 8.2: With compressed IPsec, packet sizes are similar to 802.15.4 while IPsec provides end-to-end security.

authentication and encryption in IPsec. For AH, we implement the mandatory HMAC-SHA1-96 and AES-XCBC-MAC-96. For ESP, we implement the mandatory AES-CBC for encryption and HMAC-SHA1-96 for authentication. Additionally, in ESP, we implement the optional AES-CTR for encryption and AES-XCBC-MAC-96 for authentication. Our Contiki IPsec 6LoWPAN implementation uses pre-shared keys to establish SAs which work with any IPv6 node on Internet as a pre-shared mechanism is mandatory in IPsec. Manual key distribution, however, is currently also used for traditional 802.15.4 link-layer security.

Our evaluation setup shown in Figure 8.1 consists of four Tmote Sky [25] sensor nodes, a 6LoWPAN soft bridge (implemented by a fifth Tmote) nd a Linux machine running Ubuntu OS with IPsec enabled. The four sensor nodes on the right side in Figure 8.1 form a multihop network. They execute a single application that listens to a fixed UDP port. When a packet is received, it is processed by the 6LoWPAN layer, interpreted by the IPsec layer and by μIP. Then its payload is forwarded to the application. As a reply, a new datagram of the same size is sent back, following the opposite process. Thus, IPsec is used to secure the end-to-end (E2E) communication between the sensor node and the Internet host. To avoid the delay of a duty-cycled MAC layer, we use Contiki's NullMAC in the experiments and hence all nodes keep their radio turned on all the time.

8.6.2 Memory footprint

We measure the ROM and RAM footprint of our IPsec implementation. Table 8.1 compares IPsec AH and IPsec ESP using the multiple modes of operation we implemented. The footprints are compared with a reference Contiki system including uIP and SICSLoWPAN.

The ROM footprint overhead ranges from 3.8 kB (AH with HMAC-SHA1) to 9 kB (ESP with AES-CBC + AES-XCBC-MAC). This always keeps the system footprint under 48 kB, the Flash ROM size of the Tmote Sky. It is worth mentioning that unlike AES-CBC, the AES-CTR mode of operation only relies on AES encryption. Thus, the AES-CTR + AES-XCBC-MAC-96 configuration can be implemented without AES decryption, resulting in a particularly low memory footprint.

The RAM footprint is calculated as the sum of the global data and the runtime stack usage that we measure by running Contiki in the MSPSim emulator [26]. With an additional footprint of 1.1 kB, the AH HMAC-SHA1 configuration is the most RAM-consuming configuration. When using other

modes of operation, the RAM usage lies between only 0.3 and 0.5 kB. These results show that both IPsec AH and ESP can be embedded in constrained devices while leaving space for applications.

8.6.3 Packet Overhead Comparison

Currently WSN communication is secured using 802.15.4 link-layer security. This security mechanism can only provide hop-by-hop security and, in contrast to IPsec, lacks the ability to provide proper E2E security. Nevertheless, we provide here a comparison of packet overheads between 802.15.4 link-layer security and IPsec security. Table 8.2 summarizes the packet overhead when using uncompressed IPsec, compressed IPsec and 802.15.4 link-layer security.

When using link-layer security, the packet overhead for the authentication scheme is exactly the length of the MAC. In IPsec when using AES-XCBC-MAC-96, the MAC has a length of 12 bytes. The additional AH header fields increase the overhead to 24 bytes. Thanks to the IPsec header compression we defined, this overhead is reduced to 16 bytes. The ability to provide E2E authentication with IPsec has hence a cost of 4 bytes compared to the 802.15.4 baseline which provides only hop-by-hop security.

If only message encryption is required, the 802.15.4 link-layer security provides AES-CTR which has a 5 bytes overhead. In comparison, IPsec with ESP and AES-CBC leads to an overhead of 18 bytes, reduced to 12 bytes thanks to header compression. Here, the ability to provide E2E encryption with IPsec has a cost of 7 bytes compared to the 802.15.4 baseline.

With AES-CCM-128, the overhead for 802.15.4 is 21 bytes while IPsec ESP has an overhead of 30 bytes, reduced to 24 bytes when using our 6LoWPAN compression extension. The ability to provide E2E encryption and authentication with IPsec has hence a cost of 3 bytes compared to the 802.15.4 baseline.

Moreover, when carrying large IP datagrams, link-layer fragmentation has to be used. With link-layer security, one pays the header overheads for every fragment. In contrast, the IPsec header is included only once for all the fragments of a single datagram. This means that as soon as two or more fragments are needed, IPsec offers a lower header overhead than 802.15.4 link-layer security.

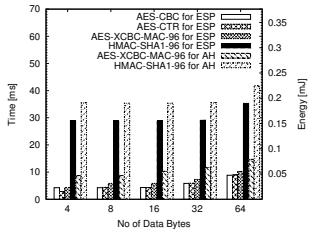


Figure 8.7: The comparison of our implemented algorithms shows that among the ones specified in the standards, AES-CBC and AES-XCBC-MAC-96 are the most efficient in terms of processing time and energy consumption. They are also mandatory and the most secure.

8.6.4 Performance of Cryptography

We evaluate the efficiency of the different cryptographic algorithms and modes supported by our IPsec implementation. Figure 8.7 details the performances and energy consumption for each mode of operation and depending on the size of the IP payload. The authentication algorithms are compared separately for AH and ESP: with AH the MAC is calculated over the IP header and payload packet, while in ESP the IP header is neither encrypted nor authenticated.

Our results show that for encryption, AES-CBC and AES-CTR have similar performances and energy consumption. Regarding authentication, the cost is as expected higher for AH than for ESP because of the processing of the 40 byte IP header. In all cases, the energy consumption has a fixed-cost and grows linearly with the data size. HMAC-SHA1-96 is not as efficient as other solutions because of its particularly high fixed-cost when data sizes are small.

The proposed standard for Cryptographic Suites for IPsec specifies that the future IPsec systems will use AES-CBC-128 for encryption and AES-XCBC-MAC-96 mode for authentication [27]. Figure 8.7 shows that these are also

8.6.5 System-wide Energy Overhead

Securing the Internet of Things has a cost in terms of added energy usage. We measure the energy overhead of the available security options on the Tmote Sky using Contiki's integrated energy estimator. We measure the total number of CPU ticks from the reception of the first fragment of a message, when starting link layer decryption. We stop counting when the link layer encryption of the

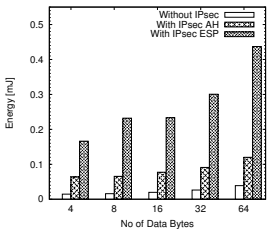


Figure 8.8: Node energy consumption is lower without IPsec and higher for ESP than for AH. Compared to other activities e.g. idle listening it is not significant.

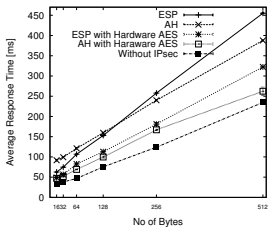
last packet is finished, but we ignore the network time between the packets. In total we the link layer processing, 6LoWPAN processing, μIP stack handling, and application-layer processing. These experiments are run with and without hardware support. For the

Figure 8.8 shows the energy consumption of Link Layer security only, IPsec using either AH or ESP, and without using any security. Since the variance of the 20 runs was very low, it is not not shown. The results show that ESP consumes more energy than AH; this is because for ESP we use both authentication and encryption. Although the energy consumption with IPsec is significantly higher than without IPsec we argue that this is negligible when compared to the consumption of typical radio chips. In the worst measured case, AH on 64 bytes, the energy consumed is around 0.5 mJ. The radio chip of the Tmote Sky consumes the same amount of energy after 8 ms of idle listening.

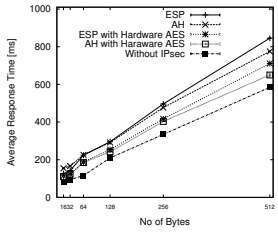
8.6.6 System-wide Response Time Overhead

We measure and evaluate the response time for different data sizes with IPsec and without IPsec. The response time is the time it takes to send a message from an IP connected Linux machine to a sensor node and to receive a response. We conduct experiments using a routing distance in the WSN ranging from 1 to 4 hops and for IP datagrams with a size ranging from 16 to 512 bytes. We execute every experiment 10 times.

Figure 8.9 shows the response time in dependency of the IP datagram size. When the datagram size is too large to fit a single 802.15.4 packet, the data are fragmented according to the 6LoWPAN standard. Consistently with the mirco-

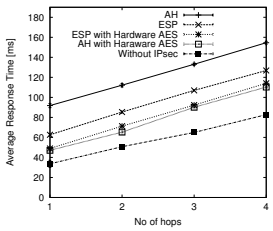


(a) Single Hop with different data sizes

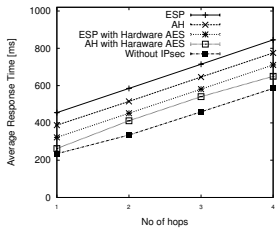


(b) Multi Hop (4) with different data sizes

Figure 8.9: Response time versus datagram size with AH, ESP and without IPsec. ESP is faster than AH for small datagrams because it does not process the 40 bytes IP header. AH is faster than ESP for large datagrams because it processes authentication but no encryption.



(a) Multi Hop with 16 bytes data size



(b) Multi Hop with 512 byte data size

Figure 8.10: Response time versus number of hops with AH, ESP and without IPsec. The overhead of IPsec is constant across a single hop and a multihop network.

benchmarks in Figure 8.7, the overhead of IPsec grows linearly with datagram sizes. We observe that for small sizes, ESP is faster than AH. This is because unlike AH, ESP does not process the full 40 bytes IP header. With larger sizes, AH is faster than ESP, because it ensures authentication only, while ESP authenticates plus encrypts and decrypts the messages.

Figure 8.10 shows the response times obtained in dependence of hop distance. For a given data size, we observe that the overhead of either AH or ESP is constant, whatever the number of hops. This is because, for the intermediate nodes, the cost of forwarding the data with and without IPsec is the same; the overhead is only due to computation on the end nodes. In the worst case (512 bytes), we measured an overhead of 261 ms.

8.6.7 Improvements Using Hardware Support

The efficiency of IPSec can be improved by employing cryptographic functions provided by sensor node hardware. For example, the CC2420 radio chip present on many sensor node platforms provides such functionality. To investigate possible improvements we extend our prototype implementation to use this hardware for the required AES computations. Figure 8.9 and Figure 8.10 show the impact of hardware supported cryptography on the achievable response time. In all cases hardware-based implementations are faster than pure software implementations. When processing 512 byte datagrams over a single hop the overhead of pure software AH is 65 % which decreases to 12 % with the help of the cryptographic coprocessor. For ESP the decrease ranges from 64 % to 37 %.

8.7 Conclusions and Future Work

WSNs will be an integral part of the Internet and IPv6 and 6LoWPAN are the protocol standards that are expected to be used in this context. IPsec is the standard method to secure Internet communication and we investigate if IPsec can be extended to sensor networks. Towards this end, we have presented the first IPsec specification and implementation for 6LoWPAN. We have extensively evaluated our implementation and demonstrated that it is possible and feasible to use compressed IPsec to secure communication between sensor nodes and hosts in the Internet.

To securely communicate with any IPv6 enabled node on the Internet preshared keys are sufficient but not very flexible. Therefore, we plan to investi-

gate if an automatic key exchange protocol for 6LoWPANs based on IPsec's Internet Key Exchange protocol (IKE) is feasible.

Acknowledgments

This work has been performed within the SICS Center for Networked Systems funded by VINNOVA, SSF, KKS, ABB, Ericsson, Saab SDS, TeliaSonera, T2Data, Vendolocus and Peerialism. This work has been supported by VINNOVA, SSF and by the European Commission with contract FP7-2007-2-224053 (CONET). This work was also partially funded by ERCIM through the Alain Bensoussan postdoc fellowship program.

Bibliography

[1] A. Liu and P. Ning. TinyECC: A configurable library for elliptic curve cryptography in wireless sensor networks. In *IPSN 2008*, Washington, DC, USA, 2008.

[2] P. Szczechowiak, L. Oliveira, M. Scott, M. Collier, and R. Dahab. Nanoecc: Testing the limits of elliptic curve cryptography in sensor networks. In *EWSN 2008*, February 2008.

[3] D. Liu and P. Ning. Establishing pairwise keys in distributed sensor networks. In *10th ACM conference on Computer and communications security (CCS)*, New York, NY, USA, 2003.

[4] A. Chung and U. Roedig. DHB-KEY: An Efficient Key Distribution Scheme for Wireless Sensor Networks. In *WSNS2008*, Atlanta, USA, September 2008.

[5] ArchRock Corporation. Phynet n4x series, 2008.

[6] IEEE Computer Society. Ieee std. 802.15.4-2006, 2006.

[7] A. Wood and J. Stankovic. Poster abstract: AMSecure - secure link-layer communication in TinyOS for IEEE 802.15.4-based wireless sensor networks. In *ACM SenSys*, Boulder, USA, November 2006.

[8] W. Hu, P. Corke, W. Shih, and L. Overs. secfleck: A public key technology platform for wireless sensor networks. In *EWSN 2009*, Cork, Ireland, February 2009.

[9] J. Granjal, R. Silva, E. Monteiro, J. Sa Silva, and F. Boavida. Why is IPsec a viable option for wireless sensor networks . In *WSNS2008*, Atlanta, USA, September 2008.

[10] J. Vasseur and A. Dunkels. *Interconnecting Smart Objects with IP - The Next Internet*. Morgan Kaufmann, 2010.

[11] N. Kushalnagar, G. Montenegro, and C. Schumacher. IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals. RFC 4919, August 2007.

[12] G. Deloche, N. Kushalnagar, J. Hui, and D. Culler. Transmission of IPv6 Packets over IEEE 802.15.4 Networks. RFC 4944, September 2007.

[13] S. Kent and R. Atkinson. Security architecture for the internet protocol, 1998.

[14] R. Riaz, Ki-Hyung Kim, and H.F. Ahmed. Security analysis survey and framework design for ip connected lowpans. In *ISADS '09*, mar. 2009.

[15] R. Roman and J. Lopez. Integrating wireless sensor networks and the internet: a security analysis. *Internet Research*, 19(2):246–259, 2009.

[16] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. RFC 2460, December 1998.

[17] S. Kent and K. Seo. Security architecture for the internet protocol. RFC 4301, 2005.

[18] Stephen Kent. IP Authentication Header. RFC 4302, 2005.

[19] S. Kent. IP Encapsulating Security Payload. RFC 4303, 2005.

[20] V. Manral. Cryptographic algorithm implementation requirements for encapsulating security payload (esp) and authentication header (ah). RFC 4835, 2007.

[21] J. Hui and P. Thubert. Compression Format for IPv6 Datagrams in 6LoWPAN Networks. draft-ietf-6lowpan-hc-13, September 2010.

[22] Shahid Raza, Tony Chung, Simon Duquennoy, Dogan Yazar, Thiemo Voigt, and Utz Roedig. Securing internet of things with lightweight ipsec. Technical Report T2010:08, SICS, 2010.

[23] A. Dunkels, B. Grönvall, and T. Voigt. Contiki - a lightweight and flexible operating system for tiny networked sensors. In *EMNets'04*, Tampa, USA, November 2004.

[24] Shamus Software. Multiprecision Integer and Rational Arithmetic C/C++ Library. Web page. Visited 2010-04-17.

[25] J. Polastre, R. Szewczyk, and D. Culler. Telos: enabling ultra-low power wireless research. In *IPSN'05*, apr. 2005.

[26] J. Eriksson, A. Dunkels, N. Finne, F. Österlind, and T. Voigt. Mspsim – an extensible simulator for msp430-equipped sensor boards. In *Demo session at EWSN 2007*, Delft, The Netherlands, January 2007.

[27] P. Hoffman. Cryptographic Suites for IPsec. RFC 4308, December 2005.