Design and Implementation of a Security Manager for WirelessHART Networks

Shahid Raza, Thiemo Voigt, Adriaan Slabbert
Swedish Institute of Computer Science (SICS)
SE-16429 Kista Stockholm, Sweden
{*shahid, thiemo, adriaan*}*@sics.se*

Krister Landernäs
ABB Corporate Research
SE-72178 Västerås, Sweden
krister.landernas@se.abb.com

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.

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 [3] 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™. HART 6 and earlier are current looped wired protocols and are insecure. Only a single parity check coding schemes [1] is used to detect communication errors in legacy HART. In contrast, WirelessHART [6] 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 [7]. 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 2 we give

978-1-4244-5113-5/09/$25.00

an overview of the provided security in the WirelessHART standard. In Section 3 we elaborate specifications of the SM. Section 4 and 5 provide the design and the implementation of the SM respectively. In Section 6 we provide an evaluation of the SM. Section 7 shows related work and the paper ends with the conclusion and future work.

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, implement, 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) [8] 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.

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.

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.

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:

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 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 [4]. 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 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 [15] 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.

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.

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) [12] 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 [5] 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.

4. Security Manager Design

We propose the first open design of the SM for WirelessHART networks. Our design is fully compatible with the

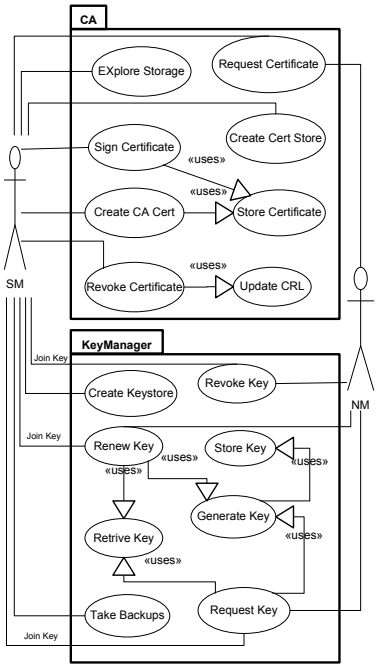

Figure 1. Our proposed use-case diagram of WirelessHART SM

WirelessHART standard and covers all the features specified in Section 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 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.

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.

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 3.1. Figure 2 shows the key request process.

4.2. Key Renewal

All the keys in the WirelessHART network are renewable except the Well-known key. The key renewal request can be

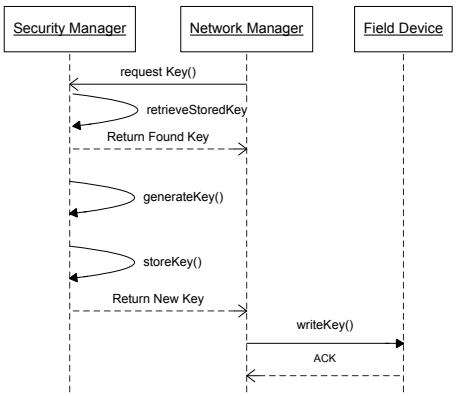

Figure 2. Our proposed Key Request Protocol steps

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 3 shows the key renewal process.

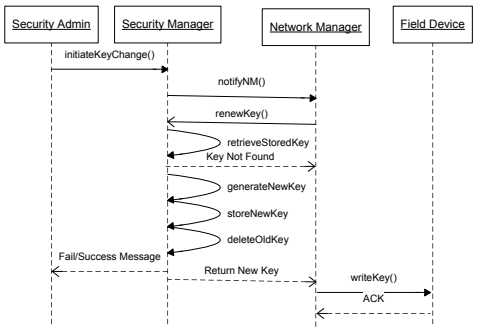

Figure 3. Our proposed Key Renewal Protocol steps

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 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.

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 [21]. But if the generated keys are not random enough the statistical attacks can break the keys in less time and with few resources [10]. 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 [11], 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 4 shows the key generation process.

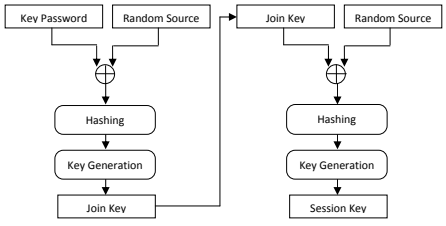

Figure 4. Our proposed Key Generation process

4.5. Key Storage

The SA registers the new device in the WirelessHART Network by generating and adding the Join key and re-

lated 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 5 shows our key storage model.

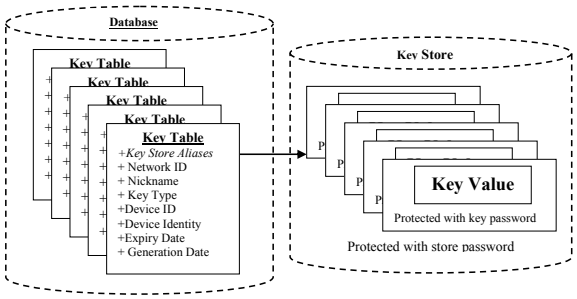

Figure 5. Our proposed Key Storage Model

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 [16]. 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 [18].

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

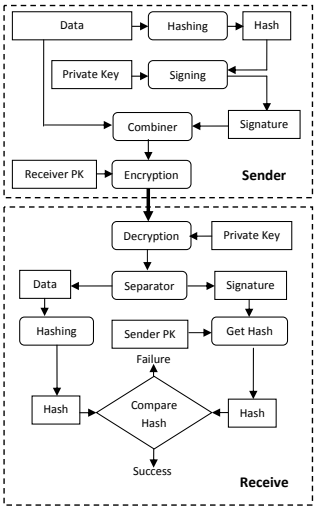

Figure 6. Generic PKI based authentication

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 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) [18], Mutual Authentication [19], etc. In our implementation we use the Metro Web Services architecture [20] 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.

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 [20].

We develop our own CA as a web service using Metro Web services, Java Cryptographic Extension (JCE), and Bouncy Castle API [14]. 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.

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.

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

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

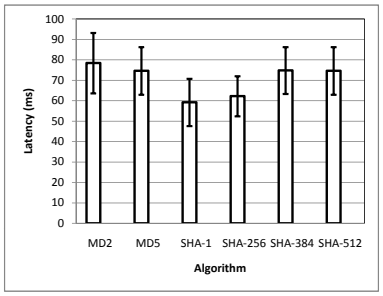

Figure 7. Latency between the NM and the SM with Key generation and Hashing

response time of 71ms which is far less than different reply time requirements in the WirelessHART standard such as maxReplyTime (30s), JoinReplyTimeout (default is KeepAlive), BcastReplyTime (60s), etc. [17]. We also calculate the standard deviation to explain the variation in the latencies. Figure 7 shows the average latencies and standard deviations of different hash algorithm and SHA1PRNS as Pseudo-Random Number Generator (PRNG). Figure 7 depicts that the SHA-1 and SHA-256 algorithms have a lower average latency 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 8 shows

latencies for 10 executions when the NM requests a pre-generated key from the SM.

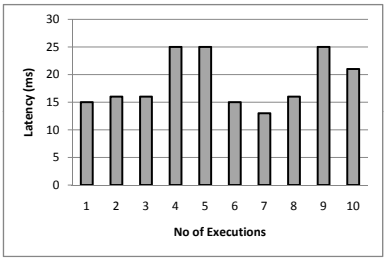

Figure 8. Latency between the NM and the SM when requesting pre-generated keys

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).

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. 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_0 stream cipher algorithm, has improper key management, prone to impersonation attacks, no application level security, etc. [23]. 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. [24] 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 [24], 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 [2].

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.

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 [13] using the reserved security bits in the security sub-layer [17].

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.

References

[1] C. Leung, *Evaluation of the Undetected Error Probability of Single Parity-Check Product Codes*. IEEE Transactions on Communications, vol. 31(1):250-253, 1983.

[2] T. Lennvall, S. Svensson, F. Heklan, *A Comparison of WirelessHART and ZigBee for Industrial Applications.* IEEE International Workshop on Factory Communication Systems, 2008. WFCS 2008 (ISBN: 978-1-4244-2349-1), Pages 85-88

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

[4] J. Lopez and J. Zhou, *Wireless Sensor Network Security*, Cryptology and Information Security Series. ISO Press, 2008.

[5] *VeriSign Sarl*, Route des Arsenaux 41, CH-1705 Fribourg, Switzerland. http://www.verisign.ch/index.html. (May 2009)

[6] J. Song, S. Han, A. K. Mok, D. Chen, M. Lucas, and M. Nixon, *WirelessHART: Applying Wireless Technology in Real-Time Industrial Process Control*. Real-Time and Embedded Technology and Applications Symposium, pp. 377 - 386, April 2008.

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

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

[9] G. H. Larsen, *Software: Man in the Middle*, Datamation, Vol. 19(11):61-66, November 1973.

[10] A. Bogdanoy, *Multiple-Differential Side-Channel Collision Attacks on AES*. Lecture Notes In Computer Science, Vol. 5154, pp:30-44. Proceedings of the 10th international workshop on Cryptographic Hardware and Embedded Systems Washington, DC, USA.

[11] M. Jakobsson , E. Shriver, B. K. Hillyer, and A. Juels, *A Practical Secure Physical Random Bit Generator*. Proceedings of the Fifth ACM Conference on Computer and Communications Security, November 1998.

[12] P. Resnick, Ed., *Internet Message Format*. RFC 2822, RFC Editor, 2001.

[13] W. Hu, P. Corke, W. C. Shih, and L. Overs, *secFleck: A Public Key Technology Platform for Wireless Sensor Networks*. In Proceedings of the 6th European Conference on Wireless Sensor Networks, 2009.

[14] *The Legion Of The Bouncy Castle*. Lock Box Inc, 675 N. 1st Street, Suite 1200 San Jose, CA 95112 USA. http://www.bouncycastle.org/java.html (April 2009)

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

[16] Mike Hendry, *Smart card security and applications*, 2nd Edition, page 62. Artech House, 2001.

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

[18] T. Dierks and E. Rescorla, *The Transport Layer Security (TLS) Protocol Version 1.1*, Internet proposed standard RFC 4346, April 2006.

[19] *Entity Authentication Using Public Key Cryptography*. Federal Information Processing Standards Publication, FIPS PUB 196, February 1997.

[20] *Metro Users Guide*, GlassFish Community, April 2009. https://metro.dev.java.net/guide/

[21] 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 of the National Institute of Standards and Technology, vol. 106(3):511577, May 2001.

[22] International Society of Automation (ISA), *ISA100.11a, Release 1*. www.isa.org/source/ISA100.11a_Release1_Status.ppt.

[23] C. Gehrmann, J. Persson, and B. Smeets, *Bluetooth Security*. Artech House Boston, London, 2004, ISBN: 1-58053-504-6.

[24] N. Baker, *ZigBee and Bluetooth strengths and weaknesses for industrial applications*. Computing & Control Engineering Journal, vol 16(2):20-25, June 2005.