A Security Design for a General Purpose, Self-Organizing, Multihop Ad Hoc Wireless Network

Thomas S. Messerges
Motorola Labs
1301 E. Algonquin Road
Room 2712
Schaumburg, IL 60196
+1 (847) 576-5827
Tom.Messerges@motorola.com

Johnas Cukier
Mitsubishi Electric Research Laboratories
201 Broadway
Cambridge, MA 02139
+1 (617) 621-7506
jic@merl.com

Tom A.M. Kevenaar
Philips Research Laboratories
Prof. Holstlaan 4(WY71)
5656 AA Eindhoven, The Netherlands
+31 40 2742852
tom.kevenaar@philips.com

Larry Puhl
Motorola Labs
1301 E. Algonquin Road
Room 2256
Schaumburg, IL 60196
+1 (847) 576-5463
Larry.Puhl@motorola.com

René Struik
Certicom Research
5520 Explorer Drive, 4th Floor
Mississauga, ON
Canada L4W 5L1
+1 (905) 501-6083
rstruik@certicom.com

Ed Callaway
Motorola Labs
8000 W, Sunrise Blvd.
Room 2141
Plantation, FL 33322-
+1 (954) 723-8341
Ed.Callaway@motorola.com

ABSTRACT

We present a security design for a general purpose, self-organizing, multihop ad hoc wireless network, based on the IEEE 802.15.4 low-rate wireless personal area network standard. The design employs elliptic-curve cryptography and the AES block cipher to supply message integrity and encryption services, key-establishment protocols, and a large set of extended security services, while at the same time meeting the low implementation cost, low power, and high flexibility requirements of ad hoc wireless networks.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General – *Security and protection (e.g., firewalls)*; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design – *Distributed networks, Network communications, Wireless communication.*

General Terms

Design, Security.

Keywords

Security, ad hoc networks, wireless, 802.15.4.

1. INTRODUCTION

Until recently, research on self-organizing wireless ad hoc networks mainly focused on routing approaches and the problems of data transfer over a network with dynamically changing topology. Stajano and Anderson [14] were among the first to realize the importance of security in such networks and described the most important security issues, illustrated by some real-life examples. Later publications arrived at the same conclusion (e.g., [15] and [17]). The very nature of ad-hoc networks and the cost constraints that are often imposed make these networks difficult to secure. Communications cannot rely on the online availability of a fixed infrastructure, thus necessitating decentralized key management, rather than centralized key management, such as implemented, for example, with the well-known 802.11 WLAN standard [4]. Furthermore, ad hoc networks may be highly versatile, involving short-lived communications between devices that may never have met before, thus complicating initial trust establishment and trust maintenance. Well-designed security services can contribute to the reliability and robustness of the network's communication infrastructure and to the protection of application data sent over this network. Recent security research for ad hoc networks seemed to focus on distributing the role of the Certifying Authority over some or all devices in the network (e.g., [1], [7], [8], [13], [16], and [18]), the main approach being based on threshold cryptography [12] and allowing specific coalitions of devices to act together as a source of trust such as a certificate authority (e.g., to generate public-key certificates). Unfortunately, most of these approaches are not very efficient, either in terms of computational or communication overhead. Further, even if efficient methods to share trust were available, this only partially solves security issues that may arise in constrained ad hoc networks, because one still has to address how to use these methods in a real-life communication scenario.

In this paper, we describe our security architectural design for a low data rate, multihop wireless ad-hoc network built on the IEEE 802.15.4 standard [5]. We focus on architectural choices that allow an efficient implementation and a low communication overhead. In Section 2, we describe the properties of the wireless network. Section 3 describes the security requirements for typical application examples. The basic security services provided by our architecture are described in Section 4 and some extended services are given in Section 5. We conclude in Section 6.

2. THE NETWORK AND DEVICES

In this paper we deal with ad hoc wireless networks that are connected via very low-cost, simple devices. Potential applications we are targeting include toys, home automation and security, commercial lighting and climate controls, industrial and military control and sensing.

Ad hoc wireless networks have performance metrics that differ from those of a conventional communication network, such as 802.11b WLAN. For WLANs, data throughput is of major importance: the higher the throughput, the better the WLAN. For the ad hoc wireless networks that we consider, however, major metrics are: low manufacturing cost (a few dollars per device), very low average power consumption (roughly 10 to 100 μW per device), and limited computing requirements, without sacrificing the flexibility to service a very wide range of applications. Devices we consider typically have power and memory comparable to an 8051 or HC08 processor with clock frequency between 4 and 12 MHz, small memory (typically 10 to 32KB of ROM, 1 to 4KB of RAM), and possibly no non-volatile memory, such as FLASH.

2.1 The Device Architecture

The protocol stack of a network device is shown in Figure 1. The network employs the IEEE 802.15.4 Low-Rate Wireless Personal Area Network (LR-WPAN) [5] standard. This standard specifies the Physical (PHY) layer and Medium Access Control (MAC) sublayer of a low cost, low power consumption, ad hoc wireless network. The higher layers we build on top of the IEEE 802.15.4 layers include a network layer, an application support layer, application endpoints, and security services.

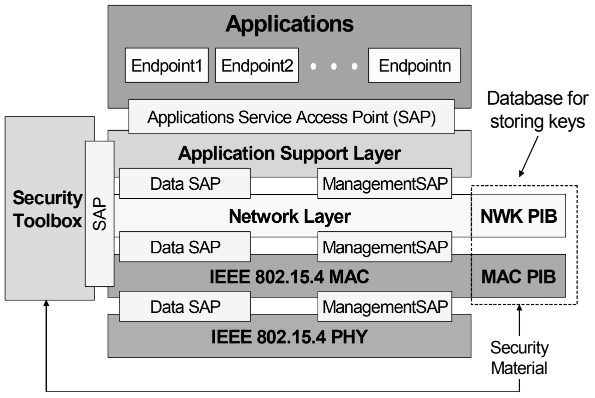

Figure 1 The protocol stack

The PHY layer actually presents a choice between two different layers; the implementer may choose operation in the 2.4 GHz ISM band, or operation in the 868 MHz band (in Europe) and the 915 MHz band (in the Americas). The standard specifies a raw data rate of 250 kb/s in the 2.4 GHz band, 20 kb/s in the 868 MHz band, and 40 kb/s in the 915 MHz band. The range of individual nodes may be limited (e.g., typically around 30-50 meters).

The MAC layer provides services that higher layers can use to access the physical radio. MAC layer protocols provide a means for reliable, single-hop communication links between devices. These protocols include special beaconing and sleep mode functions that enable low duty cycles and ultra-low power consumption – a vital concern for battery-operated network devices. To meet its low implementation cost and low power consumption goals, the IEEE 802.15.4 standard specifies a worst-case maximum MAC payload length of 102 bytes. Since message fragmentation is expensive, both in terms of implementation hardware (buffers, etc.) and power consumption, this limit makes it advantageous to keep exchanged messages below this length. IEEE 802.15.4 supports unicast (i.e., one-to-one) and broadcast (one-to-all) messages, but does not natively support multicast (one-to-many) messages.

The network (NWK) layer, which is not part of the 802.15.4 specification, provides routing and multi-hop functions needed for creating different network topologies, such as peer-to-peer, star, cluster tree, and mesh structures. Multi-hop communication capabilities increase the effective range of a device, allowing it to extend its reach beyond its immediate neighbor devices, while at the same time reducing transmission power and saving battery life. An example network topology (consisting of devices described in Section 2.2) is given in Figure 2.

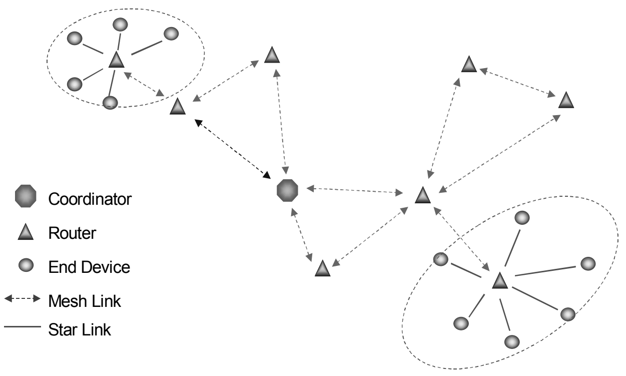

Figure 2 Example network topology

On top of the NWK layer sits the Application Support (APS) layer, which provides a solid foundation for constructing applications. All applications residing in the Application (APL) layer, from lighting to home security, toys and games, will use the same basic structure for over-the-air commands. Multiple applications (the so-called application endpoints) may reside on a single network device and, thus, share its single radio. The application support layer helps ensure that these will interoperate with one another.

Alongside all of the layers we position the so-called security toolbox, which is the main focus of our paper. Figure 1 shows the security toolbox alongside the other layers to stress that each layer needs the services it encapsulates. These services include the mechanisms (e.g., software for security protocols and algorithms) common to each layer, thus enabling an efficient design. The IEEE 802.15.4 MAC layer provides the means for securing single-hop data communications, but does not provide facilities for key establishment and key maintenance. The security toolbox provides these complementary services, as well as the end-to-end protection of data communicated over multiple hops and the enforcement of security policies.

2.2 Device Types

It is useful to classify devices according to device type. Figure 3 shows a three-tier modeling of device types according to physical type, logical type, and application type.

With physical device types, we distinguish between so-called Full-Functional Devices (FFDs) and Reduced-Functional Devices (RFDs), as also defined in the IEEE 802.15.4 standard. An RFD may take on the logical role of an end device, whereas an FFD may additionally take on the role of coordinator or router (see Figure 2). RFDs generally have less computational resources and memory than FFDs, since these are aimed at providing simpler capabilities than FFDs, at reduced cost. Implementing security provisions with RFDs poses considerable challenges, due to severe constraints on implementation cost; FFDs are less constrained.

We distinguish three different logical device types: coordinators, routers, and end devices. Each network has exactly one coordinator, who is responsible for identifying the network and setting most network parameters. There may be more than one router. Coordinators and routers may connect to routers and end devices, but end devices may not connect to other end devices.

Finally, we distinguish device types according to their representation of a particular application (e.g., switches, lights, sensors, thermostats, etc.). These application device types distinguish devices from the end-user perspective.

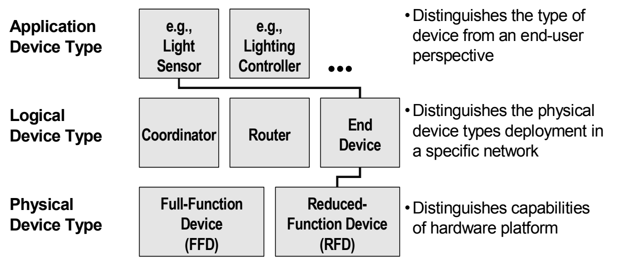
Application Device Type
e.g., Light Sensor
e.g., Lighting Controller
•••
• Distinguishes the type of device from an end-user perspective
Logical Device Type
Coordinator
Router
End Device
• Distinguishes the physical device types deployment in a specific network
Physical Device Type
Full-Function Device (FFD)
Reduced-Function Device (RFD)
• Distinguishes capabilities of hardware platform

Figure 3 Device model

3. SECURITY REQUIREMENTS

There are many applications, such as industrial control and monitoring, asset tracking, intelligent agriculture, home automation, and consumer electronics, for which conventional communication networks are unsuitable, due to excessive power consumption, high implementation cost, or the need to employ some type of trained network administrator for network establishment or maintenance. Self-organizing ad hoc wireless networks can overcome these limitations, but introduce security issues of their own that must be satisfactorily addressed to ensure successful deployment of these networks in the market.

The ad hoc wireless network described in this work is designed to offer standard solutions for a diverse range of applications that may have different security requirements or none at all. This places great demands on the flexibility of the security system, since one must be capable of tailoring the range of supplied security services towards a wide variety of network topologies and application requirements, rather than fixing these. Further, each application imposes cost, performance, complexity, flexibility, and ease-of-use constraints that impact the feasibility of a particular security solution. The "security toolbox" approach satisfies this need for flexibility of deployment towards a wide range of applications. The security toolbox includes all the security services we foresee to be useful for potential products and applications Application designers and implementers may then pick and choose services from this toolbox based on the needs of individual application profiles. At the same time, we took care that these application designers would not have access to the protocols themselves that underlie these services.

To decide which tools and services to include into the security toolbox, we considered a few representative applications, including residential lighting, industrial control and building automation, and autonomous sensor networks. These application areas are now described.

3.1 Residential Lighting

In a typical residential application a consumer might visit a home improvement store to buy a wireless dimmer switch and a wireless light socket (e.g., Figure 4). The dimmer switch would be battery powered and the light socket would be mains powered. The fact that each device has an identifying label or logo, even though different companies may manufacture them, can give the consumer assurance that the two devices will interoperate. The consumer will install the devices by mounting the dimmer switch on the wall and screwing the light socket onto his lamp. Next, the switch and socket would be paired together by manually pressing a button on each device. This would cause the devices to activate, detect each other, and establish a communication link. In the future, the customer might want to add more devices to this network. For example, he might want two lamps controlled by the same switch, he might want an extra switch added to another location, or he might want to add another type of device, such as a ceiling fan, home alarm system, sensors, or a thermostat.

Most residential applications might require minimal security, but one still does not want a neighbor's device (or a hacker) to gain control of one's own system. Hence, one needs logical separation of networks to ensure that the socket only executes commands originating from the dimmer switch and not from, e.g., one of the neighbor's devices or a hacker's device. In more complex scenarios, devices may have to communicate with more than one device (e.g., two switches controlling one light socket). Even when modifying residential networks, minimal human interaction should be required while maintaining security. The security toolbox needs to support all of the underlying security functions (such as key establishment, key update). These types of low-flexibility applications point towards a security architecture based on symmetric-key mechanisms.

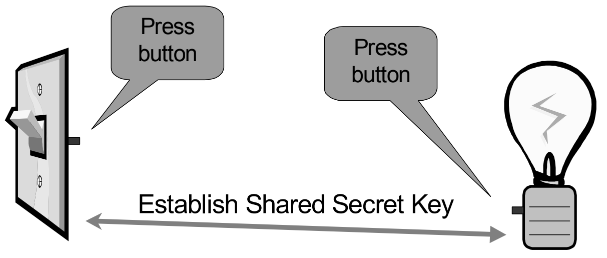

Figure 4 Manual configuration (residential setting)

3.2 Industrial Control and Building Automation

In an industrial setting, a contractor may decide to use wireless devices to save on the cost of wiring a large building. Many hundreds, and even thousands, of lights and controls for offices, conference rooms, hallways, warehouses, etc., need to be configured. Pressing buttons to manually configure such as large system is impractical, so an alternate method is needed. In this case, a configuration device, such as a portable computer, is used to virtually wire the building by linking switches to lights, binding sensor outputs to alarms, or connecting thermostats to the ventilation system.

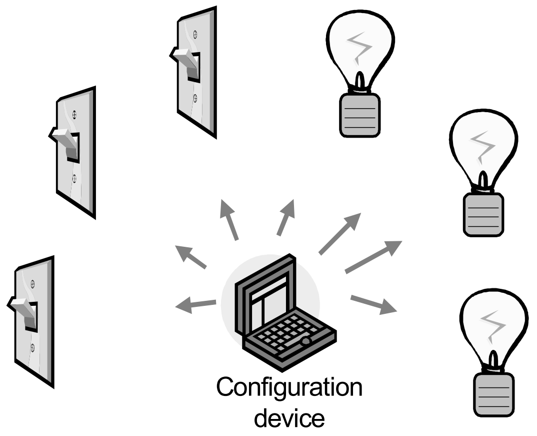

Figure 5 Assisted configuration (industrial setting)

This configuration device should be able to commission the network in a secure manner such that it is difficult for outsiders to reconfigure the system. In this sense it may also play the role of "security manager" (e.g., Figure 5). A security manager is capable of installing initial security material into devices in order to help create trust relationships. Building maintenance personnel might use configuration devices to override, repair, or control already-configured devices. In industrial settings, such as these, the configuration device needs a secure method to identify itself to the other devices and the other devices need to be capable of receiving security information from it. The security toolbox will have services to support this security manager role. Applications will only implement those capabilities that are needed, though many will have to resort to public-key based trust establishment mechanisms to achieve sufficient flexibility.

3.3 Autonomous Sensor Networks

In some scenarios, the setup of wireless network devices needs to be either partially or entirely autonomous. For example, a large warehouse owner may wish to install smoke detectors throughout his building (e.g., Figure 6). An economical way to do this is to buy one central controller device and many individual smoke detectors. Instead of using a dedicated configuration device to configure this system, the warehouse owner simply mounts the detectors throughout his building and allows the devices to self-configure. The devices would automatically discover each other, detect communication routes to the central controller, and create a secure mesh network. However, securing this autonomous network presents unique challenges. For example, the devices in the network must be capable of identifying each other and determining whether other devices should be allowed into the network. These types of applications point towards a security architecture that includes public-key based mechanisms.

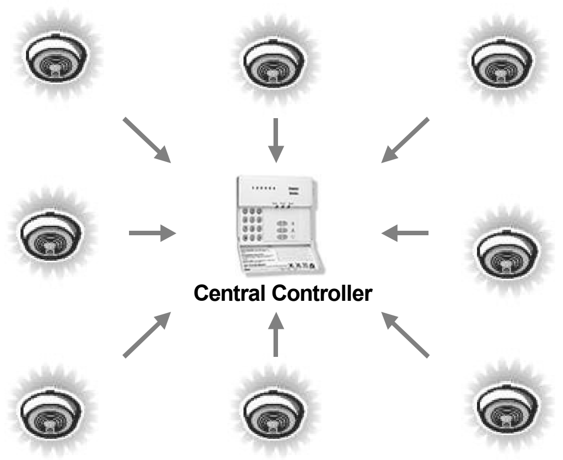

Figure 6 Autonomous configuration (sensor networks)

3.4 Different Applications on One Network

An important requirement is that it should be possible for several applications to run within a single network, even if not all applications trust each other with sensitive data. This means that application data traveling over several hops in a network should be protected from applications running on the intermediate devices. Furthermore, some applications in a network might need to communicate securely while others can do without security.

3.5 Different Applications on One Device

A meter reader device is an interesting example that shows the need for multiple secure applications on a single network device. In this example, a metering device would have concurrent applications running on behalf of water, gas, and electricity providers. The gas, water, and electricity applications should communicate only with equipment operated on behalf of their respective utility providers. To enforce this security policy, each application would set up its own link keys. The metering device (attached to the home) would have keys for each of the applications (e.g., water, gas, electric), but meter reading devices operated on behalf of each utility provider (e.g., operated by a representative hereof, or installed in the utility provider's truck) would only contain the individual keys that are relevant for operating their own business.

3.6 Securing the Network Infrastructure

Applications depend on the reliability and robustness of the underlying network infrastructure. Hence, the need to secure the network against attacks on the infrastructure itself (see, e.g., [6]). Attackers should have no entrance to the network at the MAC and NWK layer to prevent attacks, such as the denial of service attack based on spoofing the 'device disassociation' command (which would remove the device from the network).

4. SECURITY DESIGN

In this section, we describe our security architecture. This architecture complements the security services that are already present in the 802.15.4 security specification.

4.1 Security Assumptions

The security provided by our security architecture depends on the security of the symmetric and public keys the security mechanisms operate upon and on the secure implementation of the cryptographic mechanisms and associated security policies involved. Thus, trust in the security architecture ultimately reduces to trust in the secure initialization and installation of keying material and to trust in the secure processing and storage of keying material.

We assume that secret keys do not become available outside the device in an unsecured way. Thus, we assume that a device will not intentionally or inadvertently transmit its keying material to other devices, unless the keying material is protected, such as during key-transport. We assume that the security software and hardware operates as expected. Thus, implementations of security protocols, such as key-establishment, properly execute the complete protocol and do not leave out any steps hereof. Further, random number generators operate as expected.

We do realize the following caveat in these assumptions: due to the low-cost nature of ad hoc network devices, we cannot generally assume the availability of tamper-resistant hardware. Hence, physical access to a device may yield access to secret keying material and other privileged information and access to the security software and hardware.

Due to the cost constraints mentioned in Section 2, we have to assume that different applications using the same radio are not logically separated using a firewall or sandbox. In addition, from the perspective of a given device it is even not possible (barring certification) to verify whether cryptographic separation between different applications on another device, or even between different layers of the communication stack hereof, is indeed properly implemented. Hence, we have to assume that separate applications using the same radio – the so-called application endpoints – trust each other (i.e., there is no cryptographic task separation). In addition, lower layers (e.g., NWK or MAC) are fully accessible by any of the application endpoints. These assumptions lead to an open trust model for a device: different layers of the communication stack and all applications running on a single device trust each other. By design of the security toolbox Service Access Point (SAP), layers of the protocol stack do not have direct access to the security protocols or security-relevant data. These layers may merely request security services and can expect to receive results hereof.

In summary: the security services offered by the security toolbox cryptographically protect the interfaces between different devices only; separation of the interfaces between different stack layers on the same device is arranged non-cryptographically, via proper design of security service access points.

4.2 Security Architecture

In the previous section, we observed that there is no cryptographic task separation between different stack layers on the same device; the only cryptographic separation being between different devices. This open trust model on a device has far-reaching consequences, since it allows re-use of the same keying material among different layers on the same device and it allows end-to-end security to be realized on a device-to-device basis rather than between pairs of particular layers on two communicating devices.

The observations above allow us to make the following architectural design choices.

First, we establish the principle that "*the layer that initiates a message is responsible for securing it*." For example, NWK-layer commands are secured by calls to the security toolbox from the NWK layer, and do not undergo further security processing at the MAC layer prior to transmission. This contributes to an efficient implementation by eliminating duplicate security operations, such as message integrity code calculations at each layer of the protocol stack, and by limiting the security communication overhead to one integrity code field and associated security status information field per message. Note, that the open trust model allows the above principle to be implemented quite flexibly. In particular, the layer that initiates a message can delegate the actual implementation of security processing on this message to a lower layer.

Second, we base security on the premise that when two devices exchange messages in a secure way, they will use the same link key, irrespective of whether this message is a MAC message, a NWK message, or an application endpoint message. This is possible due to the open trust model for a device. Using the same link key for all layers limits the amount of keying material that has to be stored on a device and hence storage cost. Thus, our view is that the primary security relationship exists between pairs of devices, irrespective of the internal composition of these devices.

Third, we apply end-to-end security such as to ensure that only source and destination devices have access to their shared link key. This limits the trust requirement to those devices whose information is at stake. Additionally, this ensures that routing of messages between devices can be realized independent of trust considerations (thus, facilitating considerable separation of concern).

The architecture includes security needs at three layers of the protocol stack. The MAC, NWK, and APS layers need services for securing MAC, NWK, and APS frames, respectively. The APS layer also needs services for the establishment and maintenance of security relationships, as evidenced by, e.g., link keys and group keys. Here, MAC frames are delivered from one device to another via a single hop and NWK frames and above are delivered from one device to another via one or more hops. The security toolbox provides services to the MAC, NWK, and APS layers to protect these frames (see below). Although applications are responsible for selecting their own security protection level, they will delegate the actual cryptographic processing to the lower layers. This design makes sure that applications do not need direct access to the security toolbox itself, thus realizing considerable separation of concerns.

4.3 Security Services per Stack Layer

In this section, we discuss the security services at each layer, based on the security requirements considerations in Section 3.

4.3.1 MAC Layer Security Services

To protect the reliability and robustness of the network infrastructure, one needs to secure MAC messages (see discussion in Section 3.6). As a minimum, we would like MAC data and MAC commands to be integrity protected and want to guard against replay attacks. The level of integrity protection might vary from one network to the next (as not all networks have the same robustness requirements or tolerance towards data expansion associated with strong integrity protection). Further, it must be possible for a receiving device to verify the identity of the sending device. Secrecy of messages is not always required, although we do offer this functionality. Within the IEEE 802.15.4 standard itself, received MAC messages are accepted after successful cryptographic processing of the received frame (if security is enabled) or passing of a membership test. Here, cryptographic processing is based on a pre-shared key established between the purported sender and the recipient. The membership test entails checking whether the purported sender of a MAC frame is contained in a locally maintained list of devices the recipient expects traffic from, the so-called Access Control List (ACL). Thus, three choices for protecting messages are offered in the standard: no security, cryptographic security, and source address filtering (via the ACL).

When using cryptographic security, the MAC layer uses the Advanced Encryption Standard (AES), a symmetric block-cipher [2]. The MAC layer can be configured to send messages that are encrypted, appended with integrity bits, or both. The number of integrity bits can be set to 32, 64, or 128. When the MAC layer receives a frame that is cryptographically protected, it looks at the source of the frame and retrieves the link key associated with that source from the MAC PIB database (see Figure 1). It then uses this link key to check any integrity bits and decrypt (if encrypted) the payload. The MAC layer then notifies the upper layer that a message was received and it reports the results of these security operations.

Encryption at the MAC layer is provided using AES in Counter (CTR) mode; integrity using AES in Cipher Block Chaining (CBC) mode [10]; combined encryption and integrity using the so-called CCM mode ([3], [11]), a combination of the CTR and CBC modes. Figure 7 shows a MAC frame with security. Security adds a frame count, a key sequence count, and integrity field to the MAC frame. The entire frame is integrity protected, but only the payload can be encrypted. Frame and key sequence counts are included to prevent replay attacks. Sending devices will increase these counts with every message sent and receiving devices will keep track of the last received count from this sending device. If a message with an old count is detected, it is flagged with a security error.

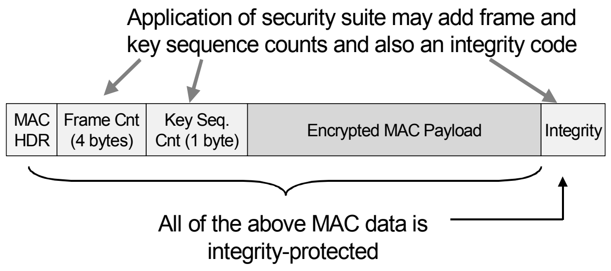
Application of security suite may add frame and key sequence counts and also an integrity code
MAC HDR
Frame Cnt (4 bytes)
Key Seq. Cnt (1 byte)
Encrypted MAC Payload
Integrity
All of the above MAC data is integrity-protected

Figure 7 MAC frame with security at the MAC level

4.3.2 NWK Layer Security Services

The discussion exemplifying the need to secure MAC messages carries over to NWK layer communications as well. Hence, NWK communications requires protection similar to that at the MAC layer. As shown in Figure 8, NWK frames are protected with a scheme similar to that used for protecting MAC frames. In contrast to MAC messages, NWK messages might travel over several hops before reaching the destination. Therefore, the NWK header is not encrypted, such as to enable intermediate nodes along a multihop route to access routing information in the NWK header without the need to perform cryptographic operations or to have access to keying material. This obviates the need to reapply security along a routing path and, thereby, the need to trust every device on every possible multihop route between source and destination device. Note that this use of end-to-end security, where one only protects information that does not change on a hop-by-hop basis, allows the implementation of routing to be independent of security considerations. This independence facilitates system design.

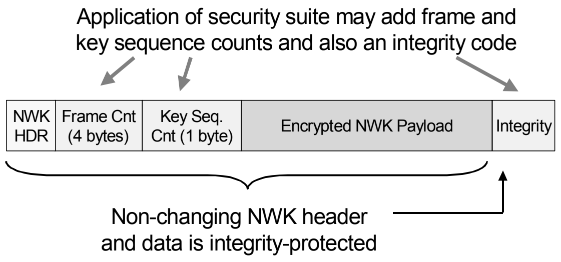
Application of security suite may add frame and key sequence counts and also an integrity code
NWK HDR
Frame Cnt (4 bytes)
Key Seq. Cnt (1 byte)
Encrypted NWK Payload
Integrity
Non-changing NWK header and data is integrity-protected

Figure 8 NWK frame with security at the NWK level

4.3.3 Application Layer Security Services

A single radio may support multiple applications. Some maintain that, in this situation, the application layer would need to provide individual security for each of these applications. However, one of our basic assumptions is that separate applications using the same radio trust each other (see Section 4.1). This assumption implies that the application layer may rely on lower layers for securely sending its messages. So, an application that may require security might delegate the actual implementation of the security protection operation to one of the lower layers (e.g., APS, NWK, or MAC layer). This design decision results in a more efficient implementation. Secured APS frames have a format very similar to the one depicted in Figure 8. It is important to note that also application messages might travel over several hops before reaching the destination. However, due to the end-to-end security policy, the applications running on intermediate nodes need not be trusted.

4.4 Security Roles

In order to be able to manage and control the security of a network in the architecture we propose special security roles that should be implemented in certain devices. These devices will pre-configure other network devices with initial trust data that allows a device to recognize other devices it should trust. A trust relationship can be based on different types of information. For example, a rudimentary level of trust can be achieved using a filtering mechanism based on the unique 64-bit addresses required in every device implementing the IEEE 802.15.4 standard. Two devices might have a non-cryptographic way of establishing the identity of the other device. An example hereof is "pushing buttons", where a human operator controls which devices are executing a protocol. In any case, from a security viewpoint a secure device will have to go through some enrollment phase during which it receives its initial trust data to bootstrap the security process. A brief description follows below. A complete description of the security model underlying the security architecture, including classification of security roles and granularity of cryptographic assertions one can make regarding the provided security services, is beyond the scope of this document.

4.4.1 Security Manager

A device with security manager capabilities is used to configure a network by provisioning initial trust data to network devices. The initial trust data tells a device which devices to trust and can include device addresses, master keys, public keys, or certificates. Devices in the network use initial trust data to establish a link key, which is then used for secure communications of the actual payload. Note that the role of a security manager does not necessarily coincide with that of a configuration manger (see Section 3.2). A decomposition of functionality is possible, separating the tasks of configuration manager and security manager (details are beyond the scope of this paper).

End devices need the capability to recognize (i.e., authenticate) a security manager before accepting initial trust data. Authentication of a security manager can be done using cryptographic means (e.g., public key) or physical means (e.g., a cable). Since a security manager device may not always be kept physically secure, a system needs to be designed so that it can recover when a security manager is stolen or lost. A security manager may also be used as a proxy for relaying initial trust data.

4.4.2 Additional Authorities

A Certificate Authority (CA) or Key Distribution Center (KDC) is also capable of provisioning initial trust data. The CA or KDC is kept physically secure. So, compared with a security manager, it should be less likely that a CA or KDC will be stolen, lost, or under physical control of an attacker. In keeping with traditional definitions, a CA distributes initial trust data for public-key based systems (e.g., private keys, public keys, root keys, and certificates) and a KDC distributes initial trust data for symmetric-key based systems (e.g., a master key).

4.5 Security Toolbox Concept

In our design of the protocol stack (see Figure 1) all the security functionality is shielded from the layers of the stack. The different layers can request certain services from the toolbox or the toolbox might initiate a certain action itself (e.g., it might start up a key establishment process when a required link key is not yet available). The main tasks of the security toolbox are to establish and maintain keying relationships . Additionally, it may handle the security of incoming and outgoing frames for all layers of the stack, if this functionality is not implemented in each individual layer itself (note that the MAC layer handles its own cryptographic processing, as defined in 802.15.4 [5]).

In Section 4.6 below we describe the key-establishment protocols available in the security toolbox in some more detail. Section 4.7 describes some the other basic protocols provided.

4.6 Key-Establishment Protocols

Our design includes both symmetric-key based and public-key based key establishment protocols. This allows implementers to balance the cost and benefits of either class of key establishment mechanisms. Symmetric-key based key establishment allows a smaller footprint implementation and less computational overhead in establishing a shared key than public-key based techniques (see Figure 9), whereas public-key based techniques offer easier configuration and trust management, more flexibility and scalability, and greater resistance to some forms of attack.

	Block-cipher AES-128	**ECC (163-bits)**
Software	ROM: 800 – 2,300 bytes RAM: 60 bytes	ROM: 3-5 KB RAM: 300 bytes
Hardware (Assisted)	≈ 6-7,000 gates	≈3,500 gates ROM: 1-2 KB

Figure 9 Implementation cost of AES-128 and ECC-163 in software and hardware (strongly platform dependent) [figures provided by Certicom Research]

Symmetric-key based key establishment techniques might, therefore, be deployed best in small, less flexible networks (such as those described in Section 3.1 and, possibly, some of those described in Section 3.2), whereas public-key based techniques come at a premium in larger scale networks or networks that require more flexibility (such as those described in Section 3.2 and Section 3.3). Figure 10 shows features shared by all key-establishment protocols. Each protocol involves two entities, an initiator device and a responder device, and each involves four steps, the establishment of a trust relationship, the exchange of ephemeral data, the use of this ephemeral data to derive a link key, and the confirmation of the link key.

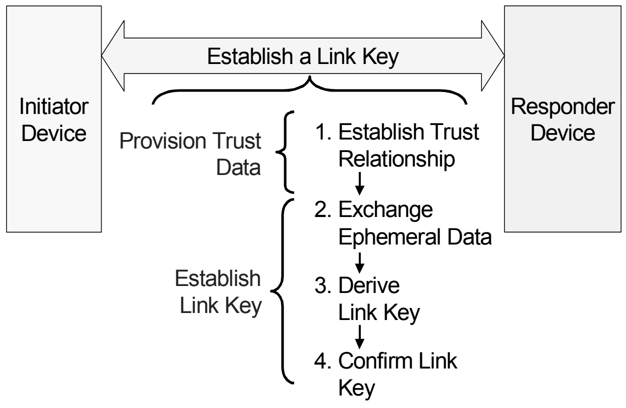

Figure 10 Four steps for key establishment

4.6.1 Symmetric-Key Key Establishment

The two symmetric-key protocols we propose for establishing link keys are the Symmetric-Key Key Establishment (SKKE) protocol and the Unprotected Key Establishment (UKE) protocol. In the SKKE protocol, a trust relationship is established using a shared, secret, and symmetric key, referred to as a master key. This master key, for example, may be pre-installed during manufacturing, may be derived using some cryptographic method, may be installed by a KDC, may be based on user-entered data (e.g., PIN, password, or key), or may be read off the package or chassis of the wireless product (e.g., a bar code). The secrecy and authenticity of the master key needs to be upheld in order to maintain the trust foundation of our SKKE protocol.

In the UKE protocol, the master key is a fixed, default value. In this case, potential adversaries may know the master key. The trust relationship for the UKE protocol is based merely on knowledge of the other device's IEEE 64-bit address, which could be obtained from initially exchanged messages. This approach does not give any cryptographic security on the derived link key because it is susceptible to eavesdropping. Nevertheless, the UKE protocol might be useful in settings where risk during the limited duration of the key-establishment protocol can be tolerated. It is especially suitable for low-cost implementations that cannot change a master key (e.g., systems without flash or other nonvolatile programmable memory).

Successful completion of either the UKE or SKKE protocols results in the following:

– Both devices share a link key.

– Key confirmation: Each device knows that the other device has computed the correct link key.

– No unilateral key control: No device has complete control over the link key that is established.

– No forward secrecy: Eavesdropping during the UKE protocol or eavesdropping and compromise of the master key in the SKKE protocol exposes all the future and past communications.

If the SKKE protocol is successfully used and each device knows beforehand that only the other device possess the master key, then implicit key authentication is achieved (i.e., each device is sure that no other device has access to the established link key). If the SKKE protocol is used without each device knowing who has access to the master key (also true for the UKE protocol), then implicit key authentication can still be achieved provided that no eavesdropping occurs (i.e., no passive attacks), that no messages are modified (i.e., active attacks), and that the protocol is executed in an environment where the two devices have a non-cryptographic way of establishing the identity of the other device.

The building blocks required for the UKE and SKKE protocols include the AES block cipher, an unkeyed hash function (e.g., the Matyas-Meyer-Oseas hash [9] based on AES), and a (pseudo) random number generator. For maximum hardware and software reuse, the random number generator itself may employ the AES algorithm.

4.6.2 Public-Key Key Establishment

The two public-key protocols we propose are the Public-Key Key Establishment (PKKE) protocol and the Certificate-Based Key-Establishment (CBKE). The PKKE protocol (which does not employ certificates) can be used to prevent passive attacks that involve an eavesdropper monitoring the key-establishment procedure. Trust between devices is established by exchanging unsigned public keys together with the corresponding device IDs. With this method, one cannot check cryptographically to whom a public key belongs and hence one should rely on the environment to provide assurance of the trustworthiness of a public key and purported device ID. The PKKE protocol is especially useful for network applications that lack the ability to employ a CA.

The Certificate-Based Key-Establishment (CBKE) protocol uses public-key technology with (digital) certificates and root keys. A digital certificate is simply a public key together with the device's 64-bit IEEE address, signed by the CA. Certificates can provide a mechanism for checking cryptographically to whom the public key belongs and whether a device is a legitimate member of a particular network. Once a certificate and root key are securely provisioned to a device, active and passive attacks in subsequent key-establishment protocols can be thwarted. We propose the use of 163-bit Elliptic-Curve Cryptographic (ECC) techniques for the PKKE and CBKE protocols. ECC techniques offer a reasonable computational load (especially since they will be performed infrequently in most application scenarios), and smaller key lengths for equivalent security than other techniques. Small key lengths are important to minimize the size of message storage buffers on network devices, which in turn reduces their implementation cost.

Both the symmetric-key based and the public-key based protocols were designed such as to maximize commonalities between message flows and to allow re-use of cryptographic building blocks. This way, a single implementation of telecommunication overhead and error handling seems to be possible. Furthermore, each protocol step, including those for elliptic-curve based key establishment with certificates, could be implemented within a single frame, due to their small length of less than 100 bytes.

4.7 Other Basic Security Services

In addition to key establishment, our security design also provides basic services for key transport, entity authentication, and frame protection. Key transport services are needed when keys are delivered via a security manager, sent to or restored from a backup device, distributed to a group of devices, or refreshed. Entity authentication services are needed to detect whether a device is still available (e.g., memory can be freed by deleting keys of devices that are no longer available). Frame protection services are needed to provide the NWK and APS layers with capabilities to protect frames that are delivered over multiple hops.

4.8 Using the Security Toolbox

The interface to our security toolbox is given using the notation described in the IEEE Standard 802.2 (1998 edition) and used in the IEEE 802.15.4 specification. Each service is represented as a primitive.

A primitive can be one of four generic types:

Request: The request primitive is passed to the security toolbox to initiate a security service.

Indication: The indication primitive is passed from the security toolbox to indicate the occurrence of an internal security event that may be logically related to an external request or an internal security event.

Response: The response primitive is passed to the security toolbox to complete a procedure previously invoked by an indication primitive.

Confirm: The confirm primitive is passed from the security toolbox to convey the results of one or more associated previous service requests.

A typical sequence of request, indication, response, and confirm primitives is illustrated in Figure 11.

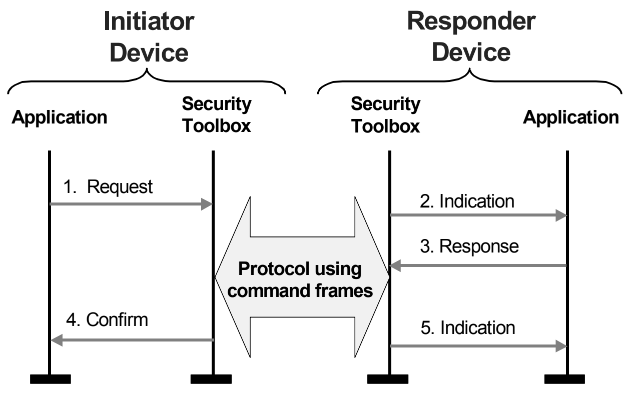

Figure 11 Message sequence chart for security primitives

5. EXTENDED SECURITY SERVICES

In practical networks, the basic security services described above turn out not to be sufficient to handle all security aspects and security policies. Extended security services are required that are described below. However, the basic security services can be used to construct these extended services that involve security policies.

5.1.1 Key Update

Keys may be updated as part of a security policy or to recover from a lost key. In our implementation, a mandated security policy is that keys shall be updated when the key or frame counts (e.g., in the MAC or NWK headers) reach their maximum value (since otherwise freshness or even confidentiality would be at jeopardy) or when a device leaves or enters a group that is sharing a key. Expiration times may also be used to cause a key update.

5.1.2 Orphan Management

End devices (i.e., RFDs) need to be kept extremely simple and cheap. To meet this requirement, end devices may not have expensive features such as nonvolatile memory (e.g., FLASH memory). When power is removed, as may occur during a power outage or due to a discharged battery, link keys, and other configuration information stored in RAM, will be lost. A device that loses its memory, referred to as an "orphaned device", will need to automatically reestablish its association with a router in a secure fashion without human intervention. An example of such a scenario is that with remote controls, where the simple act of replacing a battery should not force users to reconfigure the whole system (i.e., one should have a smooth recovery process without the need for user involvement).

Secure orphan recovery is accomplished using the security toolbox’s key transport primitive. It involves a two-step process: enrollment and recovery. When an end device first connects to a ‘parent’ device (e.g., a router), it establishes a secure connection and then executes the orphan enrollment procedure. This procedure involves the end device encrypting its configuration information (e.g., long-term keys and access control lists) and storing this data on the more powerful ‘parent’ device, which has nonvolatile memory. If at some later time the end device becomes an orphan, it will execute the orphan recovery procedure to recover its configuration information from its ‘parent’ (who acts as a secure remote memory service for the device). Note that this allows storage of configuration data securely anywhere on the network and arbitrary replication of secured data, should this be required.

The orphaning steps rely on a unique 128-bit *Orphan Key* stored on each device. For example, this key can be kept in any type of one-time-programmable memory, such as laser-etched memory cells. An RFD already needs to store its unique 64-bit IEEE address, so storing an extra orphan key is an acceptable cost. An end device uses this key to protect the configuration data it stores on the router.

5.1.3 Assisted Association

An installer accomplishes assisted association (i.e., the linking of two devices at the MAC layer) of an end device and a router by initiating a physical action on each of the devices causing the two devices to associate. For example, the installer can push a button on each of the devices within a limited time. After two devices are associated, cryptographic keys can be established to create a secure and authenticated communication link.

5.1.4 Remote Association

An installer accomplishes remote association of an end device and a router by storing the end device's address into a device table of the router. For example, a bar code reader could read the address of an end device and load it into the router. Upon power-up, the end device is initially an orphan. The router associates only with orphaned devices that are listed in its device table. Remote association is made secure by using the secure orphaning protocol, which means the router is loaded with configuration data protected with the end device's orphan key (i.e., orphan enrollment).

5.1.5 Portability

A network device might be portable, such as a universal remote control. Normally, a remote control might communicate and share keys directly with the television it controls. However, when the remote control moves, or other objects in a room move, this communication path may be disrupted. Communication paths through a network are not always consistent and the security solutions need to account for this. Orphan management can be used to securely implement portability. In this case, a device leaves the domain of its parent and enters the domain of some ‘stepparent’. If the stepparent tracks the end device's orphan configuration data, then the end device can act as an orphan and reacquire a link with the stepparent. Use of a stepparent is also useful for backup purposes. That is, if the real parent becomes disabled, then the orphaned devices belonging to the disabled parent can rejoin the network via the stepparent.

5.1.6 Indirect Addressing

Indirect addressing is an approach for allowing an extremely low-cost end device to send a message through a router to one or more other end devices. The initiating device relies on a router to get the message to the other end devices and is unaware of the address(es) of the recipient device(s). Indirect addressing can be secured in one of two ways – either the end devices share a common key with each other or the end devices share a common key with the router. If end devices do not share a key with the router, then the router is only relied upon to deliver the message. It cannot manipulate the message without the end devices noticing.

5.1.7 Multicast

In some situations, a device needs to send a message to a number of devices. For example, a light switch may need to control ten lights. It would be inefficient for the light switch to send ten separate messages to each of the lights. Multicast will be secured by using a group key that is shared by a group of devices and maintained by the device that formed the group.

5.1.8 Code Download

It is often desirable to have the ability to download new software to devices to fix bugs or add new features. The integrity and authenticity of new code needs to be protected to prevent rogue software or viruses from being introduced into the system. Code download facilities do not necessarily need to be part of our architecture, since manufacturers might opt for using a proprietary mechanism for code downloaded to their devices. For example, in the simple devices used in our proposed network, downloaded code will be native machine code (not code in an interpreted language such as Java) that is specific to a particular manufacturer's device. The manufacturer would be responsible for digitally signing this code and his devices would use a pre-installed manufacturer root key (stored in ROM) to verify a signature before enabling downloaded code. Nevertheless, the protocols for signature verification or data authentication can easily be provided in our security design, since these can be implemented by re-using building blocks that are required for, e.g., key establishment or authentic data transfer.

5.1.9 Service Discovery

In the case of service discovery, network devices will be able to discover and learn about each other's capabilities. This ability enables easier configuration, for example a light switch can automatically discover all nearby lights that it may want to control. On the security side, however, service discovery poses a privacy and security issue. We do not want burglars to be able to drive through a neighborhood and learn which houses have wireless devices, or how many wireless devices of which type are in a particular home. The solution for secure service discovery is to provide a lock out capability for the service discovery feature. One approach would be to force users to manually override this lockout when they want to add a new device to the network. As an example, when a user presses a button to add a device to the network, service discovery may be temporarily enabled for only a brief duration.

6. SUMMARY AND CONCLUSION

This paper presents a security design for a general-purpose, self-organizing, ad hoc wireless network based on the IEEE 802.15.4 standard. Our design provides message integrity and encryption services, key-establishment protocols (two based on symmetric-key techniques and two based on public-key techniques), and a large set of extended security services, while at the same time meeting the low implementation cost, low power consumption, and high flexibility requirements of ad hoc wireless networks. To maintain implementation simplicity of our design, security operations are performed at one layer of the protocol stack only, rather than on multiple layers, and keys are shared across applications on a single device (while still allowing applications to run at different protection levels). The security protocols were designed such as to maximize re-uses of the symmetric-key engine AES for different security services and to exploit commonalities in protocol message flows. We based the public-key mechanisms on ECC-based techniques, due to its small key length and acceptable performance and implementation cost, relative to other public-key techniques. The resulting security architecture allows ad hoc networks to operate securely, while offering a wide range of additional security services that support potential application-dependent requirements for flexibility, robustness, and disaster recovery. The security architectural design, although described in the context of the IEEE 802.15.4 standard and severe implementation constraints, does not depend hereon and could conceivably be used in any network topology, irrespective of whether these networks are fixed, wireless, or ad hoc wireless.

7. REFERENCES

[1] S. Capkun, L. Buttyan, and J.-P. Hubaux, “Self-Organized Public-Key Management for Mobile Ad Hoc Networks,” IEEE Transactions on Mobile Computing, Vol. 2, No.1, January-March 2003, pp. 52-64.

[2] FIPS Pub 197, Advanced Encryption Standard (AES), Federal Information Processing Standards Publication 197, US Department of Commerce/N.I.S.T., Springfield, Virginia, November 26, 2001. Available from: http://csrc.nist.gov/.

[3] R. Housley, D. Whiting, and N. Ferguson, “Counter with CBC-MAC (CCM),” submitted to NIST, June 3, 2002. Available from: http://csrc.nist.gov/encryption/modes/proposedmodes/.

[4] Institute of Electrical and Electronics Engineers, Inc., IEEE Std. 802.11-1999, IEEE Standard for Telecommunications and Information Exchange Between Systems – LAN/MAN Specific Requirements – Part 11: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications, New York: IEEE Press, 1999.

[5] Institute of Electrical and Electronics Engineers, Inc., IEEE Std. 802.15.4-2003, IEEE Standard for Information Technology — Telecommunications and Information Exchange between Systems — Local and Metropolitan Area Networks — Specific Requirements — Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (WPANs). New York: IEEE Press. 2003.

[6] C. Karlof, D. Wagner, Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures, in Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications (SNPA), May 11, 2003.

[7] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang, "Providing Robust and Ubiquitous Security Support for Mobile Ad-Hoc Networks," Proceedings of International Conference on Network Protocols (ICNP), 2001.

[8] H. Luo, P. Zefros, J. Kong, S. Lu, and L. Zhang, "Self-Securing Ad Hoc Wireless Networks," 7th IEEE Symposium on Computers and Communications (ISCC '02), 2002.

[9] S. M. Matyas, C. H. Meyer, and J. Oseas, "Generating Strong One-Way Functions with Cryptographic Algorithm," IBM Tech. Disclosure Bull., Vol. 27, No. 10A,, pp. 5658-5659, 1985.

[10] NIST Pub 800-38A 2001 ED, Recommendation for Block Cipher Modes of Operation – Methods and Techniques, NIST Special Publication 800-38A, 2001 Edition, US Department of Commerce/N.I.S.T., December 2001. Available from: http://csrc.nist.gov/.

[11] NIST Pub 800-38C, DRAFT Recommendation for Block Cipher Modes of Operation – The CCM Mode for Authentication and Confidentiality, NIST Special Publication 800-38C, Draft, US Department of Commerce/N.I.S.T., Springfield, Virginia, September 4, 2003. Available from http://csrc.nist.gov/.

[12] V. Shoup, "Practical Threshold Signatures," in Advances in Cryptology – EUROCRYPT 2000, B. Preneel, Ed., Lecture Notes in Computer Science, Vol. 1807, pp. 207-220, Berlin: Springer-Verlag, 2000.

[13] J. Staddon, S. Miner, and M. Franklin, "Self-Healing Key Distribution with Revocation," Proceedings of the IEEE Symposium on Security and Privacy (S&P2002), 2000.

[14] F. Stajano, R. Anderson, "The Resurrecting Duckling: Security Issues in Ad-Hoc Wireless networks," in Proceedings of the 7th International Workshop on Security Protocols, B. Christianson, B. Crispo, J.A. Malcolm, and M. Roe, Eds., Lecture Notes in Computer Science, Vol. 1796, Berlin: Springer-Verlag, 1999.

[15] F. Stajano, "The Resurrecting Duckling: What Next?," in Proceedings of the 8th International Workshop on Security Protocols, B. Crispo, M. Roe, and B. Criso, Eds., , Lecture Notes in Computer Science, Vol. 2133, Berlin: Springer-Verlag, April 2000.

[16] M. Steiner, G. Tsudik, and M. Waidner, "Key Agreement in Dynamic Peer Groups," IEEE Trans. on Parallel and Distributed Systems, Vol. 11, No.8, pp. 769-780, August 2000.

[17] K. Wrona, "Distributed Security: Ad Hoc Networks & Beyond," presented at the Ad Hoc Network Security Pampas Workshop, RHUL, London, September 16-17, 2002. Available from: http://www.pampas.eu.org/Position_Papers/papers.html.

[18] L. Zhou and Z. J. Haas, "Securing Ad Hoc Networks," IEEE Network Magazine, Vol. 13, No.6, pp. 24-30, November/December 1999.