Security and Authentication Primer

Manfred Jantscher and Peter H. Cole

Auto-ID Labs White Paper WP-HARDWARE-025

Mr. Manfred Jantscher
Visiting Master Student,
School of Electrical and Electronics
Engineering,
The University of Adelaide.

Prof. Peter H. Cole
Research Director, Auto-ID Lab, ADELAIDE
School of Electrical and Electronics
Engineering,
The University of Adelaide.

Contact:

E-Mail: m.jantscher@cisc.at or cole@eleceng.adelaide.edu.au

Internet: www.autoidlabs.org

Hardware

Abstract

Although information technology security or cryptography generally is a well established area in computer science this is not true for RFID. With the upcoming large-scale introduction of RFID in the supply chain a number of security and privacy concerns arise. Among them are the traceability of tags without knowledge of their owner, and the ability to clone tags allowing persons to create tags for counterfeits. The possibility to securely authenticate readers and tags would solve a majority of the threats. However, obstacles like very restricted operating power and the requirement to keep tags as cheap as possible and therefore save silicon area impede a straightforward implementation of most of the known cryptographic primitives. There arises a demand for new, lightweight cryptographic primitives that offer security services tailored to the requirements of RFID systems while considering their resource constraints. This paper first presents a comprehensive overview of state-of-the-art cryptography, followed by a discussion of privacy and security threats in RFID, and includes a research survey of works conducted in this area. As the title of this paper suggests this paper is a primer, and provides such information that can assist other researchers in developing security mechanisms, suitable for inclusion on low cost RFID tags, such as lightweight cryptographic techniques, or one-time codes, to protect the supply chain against illicit trade. If such RFID devices become available and are deployed as anti-counterfeiting solutions, hopefully they will economically secure future supply chains.

1. Introduction

Information technology security or cryptography, nowadays, is a well established area of computer science. Modern applications of cryptography allow us to transfer confidential data over insecure channels, take care of our bank transactions online using of-the-shelf computers, and sign obligatory contracts over the Internet. These applications are based on cryptographic building blocks, so-called primitives that provide certain desirable services like encrypting and decrypting messages.

One of the most important of these services is authentication. It allows an entity to prove its identity to a verifying party. Often, we can find this service as part of the process of establishing secure channels. In this case the concerned parties mutually verify their identities in a process which is referred to as mutual authentication. It is desirable to have mutual authentication for RFID systems too. This would, on the one hand, prevent persons from being able to clone tags and therefore provide aid against counterfeiting, and in addition, prevent intruders from being able to identify arbitrary tags. RFID tags that provide an authentication capability may be used to prevent (or at least detect) tag cloning. The assumption is that tag with authentication compatibilities cannot be cloned, without being detectable. If a security mechanism exists and even if an authenticated tag successfully cloned , the RFID system, consisting of middleware and back-end systems should be able to detect the presence of two or more apparently identical authenticated tags, and hence flag security issues. While the real tag (uncloned) may not be identifiable, in time increased security measures deployed will detract counterfeiters from cloning tags, and hence improve the security of the supply chain.

Unfortunately, often it is not possible to straightforwardly implement cryptographic primitives in RFID systems because tags normally are very restricted in available power and, because they have to be cheap, chip area plays a crucial role. A demand for new, lightweight cryptographic primitives arises offering security services tailored to the requirements of RFID systems while considering their resource constraints.

This paper is structured as follows: Section 2 discusses information technology security in general presenting a state-of-the-art overview on cryptography. After dealing with desirable security services, common attacks, and security models this section covers cryptographic primitives divided into the three main groups unkeyed, secret-key and public-key primitives. Section 3 focuses on security and privacy issues in RFID systems. A discussion of security and privacy threats arising with RFID and associated obstacles on introducing secure RFID is followed by a research survey of existing work in this area. Finally, a conclusion summarizes the main features of the work.

2. Information Technology Security

In the age of information processing, Internet and digital communication obviously there is a strong need for information technology security. Exchange of confidential messages, online-transfer of money and access to information services are just a few examples of procedures that rely on the security of computer systems and networks. Therefore, information systems and information processed by information systems have to be protected. Cryptography is the science that deals with protection of information [1]. In this section we cover the basics about information technology security and cryptography, and its appropriateness to RFID systems and in particular implementation of cryptographic primitives on an RFID Tag. Possible cryptographic solutions may be applied to the anti-counterfeiting applications for a secure supply chain. The reader may be adverted that Appendix A of this work describes important terms used in this section. This may be especially helpful if terms are used prior to their actual definition which is sometimes unavoidable.

2.1. Model

Although cryptography is much more than just encryption, encryption could be seen as the primary goal. Thus, to start to describe information technology security, often the following simple model is provided.

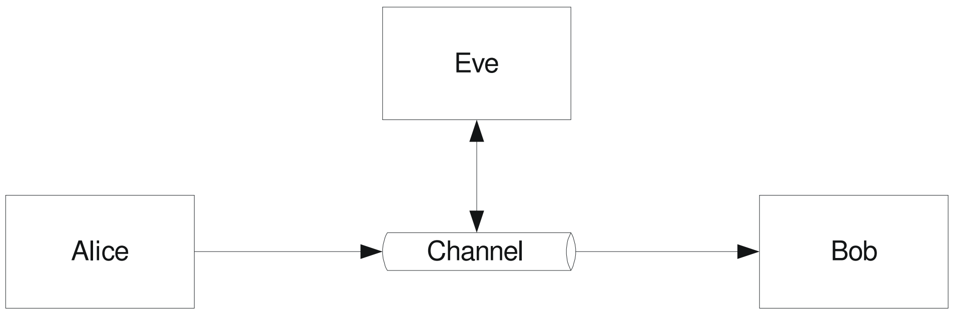

Fig. 1: Encryption Model

Alice and Bob are two parties who trust each other. They want to communicate using the channel which is assumed to be insecure by default. More precisely, in the above example, Alice sends a message which holds information designated for Bob. Eve is the unknown third party who also has access to the channel. The model could be interpreted in two different ways, either as transmission in space, where Alice and Bob sit at different places, or as transmission in time, where e.g. Alice stores a message onto the hard disk of a computer and Bob recalls it at a later time. Given this model, there are a number of concerns. Can Alice be sure that Bob is the only one who can read her message? Can Bob be sure that the message was sent by Alice and if it was, can he be sure that it wasn't altered? Isn't Bob able to deny that he has received the message and isn't Alice able to deny that she has sent the message? These questions lead us to the services that are required for information technology security.

2.2. Security Services

As described in the preceding section, there are a number of security concerns when thinking about computer systems and digital communication. Therefore, a number of services have to be provided in order to enhance information technology security. The following paragraphs are dedicated to the most important of them [2], [3].

Confidentiality ensures that only authorized parties are able to understand information.

Authentication refers to the ability for a party to be sure the received information is from the source it claims to be from.

Integrity assures that a message was not altered on the way to its recipient. Hence, it provides the recipient with the certainty to know what it has received is what was sent by the trusted origin of the message.

Non-repudiation ensures that neither the sender is able to deny that it has sent the message nor the receiver is able to deny that it has received it. Therefore, it provides a proof of transmission and reception of a message.

Access Control refers to the ability to restrict and control access to a system.

Availability provides means to ensure that a system is available whenever needed. Thus, availability services guard systems from attacks against loss or reduction in availability.

Most of these services might be achieved by applying an appropriate cryptographic tool, like an encryption algorithm or a cryptographic hash function, or a series of those. If a series of tools has to be applied in a well-defined way, this way is referred to as a cryptographic protocol. Before we will discuss the main types of cryptographic tools we would like to spend a few words about attacks and security models.

2.3. Attacks

Modern cryptography tools (primitives) face a variety of attacks they have to withstand. Before classifying these attacks a basic principle of state-of-the-art cryptography has to be explained. According to Kerkhoff's principle, stated in the nineteenth century by Auguste Kerkhoff, the security of a cryptosystem must not depend on the secrecy of data independent details about the system [3]. Data independent details of a cryptographic system are the algorithm and its implementation. Therefore, attacks on modern primitives often aim at the recovery of plaintexts from ciphertexts or even worse on the recovery of secret keys [4].

Attacks may be classified into passive and active attacks [3]. Passive attacks denote monitoring of channels or side channels, but not alteration of messages. Obviously monitoring of a channel includes directly listening to data being transferred. Monitoring of side channels, is listening to effects that come along with the activities on the channel like electromagnetic emanation or current consumption [5]. Passive attacks on encryption schemes may be further subdivided into the following kinds of attacks [3]:

Ciphertext-only attack: The attacker tries to recover plaintext or the secret key just by analysing the corresponding ciphertext.

Known-plaintext attack: By analysing a given block of plaintext and the corresponding ciphertext the attacker tries to extract useful information for the recovery of plaintext encrypted in different ciphertexts or the secret key.

Chosen-plaintext attack: Given the attacker is able to choose plaintexts and generate corresponding ciphertexts it tries to extract useful information in order to recover new plaintexts from new ciphertexts or even may try to extract the secret key.

Adaptive chosen-plaintext attack: The attacker tries to recover plaintext or the secret key by subsequently applying chosen-plaintext attacks where the choice of the plaintext for later attacks depends on the outcome of prior attacks.

Chosen-ciphertext attack: This attack is based on the assumption that the attacker, for a limited amount of time, is able to access means to decrypt ciphertexts. Therefore, we assume the means to be a black box which is able to decrypt a limited number of ciphertexts. The attacker chooses ciphertexts and analyses the corresponding plaintexts in order to gain useful information for the later purpose of recovering plaintexts from ciphertexts or of recovering the secret key without the availability of the black box.

Adaptive chosen-ciphertext attack: As with the adaptive chosen-plaintext attack the attacker subsequently applies chosen-ciphertext attacks, with ciphertexts for later attacks depending on the outcome of prior attacks.

Although we mentioned the attacks above are aimed at encryption schemes, most of them are also applicable for other primitives like cryptographic hash functions or digital signature schemes.

Active attacks as opposed to passive attacks are based on alteration of data transmitted over a channel or alteration of computation in a device. The later is also referred to as fault attack [5]. Note that fault attacks and side-channel attacks are sometimes denoted as implementation attacks since they do not aim at the cryptographic algorithm but its specific implementation.

As mentioned above, security services might be based on primitives or on cryptographic protocols. Cryptographic protocols face yet another type of attack – so-called protocol attacks. The following list names the most important of them [3]:

Known-key attack: Based on the knowledge of prior keys the attacker might try to obtain new keys.

Replay attack: The attacker who is able to record a series of messages exchanged by the trusted parties replays part of it or the complete series at a later time.

Impersonation attack: In this case, the attacker somehow assumes the identity of one of the trusted parties.

Dictionary attack: This is a well known attack usually applied on password schemes. The adversary somehow manages to test a very large number of probable passwords with the intention guessing the right one.

2.4. Security Models

In order to describe the security of cryptographic primitives there are so-called security models. The following paragraphs describe the most important of them [1].

Unconditional security also known as perfect secrecy is the non-plus-ultra security model. It assumes unrestricted computational power of the adversary. Therefore, for a cryptographic primitive to fall into this category there must not be an algorithm for breaking it, irrespective of the computational power available. An example of a simple primitive offering unconditional security is the one-time pad. In order to generate the ciphertext a plaintext is XOR-ed with a unique secret key of the same length as the plaintext. Because of the possible large key sizes such systems are impractical for conventional message encryption. However, there may be applications in systems with small information sizes like RFID.

Computational security assumes polynomial computational power of the adversary. Therefore, a cryptographic primitive is assumed to be computationally secure if there is no algorithm known to break it within polynomial time. Modern primitives are supposed to fall into this category.

Practical security also refers to the computational power of the adversary. However, as opposed to computational security there are no relative bounds. Instead, for a primitive falling into this category there must not be a breaking algorithm which requires less than N operations. The number of operations N is chosen sufficiently high. Modern cryptographic primitives typically offer practical security.

Provable security means that it is possible to show that the complexity of breaking a primitive is equivalent to solving a well known supposedly hard mathematical problem like the integer factorization problem. Typical cryptographic primitives based on public keys fall into this category.

2.5. Cryptographic Primitives

So far, we discussed why information technology security is important, what potential threats are and how the security of cryptographic tools can be classified. What was not covered yet are cryptographic tools themselves, the primitives that provide us with the required services discussed above.

Cryptographic primitives may be separated into three large groups: unkeyed primitives, secret-key primitives and public-key primitives. Each of these groups may further be subdivided into primitives that serve different purposes. Fig. 2 shows a possible taxonomy of cryptographic primitives [3]. Each of these groups is further discussed in one of the following sections.

Important differences between cryptographic primitives include the level of security, basic functionality, methods of operation, performance and the ease of implementation. Basic functionality deals with the objectives discussed in section 2.2 whereas the methods of operation regard to specific ways primitives can be used, e.g. for encryption or decryption. Ease of implementation refers to both software and hardware environments [3].

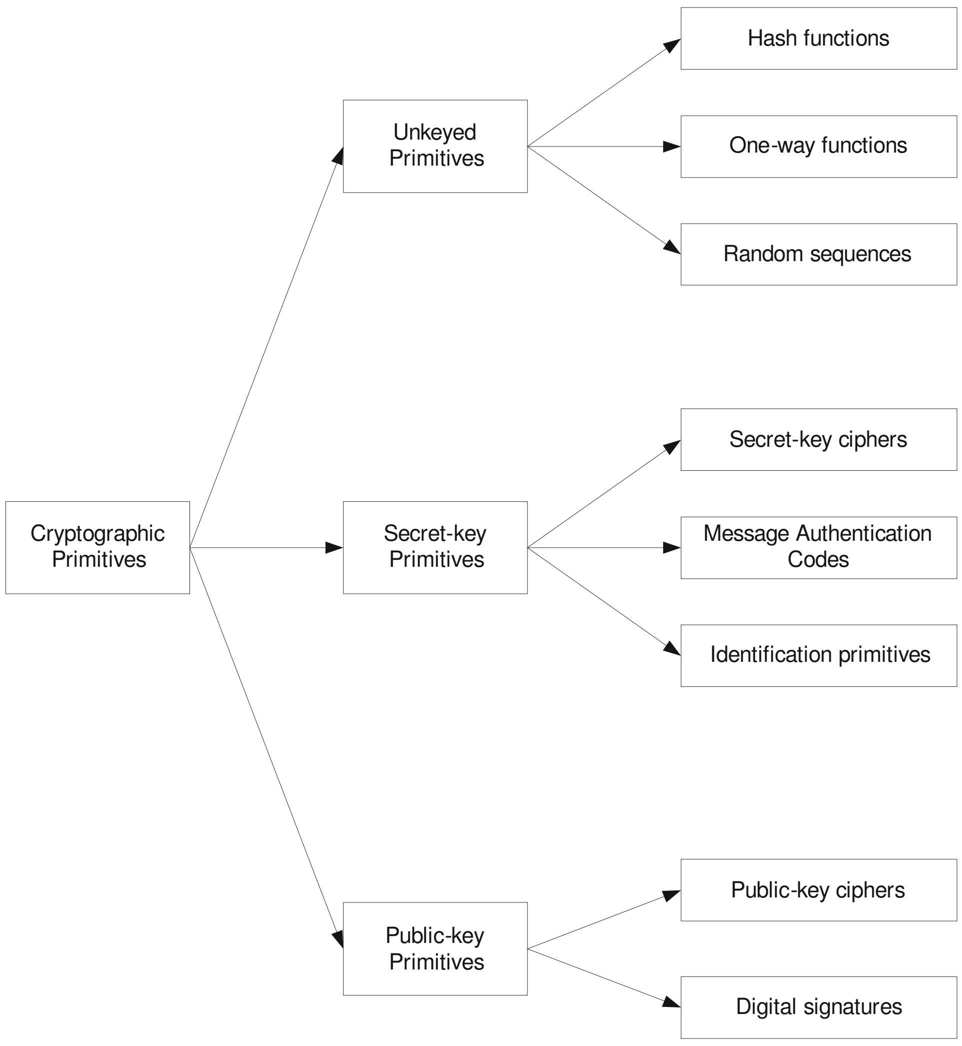

Fig. 2: Taxonomy of Cryptographic Primitives

2.5.1. Unkeyed Primitives

Unkeyed primitives, as the name betrays, are cryptographic tools that are not based on any keys at all. Examples for unkeyed primitives are cryptographic hash functions, one-way functions and random number generators. Taking into account Kerkhoff's principle, because they are not based on keys, they do not fulfil security objectives on their own but often are part of a security system or a cryptographic protocol.

2.5.1.1. Hash Functions

"A hash function is a computationally efficient function mapping binary strings of arbitrary length to binary strings of some fixed length, called hash-values."

[3]

The above definition is very basic. For hash functions used as cryptographic tools a set of requirements has to be fulfilled. First of all, it has to be a one-way function, which means, given a message m it should be easy to calculate the hash-value h(m) but it should be computationally infeasible to do the inverse operation namely to find a message m given the hash value h(m) so that m = h(m). Second, good hash functions should be collision resistant. Theoretically this would mean that there are no two messages m1 and m2 so that h(m1) = h(m2). However, since there are an infinite number of inputs but a finite number of outputs collisions are unavoidable. Therefore, in practice collision resistance means it should be computationally infeasible to find collisions. The third important requirement says that the hash-function should be a random mapping which in practice means it should be computationally infeasible to distinguish a hash-function from a random mapping [6].

Basically there are two types of attacks on hash functions, collision and pre-image attacks. Collision attacks in parallel try to find two different messages m1 and m2 that lead to the same hash value h(m1) = h(m2). If we consider digital signatures (cf. section 2.5.3) where normally hash values of messages are signed for efficiency reasons this leads to a serious security threat. By deliberately designing two messages with the same hash value a digital signature for one message automatically is valid for the second message. Pre-image attacks try to construct a message m that leads to a given hash value h(m).

Hash functions are very important primitives in practice. They are used whenever fixed length values are required instead of arbitrary length messages. They may also be used to generate various pseudorandom keys from a single input.

Unkeyed hash functions are often called modification detection codes (MDC) because of their ability to detect whether a message has been altered. Well known MDCs used in practice are MD5 and SHA-1. MD5 has an output length of 128 bits. The best known attack recently was described by Vlastimil Klima and is able to find MD5 collisions in about one minute using a state-of-the-art notebook computer (Intel Pentium 1.6 GHz) [7]. SHA-1 has an output length of 160 bits. The best known attack, illustrated on Bruce Schneier's Weblog on behalf of the author Xiaoyun Wang, shows a time complexity of 2^{63} which is even better than a brute force attack which would lead to a time complexity of 2^{80} for an output length of 160 bits [8]. Although, so far only collision attacks but no pre-image attacks are known, MD5 and SHA-1 cannot be seen as practically secure any longer. Hence there is a need for new hash functions. The US government standards agency NIST (National Institute of Standards and Technology) in 2001 published a new group of SHA algorithms collectively known as SHA-2. This group includes algorithms with output lengths of 256, 384 and 512 bits. [6]. So far no practical attacks are known for SHA-2.

2.5.1.2. One-Way Functions

As already mentioned with hash functions, one-way functions are mappings f: X → Y which are easy to compute but hard to invert. Expressed in a more formal way this means a polynomial time algorithm exists for computation but no probabilistic polynomial time algorithm for inversion succeeds with better than negligible probability [9].

There are special one-way functions called trapdoor functions with the additional property that given special information, called the trapdoor information, it is possible to calculate the inversion.

One-way functions and trapdoor functions are very important primitives in cryptography and a lot of other primitives are based on them. Examples would be hash functions which are based on one-way functions or public-key cryptography which is based on trapdoor functions.

Just to name examples, the discrete exponential function is a one-way function whereas modular exponentiation is a trapdoor function. At this place we provide simple examples of these functions rather than to describe the details of them since this will go beyond the scope of this work. However, readers are adverted to have a look into the books [3] or [4] which provide very profound knowledge about this topic.

The one-way property of the discrete exponential function is based on the assumption that there is no efficient algorithm for solving the inverse function, the discrete logarithm function. As an example it is computationally easy to solve $x \equiv 3^4 \pmod{17}$ which is 13 but determining the least nonnegative x in $3^x \equiv 13 \pmod{17}$ is a hard mathematical problem.

Modular exponentiation is a very similar problem to the discrete exponential function. Again it is easy to solve $x \equiv 5^4 \pmod{21}$ which is 16 but it is hard to determine x in $x^4 = 16 \pmod{21}$ if the modulus is not a prime (21 in this case). The trapdoor to this problem is that there exists an algorithm which allows solving the problem if the prime factors of the modulus (in this case 3 and 7, since 3*7 = 21) are known.

It should be mentioned that the existence of one-way functions is an open problem in theoretical computer science.

2.5.1.3. Random Number Generators

The third important unkeyed primitive in cryptography are random numbers or random number generators respectively. Many keyed primitives or even cryptographic protocols are based on random number sequences. Examples of use are keys used in public-key cryptography, session keys often used in secret-key cryptography or random sequences (called nonces) in cryptographic protocols.

Before looking at sources of random numbers we should discuss what is randomness? In general, in order to be able to talk about randomness a context has to be defined [3]. For example, we cannot talk about a random number 7 in general but 7 could be a number randomly selected or generated out of a container holding the numbers 1 to 10 which is the context in this case. Furthermore randomness is based on uniform distribution and independence [2]. Uniform distribution means that there is equal probability for each of the numbers in our container to be selected, i.e. the probability to randomly select or generate 7 out of the container 1 to 10 is assumed to be 1/10. Independence refers to sequences of random numbers. In order to talk about a sequence of random numbers there must not be any coherence in the sequence of these numbers, i.e. in the example with a container holding the numbers from 1 to 10, after randomly selecting 7, probabilities of selecting any of the numbers 1 to 10 should be the same.

The sources of ideal random numbers as discussed above, if there are ideal random numbers at all, are based on physical means. Examples could be gas discharge tubes or leaky capacitors. However, often they tend to be costly or slow in the generation of random numbers. Another interesting source are so-called physical unclonable functions (PUF). Based on manufacturing variations of ICs they might be used for secret key generation in electronic devices [10]. However, nowadays most of the random number generators used for cryptography are based on software algorithms. Since pure software algorithms are deterministic the sequences generated are not really statistically random. Therefore, algorithms based random number generators are called pseudo-random number generators and the sequences generated are called pseudo-random number sequences. The sequences generated are based on short random sequences called seeds. Most of the time, the generation algorithms are known but the seed is unknown. Sequences generated by good pseudo-random number generators will pass many tests of randomness and therefore they are applicable for cryptographic purposes.

2.5.2. Secret-Key Primitives

Secret-key cryptography denotes information technology security systems that are based on keys secretly shared between trusted parties. In the literature also the word symmetric-key cryptography can be found meaning the same systems. As Fig. 2 shows, there are various applications of secret-key cryptography. The following sections will highlight the most important of them.

2.5.2.1. Secret-Key Ciphers

Ciphers deal with encryption and decryption of information. In the world of secret-key cryptography basically there are two types of ciphers in use today, block ciphers and stream ciphers. Both are based on the same model which we will explain first.

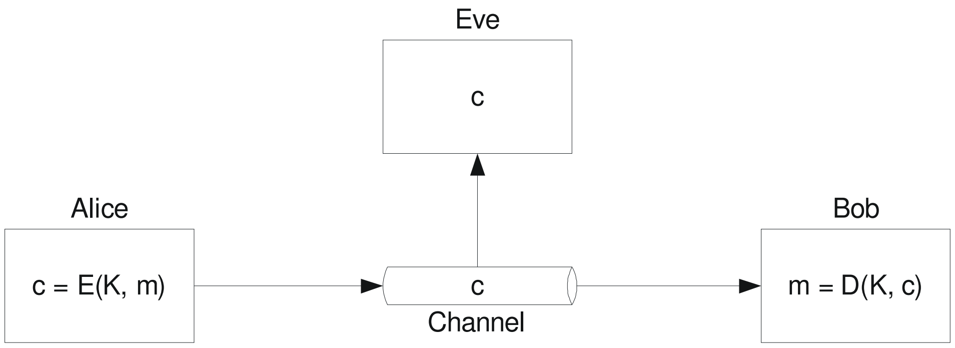

Fig. 3: Secret-Key Encryption

Assuming the basic encryption model already described in section 2.1 (Fig. 1) of this work, Alice wants to send a message m to Bob secretly, i.e. Eve should not be able to understand what she sees on the channel. Therefore, Alice and Bob agree on a secret key-pair (only known to them) for encryption and decryption before the actual start of the transmission. An important property of secret-key primitives is that the encryption key can be easily derived from the decryption key and the other way round. However, in nearly all modern secret-key encryption primitives the encryption and decryption key are the same and therefore are denoted as key K. Before sending the message m, Alice encrypts it applying an encryption function E and using the secret key K. The result of this operation is called ciphertext c which is transmitted over the channel. Therefore, c = E(K, m). The size of the ciphertext is at least the size of the plaintext. This follows from the Pigeonhole Principle [11] which, converted to this scenario, states that if the output size of an encryption is smaller than its input size there must be necessarily 2 various inputs that lead to the same output which is not acceptable with encryption. On the other side, the size of ciphertexts could be bigger than the size of plaintexts if the plaintexts are padded before encryption for whatever reason. Bob, by applying the appropriate decryption function D and using the secret key K, recovers the original message m. Therefore, m = D(K, c). Eve, who also received the ciphertext, is unable to understand it because she is missing the secret information, i.e. the key, to decrypt it. Assuming a good secret-key encryption primitive, it is computationally infeasible to recover the message from the ciphertext without knowing the key. A question that is not covered in this scenario is how to exchange the key secretly. Unless Alice and Bob are unable to meet personally and need to exchange the key over a communication channel this may lead to a "Chicken and Egg problem" well known as the key distribution problem. However, there are smart solutions to this problem which may be gleaned in [3].

Note that although both concepts, encryption based on blocks and streams, also appear in public-key cryptography, the literature uses the names block cipher and stream cipher to denote secret-key encryption concepts.

Block ciphers form a special subset of secret-key ciphers where the message to be encrypted is divided into fixed length blocks. Each of these blocks, which are elements of the set of plaintexts, is transformed into an element of the set of ciphertexts. This happens under the influence of the secret key [1]. There is no change of size of the blocks during the transformation, i.e. ciphertext-blocks are of the same size as plaintext-blocks. The encryption is reversible, which means given the secret key it is also possible to recover plaintext blocks from ciphertext blocks. Block ciphers may be applied directly to messages with lengths equal to the size of a single block. If the size of the message is smaller than the block size padding is applied, i.e. additional symbols are added at the end of the message to fit the block size. Several padding techniques exist that allow distinguishing between message and padding. There are so-called modes of operation for messages longer than the block size [6].

Examples for block ciphers in use today are DES (data encryption standard), Triple DES (3DES) and AES (advanced encryption standard) [4]. DES was invented nearly 30 years ago and was one of the most widely used secret-key encryption algorithms. However, it is based on very short 64 bit blocks and 56 bit keys which, because of the increasing computational power available, seem not to be appropriate any more. There have been successful exhaustive key search attacks on DES already. Therefore, Triple DES (3DES) has been invented where DES is applied three times with different keys. Therefore, it has a key size of 168 bits but inherits the disadvantages of DES like the small block size of 64 bits [6]. Triple DES is sometimes used because it offers more security compared with DES and for legacy reasons (Triple DES can be executed on DES hardware). However, in the meantime AES was invented to replace DES and 3DES. It is based on a block size of 128 bits and a selectable key size of 128, 192 or 256 bits. So far, there have been no successful attacks to the AES algorithm apart from side channel attacks which are implementation attacks rather than attacks on the algorithm [12].

Stream ciphers can be understood as block ciphers with block size one and the additional feature that the encryption transformation changes with every symbol processed. An advantage of stream ciphers is that they may have limited or no error propagation which means they may be able to deal with flipped bits or even with missing or inserted bits depending on their specific implementation. Therefore, often they are a good choice if errors are highly probable in transmissions.

A very simple example should clarify the operation of stream ciphers. The so-called Vernam Cipher is a stream cipher for binary streams [3]. The inputs are a binary message stream $m_1m_2m_3\ldots m_t$ and a so-called binary key stream $k_1k_2k_3\ldots k_t$. The binary key stream is a random binary sequence of the appropriate length, i.e. the length of the message stream. For encryption each binary symbol of the message stream is XOR-ed with the corresponding binary symbol of the key stream, i.e. $c_i = m_i$ XOR k_i. Obviously, to decrypt the ciphertext, the process has to be repeated, i.e. $m_i = c_i$ XOR k_i. The Vernam Cipher is exactly what we called one-time pad in section 2.4. It provides unconditional security (assumed the key stream is a truly random sequence with the same length as the message) but is hardly used in practice because of the large key size. However, stream ciphers used in practice are based on the one-time pad with the difference that the key stream is generated from a short random sequence by a deterministic algorithm. Thus, only the short random sequence has to be exchanged by the trusted parties.

As discussed above, practical stream ciphers require generators to create pseudo-random key-streams based on keys short enough to be exchanged conveniently. Often so-called linear feedback shift registers (LFSR) are used for this purpose. They exist of a series of delay blocks which are most often initialized with the secret key. Triggered by a clock signal the contents of the delay blocks are shifted. The input of the first delay block is a XOR of a subset of the delay blocks. Fig. 4 shows an example of an LFSR.

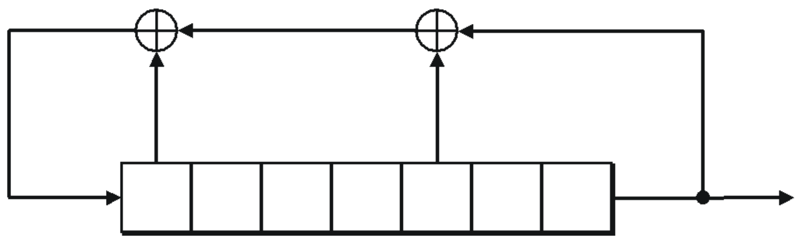

Fig. 4: Linear Feedback Shift Register [13]

Advantages of LFSRs are that they are easy to implement in hardware and software and they can produce sequences of large periods with reasonably good statistical properties [3]. The longest unique sequences are generated by so-called maximum length LFSRs. The period length of such LFSRs equals the maximum number of states that can be represented by a certain number of bits minus the zero state. Hence, a 3 bit maximum length LFSR has an output period length of 2^3-1 = 7 unless it is initialized with 3 zeros.

However, with time mathematical methods have been developed to analyse LFSRs and they are not considered secure any longer [13]. Nevertheless, they form the building blocks of more secure key-stream generators in use today called nonlinear feedback shift registers (NLFSR).

NLFSRs generally consist of LFSRs as building blocks used in combination with methodologies that destroy the linearity of the output of simple LFSRs. In [3] three methods are discussed. So-called **nonlinear combination generators** apply a nonlinear function that combines the outputs of two or more LFSRs. An example for a nonlinear combination generator is the “Geffe generator” consisting of three maximum length LFSRs of pairwise relatively prime period lengths that are combined applying the function x_1x_2 XOR x_2x_3 XOR x_3. **Nonlinear filter generators** consist of just one maximum length LFSR which output is generated by a nonlinear combination of several stages of the LFSR. The generators discussed so far are clocked regularly, i.e. at each time step each LFSR is clocked. So-called **clock-controlled generators** introduce nonlinearity by clocking downstream LFSRs based on the state of upstream LFSRs. An example is the “alternating step generator” which consists of three LFSRs. The output of this generator is the XOR result of the output of two LFSRs which are clocked depending on the output of the third LFSR. If the output of the third LFSR is 1, one of the other two LFSRs is clocked, if it is 0 the other one is clocked.

Although there is no real standard for stream ciphers so far, RC4 is most widely used and can be seen as de-facto standard [14]. As with most stream ciphers, it is based on one-time pad with a pseudo-random key-stream generator. Other than using LFSRs the key-stream generator is designed to be easily implemented in software. RC4 is in very heavy use today. It is the cipher used in WEP and WPA and may be optionally selected to be used in SSL (secure socket layer) and SSH (secure shell). Nevertheless, there are known attacks with the best of them being able to distinguish a random sequence from the pseudo-random sequence generated by RC4 given about one gigabyte of output data [15].

When comparing stream ciphers with block ciphers, there are clear practical advantages of stream ciphers. They are much easier to implement in software and hardware, generally they are faster, they do not require large memories to store blocks and they can deal with errors in the way that there is no error propagation. However, relatively few fully specified stream ciphers are published in the literature, and as opposed to block ciphers there are no standardised stream ciphers so far [3], [14].

The secret-key primitives described so far ensure that Eve is unable to understand, what is transmitted over the channel. Ciphers do not provide security against alteration of messages.

2.5.2.2. Message Authentication Codes

In order to be able to detect changes of messages transmitted over insecure channels so-called message authentication codes (MAC) or cryptographic checksums are used [2]. They can be understood as hash functions on data that includes a secret key.

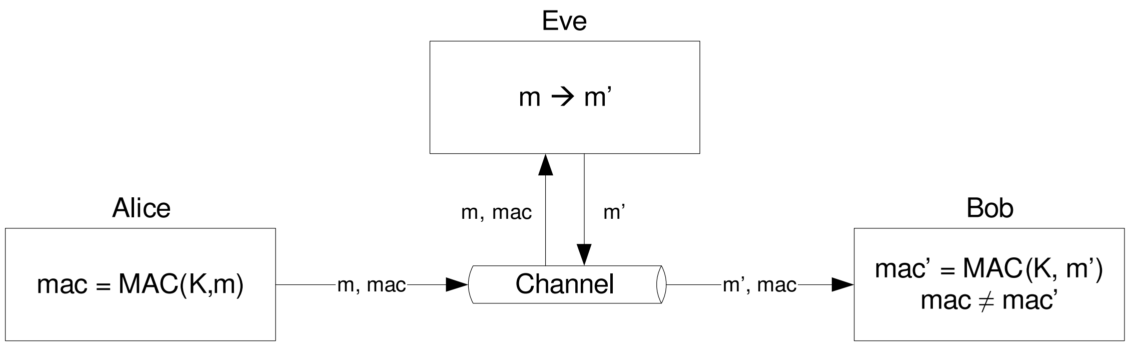

Fig. 5: Message Authentication Code

Fig. 5 shows the principle of MACs. Note that the example does not consider encryption of the message. Before sending the message, Alice generates a message authentication code mac applying a MAC-function which processes the message m and a secret key K. Then, Alice transmits both, the message and the message authentication code. Bob receives them and also generates the message authentication code using the same MAC-function which processes the received message and the shared secret key. Thereafter, Bob compares the MAC he generated with the MAC he received along with the message. If they match, because of the shared secret key, Bob knows that the message was not altered and he knows that the message was sent by Alice. In the figure above, Eve, who also receives the message and the MAC, can understand and alter the message but there is no way to change it prior to forwarding it to Bob without Bob knowing that something went wrong. Additionally, Eve cannot create messages and send them to Bob as if she were Alice [6]. Therefore, message authentication codes provide data integrity and data origin authentication [3].

In practice there exist several types of MAC-functions. They might be based on block ciphers, like the DES-CBC MAC, be based on stream ciphers, be constructed from unkeyed hash functions applying a secret key, like the MD5 MAC, or they might be based on one-time pad cipher [16]. Construction of MAC-functions from unkeyed hash functions means that the original algorithm, like MD5, is altered to incorporate a secret key into the compression function [3].

In practice there exist several types of MAC-functions. They might be based on block ciphers, like the DES-CBC MAC, be based on stream ciphers, be constructed from unkeyed hash functions applying a secret key, like the MD5 MAC, or they might be based on one-time pad cipher [16]. Construction of MAC-functions from unkeyed hash functions means that the original algorithm, like MD5, is altered to incorporate a secret key into the compression function [3].

2.5.2.3. Identification Primitives

One of the most important classes of primitives in today's computer systems are identification techniques which are sometimes also called entity authentication or identity verification techniques [3]. The purpose of entity authentication techniques is to allow one party to gain assurance about the identity of another party. Thus, entity authentication tries to prevent impersonation attempts. When compared with message authentication techniques, entity authentication techniques typically involve no meaningful message but normally they are based on a real-time process, i.e. both parties are active at the same time).

Identification techniques may be divided into three categories. They may be based on something known by the identifying party, like a password or a secret key. They may be based on something possessed by the identifying party, like a magnetic stripe card or a smart card. Or, they may be based on something inherent to the identifying party, like voice, fingerprints or handwritten signature.

Typical applications of entity authentication include access control to resources, like information systems, sometimes accompanied with the need to track resource usage, e.g. for billing purposes.

Fixed passwords are a very simply scheme of entity authentication based on something known. They are shared secrets between users and an information technology system. For identification typically the system asks the user for a user-id and the appropriate password. If the data entered by the user matches the data stored in the system the identification was successful and the user is granted access. Different fixed password schemes may store the passwords either in plaintext, or in encrypted form. A major security problem of fixed passwords is the so-called replay attack, when an attacker records the password transmitted over a channel and replays it at a later time to be granted access to a system. For that reason fixed passwords often are refereed to as weak authentication [3]. Countermeasures include encryption of the channel or even better the use of one-time passwords.

There are different kinds of **one-time password** schemes. Two parties may either share a list of passwords using one after another, or they may sequentially update their password, or they may generate one-time passwords with the help of one-time functions [3].

The problem with both of the above described password schemes, fixed and one-time, is that the actual secrets are released by the party which tries to identify itself. Therefore, whenever an active attacker is able to gain access to the secret (e.g. the list of passwords in a one-time password scheme) it is subsequently able to impersonate the party to which the secret belongs to. **Challenge-response identification** schemes address this vulnerability. Rather than releasing the actual secret they generate a response which is based on the secret and a time-variant challenge. That is why challenge-response schemes are also known as strong authentication mechanisms [3]. The time-variant challenge is provided by the verifying party every time an unknown party wants to be identified. A challenge includes a so-called nonce being the time-variant parameter. Nonces could be, for example, pseudorandom numbers, sequence numbers, or time stamps. The use of nonces prevents replay attacks as described with fixed passwords. It should be mentioned that although challenge-response schemes often are based on secret-key cryptography there are also implementations based on public-key primitives.

2.5.3. Public-Key Primitives

In addition to secret-key primitives, so-called public-key primitives form the second large group of keyed cryptographic tools [3]. Public-key primitives are based on key pairs instead of single shared keys. Each key pair is made up of a public key and a private key that are linked together mathematically. As their names betray, one of the keys has to be kept secret by the owner and the other one is shared publicly. Because public-key primitives are based on two different keys, they are often called asymmetric-key primitives. The main motivation to have public-key primitives is that with secret-key primitives each pair of trusted parties has to share one secret key [6]. Since this is very complex, if there is a larger number of parties involved, public-key primitives provide an appropriate solution because only one key has to be shared publicly for each party. On the other side, public-key primitives are less efficient and that is why there is still a need for secret-key primitives. The following sections describe the two most important applications of public-key cryptographic tools, public-key ciphers and public-key signature schemes.

2.5.3.1. Public-Key Ciphers

As already mentioned with secret-key primitives, ciphers deal with the encryption and decryption of messages. Fig. 6 shows the basic principle of public-key encryption [6].

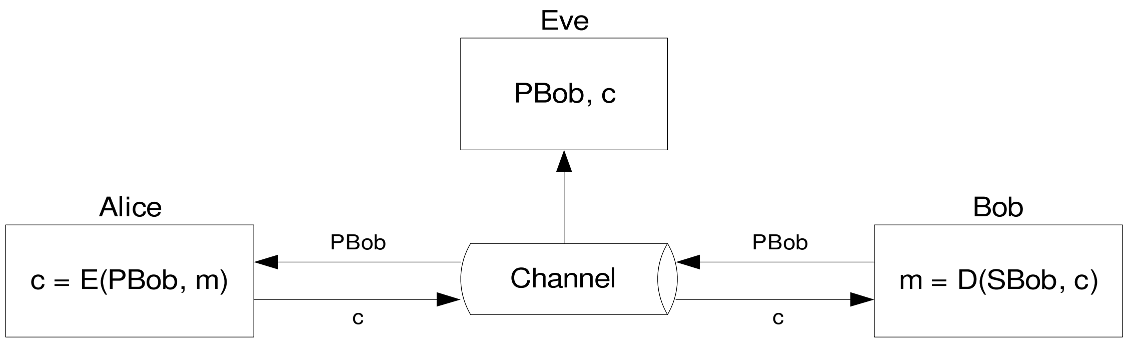

Fig. 6: Public-key Encryption

The key pair used in this example is the secret key of Bob (SBob) and the public key of Bob (PBob). As already mentioned earlier these keys are linked together mathematically in order to be able to use the public key for encryption and the private key for decryption. Therefore, a major premise for public-key cryptography is that it should be computationally infeasible to derive the secret key given a public key. Now, if Alice wants to send a message m to Bob secretly she encrypts it using the encryption function E under the influence of Bobs public key PBob. The resulting ciphertext c can be transferred over the insecure channel because only Bob is in possession of the secret key SBob which is necessary for decryption. Hence, in order to be able to read the received encrypted message c Bob decrypts it using the decryption function D under the influence of the secret key SBob. The example also shows Eve, the passive attacker, who is able to receive the public key PBob and the ciphertext c but cannot extract any useful information thereof.

It may now seem that the problem of secretly exchanging keys in secret-key cryptography is solved because public keys can be transferred over insecure channels. In fact, many practical systems are based on a mixture of secret- and asymmetric encryption. Often asymmetric-key encryption is used to agree on a shared secret short term key (session key) which further on is used in secret-key encryption to exchange the actual information. The term session key is based on the fact that it is used for a limited time (the session) only. This approach combines the advantages of both, the publicly shared key of public-key cryptography and the efficiency of secret-key cryptography [3]. However, there is a remaining issue with public-key cryptography. Since public keys normally are exchanged over insecure channels an active attacker would be able to impersonate another party by providing a public key which seems to belong to this other party but actually is part of a key pair of which the attacker is in possession of the private key. Consequently, the attacker would be able to decrypt messages which were intended for the impersonated party. For that reason public keys often are exchanged using a so-called public key infrastructure (PKI) which solves the impersonation problem by issuing certificates. Certificates basically store identities and corresponding public keys. Each certificate is digitally signed (see next section) by a trusted third party which consequently prevents impersonation attacks as described above. Ferguson and Schneier ([6]) provide further information about PKIs for the interested reader.

Probably the most important public-key technique is the **RSA** cryptosystem [3],[4]. It was invented in 1978 by R. Rivest, A. Shamir and L. Adleman and is based on the well known integer factorization problem. The idea is to multiply two sufficiently large prime numbers p and q to obtain n = p * q. Since it is believed (not proven) to be a hard mathematical problem to factorize n into its factors p and q, n is part of the public key and p and q are parts of the private key.

No practical attacks are known on the RSA cryptosystem provided it is based on sufficiently large keys (size of n). In respect to the computational power available, current references recommended to have keys of at least 2048 bits tending to 4096 or even 8192 bits for future applications.

The security of public-key cryptosystem is based on keys made up of very large integers. Usage of large keys, however, leads to less efficient execution of algorithms. Especially when considering mobile devices which are limited in computing power and energy efficient cryptosystems play an important role. Therefore, much attention is paid to **elliptic curve cryptography** (ECC) which offers public-key techniques that are much more efficient than traditional algorithms. ECC-calculations are based on points on elliptic curves. Since elliptic curves used in cryptography are defined in terms of modular arithmetic they only contain a limited number of points. The main operation used in ECC is called scalar point multiplication which means deriving a point P which satisfies the equation P = kQ for a given point Q and a given integer k. This is a one-way function, i.e. the security of ECC is based on the so-called elliptic curve discrete logarithm problem (ECDLP), namely finding a k in P = kQ for a given P and Q [1]. Solving the ECDLP is considered to be computationally infeasible if k is sufficiently large. Since the best known algorithm for solving the ECDLP shows exponential complexity key sizes of more than 224 bits are regarded to be sufficient large taking into account the computational power available at the moment [17].

2.5.3.2. Digital Signatures

Digital signatures are an essential cryptographic primitive in providing authentication, authorization and non-repudiation services. Signing primitives used in digital signatures provide a method to bind the identity of the signatory entity to the message to be transmitted.

While there are many digital signature algorithms, all of them are based on public key algorithms. That is there is some secret information that only the signing entity has knowledge of and there is some public information that allows any other entity to verify the signature. In this context the process of signing is called encrypting with the private key while the process of verification is called decrypting with a public key. Unfortunately the usage of terminology in signature schemes can be confusing when considered with that of public key ciphers.

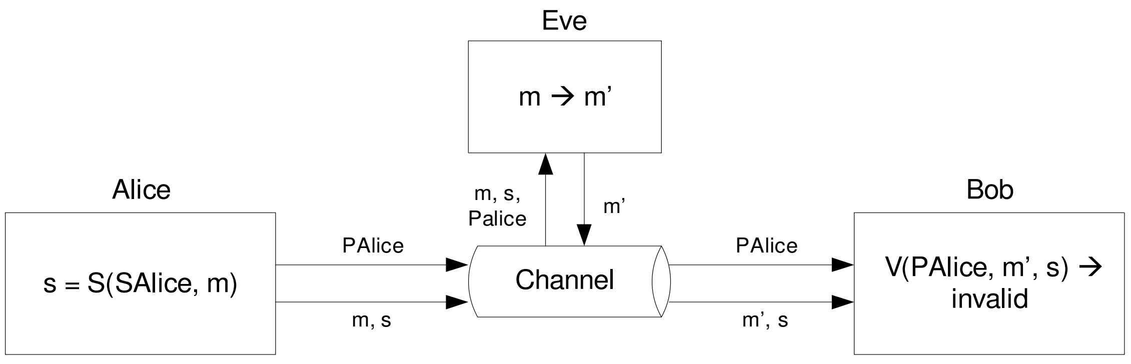

Fig. 7: Digital Signature

Fig. 7 shows the principle of a digital signature scheme [6]. We assume that Alice has already generated her key pair, i.e. SAlice, her private key, and PAlice, her public key. Whenever Alice wants to sign a message m she applies the signing algorithm S under the influence of her private key to this message with the result of a signature s. Subsequently, she distributes the message and the corresponding signature over the insecure channel. Bob, who wants to verify her message, applies the verification algorithm V under the influence of Alice's public key to the received message and the signature. The outcome of the verification could be either 'valid' or 'invalid'. 'Valid' would mean that the message was signed by Alice and it was not altered during its transmission. In our example Eve altered the message. Therefore, the result of Bobs verification is 'invalid'. Due to the fact that public-key techniques generally are less efficient, in practice often hash values of messages are signed rather than the actual messages.

One simple method of implementing a digital signature scheme is by using the RSA public key cipher where the encryption and decryption functions are both inverse operations of the other. This property is unique to the RSA cipher. Other examples are DSS (digital signature standard), ElGamal (named after its inventor Elgamal), and algorithms based on elliptic curve techniques. Obviously, a fact that all these algorithms share in common is that they are based on trapdoor functions with the trapdoor information being the private key [4].

The **ElGamal** digital signature scheme is partly based on the discrete exponential function and partly based on the Diffie-Hellman key agreement. The discrete exponential function was already covered in section 2.5.1 of this work. Since the description of the ElGamal scheme goes beyond the scope of this work the interested reader is adverted to [3] and [4] which provide detailed information about this topic and also cover the digital signature standard (**DSS**) which is another public-key technique used for digital signing and which is very similar to the ElGamal scheme.

2.6. Comparison of Secret-Key Primitives with Public-Key Primitives

Most of the advantages and disadvantages of secret-key primitives and public-key primitives were already discussed in the paragraphs above. This section is intended to summarize them and to provide a few additional notes.

The main advantages of secret-key primitives are that they are based on relatively short keys and that there are often efficient hardware implementations for them. Hence, they are applicable for processing large amounts of data. However, secret-key primitives come along with the disadvantage that a large number of keys have to be managed since each pair of trusted parties has to share an individual secret key. Additionally, it is considered as good practice to change the secret key regularly. This is to keep the amount of data being processed using a single key low in order to minimize the amount of input for potential attacks.

Public-key primitives, on the other side, have the advantage of easy key management. That is, only one key (the own private key) needs to be kept secret and public keys can be distributed over insecure channels. But most public-key techniques are based on very large key sizes which lead to less efficient execution of algorithms.

Considering the advantages and disadvantages of the large groups of primitives, today's cryptographic systems often are based on a mixture of both. These systems apply public-key cryptography to agree on so-called session keys which are shared secrets that are used for a limited time (the duration of a session) only. Subsequently, session keys are used in secret-key cryptography for processing the actual information. Therefore, the advantages of easy key management and efficient data processing have been combined in those mixed systems.

Elliptic curve cryptography, in future, may partially replace mixed systems since they combine the advantages of easy key management and fast data processing in one public-key scheme [18].

3. RFID Security and Privacy

3.1. Introduction

RFID has the potential to be efficiently used for object identification and most likely a majority of items in the supply chain will carry RFID labels in future. But object identification is not the only upcoming application where RFID tags will be deployed in masses. There are already first attempts of using tags in passports for efficient automated processing of biometric information. Additionally, ideas exist of utilizing RFID tags in order to prevent forgery like with banknotes [19]. According to these observations in all probability there will be a large number of RFID tags in our surroundings in the near future. Despite all the advantages RFID technology offers there are serious concerns about security and privacy as well.

The following sections focus on security and privacy risks coming along with RFID technology and present an overview of recent approaches described in the literature. While there are limitations on RFID solutions, some other alternatives are also presented in this section. The objective of this paper is as a source of information. An RFID solution, using one of the principles detailed in this paper is outlined in a companion paper entitled “One-Time Codes in RFID”. In addition, the scope of this section has been narrowed to security and privacy threats to the air interface protocol only, since this is the greatest risk of security. Other security risks are present past the reader interface, but these risks (on computing networks and databases) can be minimised by existing security protocols and practices.

3.2. Security and Privacy Threats

With the emerging RFID technology also a number of new security and privacy threats are coming up. Before dealing with these issues an overview should be provided about desirable security and privacy preserving features when dealing with RFID.

Ranasinghe et al. [20] identified the following security aims for RFID systems (in a general context these terms were already discussed in section 2.2 of this work):

‘Confidentiality’

‘Integrity’

‘Authentication’

‘Non-repudiation’

‘Availability’

Furthermore they identified the following two privacy-preserving goals:

‘Anonymity’

‘Unlinkability’

Based on the aims named above a number of threats and weaknesses of modern RFID systems pop up. In the following the term RFID system is focused on the EPCglobal Network [21] and the term air interface is used to denote the EPCglobal Class 1 Generation 2 UHF air interface specification [22]. Indeed, most of the declarations below are true for RFID systems in general but focusing on the EPCglobal Network comes along with the obstacles discussed in section 3.3 which provides the right context for further discussions.

Among the most important advantages of RFID technology are the non-contact and non-line-of-sight properties of its communication mechanism. At the same time, these advantages are the most important security and privacy threats of RFID systems since they allow quite easy eavesdropping of communications or even worse reading of tags of which both may happen without notice of the actual owners of tags [23].

The most obvious security threats appear at the wireless air interface of RFID systems. As with all sorts of wireless transmissions, communication between readers and tags can quite easily be monitored by third parties. Since the air interface, for most of the data being transferred, does not provide encryption the third party may gain useful information by analysing the monitored communication. Useful information can be EPCs which further may be used to create ‘faked’ (cloned) labels or data secretly stored on a tag. However, since we are dealing with passive backscattering tags it should be mentioned that it is assumed to be much harder to listen to the tag-to-reader communication based on backscattering than it is to listen to the reader-to-tag communication based on active transmission. This assumption is based on the difference of ranges. Fig. 8 should clarify this statement. Because the monitoring third party (eavesdropper in the figure), in order not to be exposed, often has to stay in some distance of the actual reader it is outside of the backward (tag-to-reader communication) range but inside of the forward (reader-to-tag communication) range.

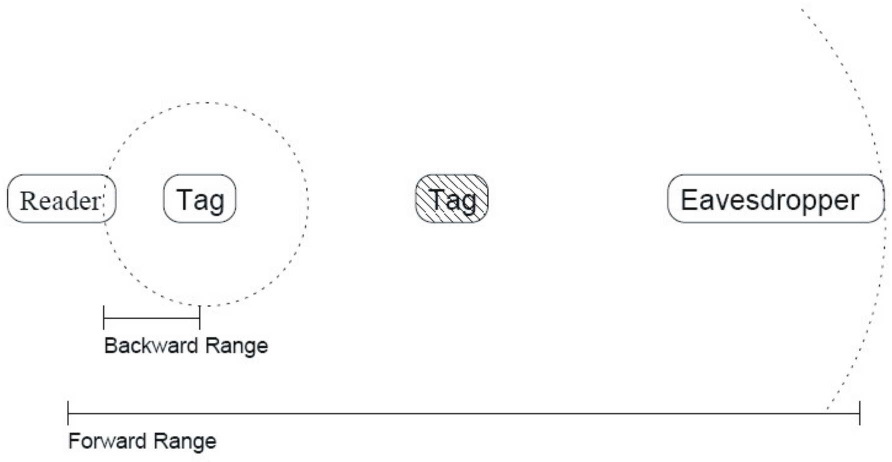

Fig. 8: Communication Ranges [24]

Besides passive listeners, active third parties form another threatening group to RFID systems. Active third parties may either be readers or tags. Since the air interface offers no approaches for authentication or integrity protection each reader may query each tag and the other way round each tag replies to queries of each reader provided that both use the same air interface. Several security threats arise from this weakness. Firstly, ‘fake’ labels may be introduced into the supply chain. In this context we want to understand ‘fake’ labels as tags that are introduced into the supply chain at some point from parties other than trusted producers or distributors who use tags to identify their products and transportation mediums. Fake labels may be used for identifying counterfeits as if they were originals. They may also be used for replacing original labels on items in order to give those items other identities maybe associated with cheaper prices to be payed at check-out points. In a more complex scenario, fake labels may be utilized for conducting a denial of service attack on a reader by placing a large number of labels into the range of the reader. Secondly, third parties might use readers in order to gain knowledge about items belonging to other parties. Examples would be retailers who somehow manage to inventory shelves of their competitors which may lead to useful business data, or marketers who scan the contents of their customer’s shopping bags which may provide information about their buying patterns. Even if it is not possible to gain knowledge about individual items, the constant serial numbers (EPCs) divulged by tags may serve as identifiers for tracking purposes.

It should be clear that there is a close connection between security and privacy. We discovered above how security weaknesses like a missing authentication mechanism lead to privacy threats like traceability. Therefore, approaches against security and privacy issues should primarily be targeted at security weaknesses.

Less obvious security threats to RFID systems are physical attacks and network and database attacks. Physical attacks denote attempts to gain valuable information from RFID systems where the attacker has physical access to one or more components of the system. For example, an attacker could try to discover the EPC of an RFID label by reverse engineering it. Network and database attacks refer to higher levels of RFID systems where, for example, data about tag reads is transmitted over network connections to servers or information belonging to EPCs is stored in databases. As with other conventional information systems based on networks and databases the main focus of attackers might be on trying to capture transmissions or breaking into databases.

While not a focus of this paper, in addition to the security aims outlined above, there are other privacy threats and attacks, some of which are:

- Association:
    - Personal identity associated with one or more tags
- Transaction:
    - Identity associated with a transaction, even if paying with cash
- Inventory:
    - Reading the tags/hence items on your person
- Location:
    - Identifying the location of tagged objects
- Preference:
    - Marketers generating a profile based on tagged goods

3.3. Secure RFID Obstacles

Admittedly, the solution to most of the threats described in the last section seems to be quite easy. Implementation of modern cryptographic tools into RFID systems would most likely instantly strengthen the security to a level as we know it from state-of-the-art network or database systems. Mutual authentication, for example, is a well known concept in cryptography which, applied in RFID systems, could stop reading attempts of malicious readers or tag replies from malicious tags. Hence, the question is why not to integrate cryptographic protocols into air interface standards. And the answer is quite simple. There are numerous limiting factors that prevent us from being able to do so. This statement especially applies to RFID technology used for supply chain applications. Ranasinghe et al. [20] identified those limitations and summarized them in a table.

Table 1: Low Cost RFID Challenges [20]

Challenge	Description
Cost	Storage limitation. Silicon area.
Regulations	Radiated power. Frequency of operations. Available bandwidth.
Power Consumption	Power consumption of the label IC circuit. Power required to operate E^2PROM.
Performance	Label performance and system performance goals.
Power disruptions	Sudden loss of power.

The most important limiting factor for many RFID systems is cost. For example, in order to be able to affix labels to billions of consumer items prices have to be kept at a minimum. The cost-dominating factor of labels is the silicon area. Therefore, in order to keep the silicon area small and the costs down only minimal functionality and memory may be implemented into the chip. Power consumption is another important factor. The cheapest tags are passive backscattering tags. Because they harvest their power from the field radiated by the reader and do not have a battery the available power for operating the chip is very limited. Other limiting factors include regulations which restrict radiated power, frequency ranges and ultimately available bandwidth, performance which should be high in order to be able to read several hundred tags per second, and power disruptions causing resets in passive tags which have to be considered during protocol design.

3.4. Research Survey

Numerous papers have been published dealing with security and privacy issues in RFID and proposing solutions to overcome these issues considering the obstacles described in the previous section. In this section we will describe the most interesting approaches and provide links with works associated to these approaches like analysis, proofs or enhancements. In addition, the interested reader should be adverted to the bibliography on security and privacy issues in RFID systems from Gildas Avoine [25] which is a very comprehensive collection of references to papers dealing with this topic and to the paper of Juels [26] which is another recent research survey of the topic.

3.4.1. Tag Killing

One of the simplest approaches of protecting consumer privacy is to render tags silent at the point of sale. The current EPCglobal Class 1 Generation 2 UHF air interface standard [22] implements this idea which leads to perfect privacy. However, as already discussed by Juels et al. in 2003 [27] this approach is not sufficient. Firstly, it does not provide any privacy protection before a tag passes the point of sale. This means there is no protection for retailers against espionage. Secondly, often it is not wanted or impossible to kill a tag because its functionality is required after the point of sale. For example, tags may be used to identify articles if returned as defective and the information provided by tags may be required for recycling purposes of a product. RFID tags identifying books in libraries are another example. These tags cannot be killed when books leave the library because at the time a book is returned the corresponding tag is required to identify the book again. However, what comes in mind here leads us to the next section ‘Sleeping Tags’.

3.4.2. Sleeping Tags

With the drawbacks of killing tags in mind another idea of rendering tags silent would be to put them into a sleep state with the opportunity of waking them up at a later time using some kind of wake-up password. Juels discussed this method in his research survey in 2005 [26]. This approach comes along with some hurdles. Since having a general password is never a good idea a dedicated wake-up password has to be assigned to each tag. This leads to the requirement of a management system for those passwords which at a first glance does not seem to be too bad. However, since tags are rendered silent there is no straightforward possibility of associating the right tag with the right wake-up password by just using the air interface. Therefore, another identification technique, like a bar code, is required in order to identify the tag and wake it up. This however almost leads to a situation where tags are no longer required because bar codes are in use again. To express it more moderately, at least some of the advantages of RFID systems are lost by applying this approach.

3.4.3. Blocker Tags vs. Tag Proxies

A different approach of privacy enhancement leading to the same result, namely preventing readers from inventorying tags, are so-called ‘blocker tags’ and their counterparts ‘tag proxies’.

In 2003 Juels et al. presented a work about blocker tags[27]. Blocker tags are small privacy enhancing devices each simulating a number of ordinary tags. Thus, when carried by a customer a blocker tag by continuously responding to readers prevents those readers from reading the actual tags carried by the customer. Blocker tags can be designed to simulate all possible tags (IDs respectively) or a selected subset of tags (IDs respectively). This is possible because the idea of blocker tags is based on the tree-walking protocol, a multiple tag reading mechanism based on binary trees. When using this mechanism, tag identifiers sharing the same prefix are located in the same sub-tree. Hence, by simulating only a sub-tree blocker tags may block only a selected subset of tags, e.g. tags belonging to one manufacturer. Indeed, the privacy conserving feature of blocker tags might be exploited by attackers to mount denial-of-service attacks against readers. Although Juels et al. identified this threat they did not provide a sufficient solution against it.

Rieback et al. in 2005 further developed the idea of blocker tags and proposed an approach called ‘selective RFID jamming’ [28]. This approach is based on active jamming devices which according to so-called access control lists (ACL) selectively block tag responses to queries of readers that are not granted access. In order to do so a jamming device continuously monitors reader commands and transmits jamming signals long enough to block tag responses if it detects unallowed queries. Since reader commands do not contain sequences identifying individual readers and therefore a jamming device cannot straightforwardly differentiate between readers that are allowed to query tags and those that are not so allowed, the work of Rieback et al. proposes authentication mechanisms between readers and jamming devices. Additionally, according to the authors, the jamming signal is resistant against digital signal analysis which means it cannot be simply averaged out. The main weaknesses of this proposal, however, are that an additional active device is required in order to ensure privacy and that the problem of possible denial-of-service still exists.

A complementary solution to blocker tags or jamming devices was presented by Juels et al. in 2005 [29]. Their work deals with devices that actively simulate RFID tags under their control, so-called RFID Enhancer Proxies (REP). A REP is a sophisticated device able to enhance privacy by acquiring tags into a group, relabelling them randomly and further on responding on behalf of each of the tags in the group. Since the real tags cannot any longer be identified by readers (they are relabelled randomly) all identification information has to be released by the REP, and since the REP is a powerful device it can execute privacy policies. The obvious weakness of this approach is that for privacy protection each group of tags has to be equipped with a REP which seems not to be very practical. A similar approach to the REP was proposed by Rieback et al. in the same year [30]. Their device is denoted 'RFID Guardian'.

3.4.4. Hash Locking

One of the very first publications dealing with RFID security and privacy was presented by Sarma et al. in 2002 [23]. They identified main security and privacy threats and limitations that prevent users from simply applying traditional cryptographic tools like private-key or public-key encryption schemes in RFID systems. Their approach, the so-called hash-lock scheme, is based on a hash value, calculated from a random key, which is issued by a reader to lock a tag. In the locked state a tag only responds with this hash value (which serves as meta-ID). In order to unlock the tag a reader has to send the original key whereupon the tag hashes this key and compares it with the stored meta-ID. If the values match the tag unlocks itself otherwise it remains in the locked state. The hash-lock scheme provides basic reader-to-tag authentication but does not solve the main privacy concerns since the meta-ID replied by a tag in the locked state can be used for tracing.

Weis et al. in 2003 presented an enhancement of the hash-lock scheme called the randomized hash-lock scheme [24]. In difference to the former scheme the later always replies with random meta-IDs. It does this by hashing a concatenation of the locking key and a randomly generated number and replying the randomly generated number and the hash code. The reader thereafter by applying a brute-force search over all its known keys reveals the right key for unlocking the tag. Although this scheme theoretically may be a possible approach to overcome privacy issues it is impractical. Since the complexity of brute-force searches in databases grows with the number of entries this scheme is restricted to a small number of keys and therefore to a small number of tags to be managed.

Avoine analysed the random hash-lock proposal of Weis et al. in 2005 [31]. He came to the conclusion that their protocol provides location security as long as the attacker has no chance to tamper with the tag. If, however, the attacker tampers with the tag it might be able to reveal the static tag identifier which might further on allow it to reconstruct former events of the tag.

In 2005 Nohara et al. presented their work on a ‘K-steps ID matching scheme’ [32]. Their approach is based on tag identifiers which can be represented as leaves of a binary tree of depth K (the reason for the name of the scheme). They propose a protocol which reduces search complexity at the back-end database by utilizing the special properties of the tree representation. The idea was incorporated with the randomized hash-lock scheme of Weis et al. [24] with the result of an improvement of search complexity from O(n) to O(log n) (where n is the number of tags known to the database). The disadvantage of this scheme, however, is that the number of hash calculations required in tags is increased from O(1) to O(log n).

3.4.5. Hash Chains

A privacy protection scheme very similar to the random hash-lock of Weis et al. was proposed by Ohkubo et al. in 2003 [33]. Their approach is based on two different hash functions and a back-end database. Tags are initialized with a random number different from the tag identifier. Both, the tag identifier and the initial random number are stored in the back-end database. Every time a tag is queried by a reader it calculates a hash value using the hash function G on the stored number (the initial random number if it is the first query of a tag) and replies with the hash code. Internally, the tag applies another hash function H on the same stored number and replaces this number by the result of this hash calculation. This is called the hash chaining mechanism. This mechanism ensures that tag responses are different for consecutive queries. In order to infer the tag ID the back-end database has to do an exhaustive search over all its entries considering that the number of hashing operations applied on the initial random number is unknown. This, because of the high computational effort required, would only be possible with a restricted number of entries in the database and therefore a restricted number of tags to be managed.

Avoine and Oechslin [34] analysed the approach of Ohkubo et al. in 2005 and came to the conclusion that it is provably secure but not scalable because of the high computational effort required during identification. They presented an enhancement to the original approach of Ohkubo et al. providing a time-memory trade-off which basically means that hash values are calculated in advance and stored in the database for efficient look-up.

Yeo and Kin [35] presented in the same year another enhancement to the work of Ohkubo et al. They also considered the non-scalability of the initial approach and proposed grouping of tags and pre-computation in order to allow efficient identification of tags at the database level.

3.4.6. Re-Encryption

Re-encryption often denotes a cryptographic concept that allows a third party altering of a ciphertext that was initially encrypted under the public key of party A with the outcome of a ciphertext that is subsequently encrypted under the public key of party B. It is important to note that this concept does not reveal the corresponding plaintext to the third party.

In RFID the term re-encryption sometimes is used to denote a concept of changing ciphertexts without the knowledge of public keys. Though, different from the general concept the ciphertext remains encrypted under the same public key and therefore still belongs to the same recipient.

A very comprehensive privacy enhancing approach based on re-encryption was presented by Golle et al. in 2004 [36]. The overall system assumes identifiers to be stored as ciphertexts in tags. These ciphertexts are generated by encryption using a cryptosystem based on the ElGamal scheme. This special cryptosystem allows re-encryption of ciphertexts in tags without the knowledge of public keys. Therefore, so-called privacy-enhancing agents can be used to change ciphertexts in tags without knowing the corresponding plaintexts or public-keys. Given that such privacy-enhancing agents are positioned along the paths of tag movements they may change ciphertexts in tags and therefore provide protection against tracing. One disadvantage of this system lies in the linear number of decryption operations required by a trusted reader to identify a tag.

Saito et al. [37] analysed the re-encryption approach proposed by Golle et al. [36]. They came to the conclusion that this approach is vulnerable to active attackers. Since there is no restriction in write-access attackers may change the memory contents of RFID tags. Thus, attackers might store ciphertexts encrypted under their own public-key in tags which allows them to track those tags subsequently. Saito et al. presented two proposals to overcome this issue.

Avoine [31] also analysed the approach of Golle et al. [36]. He came to the conclusion that it does not provide location privacy because attackers may track tags simply by eavesdropping communications between privacy-enhancing clients and tags. Avoine also showed that the proposed improvements of Saito et al. [37] have weaknesses which might lead to tag traceability.

3.4.7. Banknote Privacy Protection

Juels and Pappu in 2003 proposed a system for privacy protection in banknotes [19]. The system primarily is based on a combination of optical and RF access to banknotes, i.e. one part of the information is printed on a banknote to be read out optically (e.g. barcodes) and another part of the information is stored in an RFID tag implemented in the banknote to be read out via an air interface. Since the information obtained optically is required to gain full access to the information available over the air interface this approach limits the ability of attackers to read out tags while they pass by. Both a public-key encryption scheme and a digital signature scheme are applied to provide message confidentiality and message origin authentication. Another important fact is that, considering the limited computational abilities of tags, all the cryptographic operations take place in readers and tags only store encrypted messages or digital signatures. In order to provide basic privacy protection the system applies re-encryption of the banknotes serial number resulting in changing ciphertexts stored in the banknote. Therefore, the ciphertext cannot be used as a meta-ID for tracing. In their conclusion Juels and Pappu point out that their approach still has weaknesses but might provide helpful aspects for further developments.

In 2004 Avoine [38] published a work which proved that the banknote security scheme proposed by Juels and Pappu has weaknesses that allow attackers to modify data in the tag without optical access. Additionally, he showed that the access key used in the scheme can be used for tracing purposes.

In 2005 Zhang and King [39] also focused on the weaknesses of the work published by Juels and Pappu and presented an improved version of the privacy protection scheme for banknotes. According to Zhang and King their version offers perfect integrity of data for law-enforcement and addresses a cryptographic attack on the original protocol.

3.4.8. Noisy Tags

Castellucia and Avoine presented in 2006 their work on a key exchange protocol for RFID [40]. Their approach is based on so-called noisy tags which belong to the system and are in the presence of the interrogation zone of a trusted reader. The idea is to generate noise during tag responses. The pattern of the noise is known to the trusted reader and therefore can be subtracted in order to recover tag responses. Unlike the trusted reader an eavesdropper cannot understand tag responses because it cannot differentiate between noise and meaningful data.

3.4.9. Physically Uncloneable Functions

Ranasinghe et al. in 2004 presented a paper including various proposals for security and privacy enhancements for low-cost RFID [41]. Their work is based on the assumption that low-cost RFID probably does not allow integration of perfect security but it should be possible to meet the security required by typical applications. Their most important approach is an authentication mechanism for RFID systems based on so-called PUFs (physically unclonable functions). PUFs were already described in section 2.5.1 of this work with random number generators. In the RFID secrecy context we discuss here PUFs are used to replace traditional cryptographic tools in strong authentication protocols. The underlying assumption is that PUFs provide tags with a uniqueness that can be compared with a secret key in traditional cryptography.

Tuyls and Batina published in 2006 their work which very detailed describes the usage of PUFs for authentication of RFID tags [42]. They provide a general discussion about key extraction from PUFs, an authentication protocol and an off-line authentication mechanism which is based on elliptic curve cryptography implemented in the tag.

3.4.10. Lightweight Cryptography Implementations

Feldhofer et al. in 2004 presented their work on a hardware implementation of the AES algorithm that fits low-cost RFID tags for strong tag to reader authentication [43]. The estimated hardware complexity of the implementation is 3600 gates and the estimated power consumption is 8µA. A drawback of the implementation however is the number of clock cycles required to encrypt a block of 128 bits which is about 1000. Feldhofer et al. additionally proposed an enhancement for the ISO/IEC 18000 standard in order to integrate the authentication mechanism and to deal with the long encryption time in the tag.

Another lightweight block-cipher hardware implementation was presented by Lim and Korkishko in 2005 [44]. Their 'mCrypton' was especially designed for RFID and sensor applications and shows a hardware complexity of 3500 to 4100 gates depending on the key size (64 to 128 bits). Clear statements of the power consumption and execution time however are missing in the paper.

Wolkerstorfer in 2005 presented a hardware implementation of an elliptic curve cryptography processor and examined the feasibility of implementing it into low-cost RFID tags. Although the proposed ECC processor only allocates 1.3 mm^2 (0.35µ CMOS), which might fit an RFID tag, there are concerns about the power consumption and the execution time. According to Wolkerstorfer, in order to meet an estimated power budget of 30µW one elliptic curve operation would require about 11 seconds to finish (192 bit processor).

3.4.11. Various Mutual Authentication Schemes

Although it is not possible to sharply differentiate between approaches that aim at privacy protection and approaches that aim at authentication, as those approaches often are mixed together, most of the material discussed at the beginning of section 3.4 focuses on privacy protection. Sections 3.4.8 to 3.4.10 introduced topics closely related to authentication and in this section we would like to present approaches with a clear focus on mutual authentication.

Juels in 2004 presented a mutual authentication scheme [45]. In this scheme prior to the tag-to-reader authentication the reader has to authenticate to the tag. This procedure prevents simple replay attacks in the way that attackers cannot use their readers to learn about valid responses from authentic tags. Eavesdropping of communications between authentic parties is still possible but does not lead to any useful information since authentication sequences are used only once. New sequences for consecutive authentication runs are exchanged using a sophisticated one-time pad technique based on pads from multiple authentication sessions.

Lee et al. in 2005 presented a mutual authentication scheme for low-cost RFID tags [46]. The scheme is based on random nonces, hash functions and a back-end database keeping track of authentic tags. In order to provide location privacy the scheme proposes updates of tag identifiers after successful authentication processes. In the same year Dimitriou presented a very similar scheme [47]. Both proposals use hash values of the actual tag identifiers as database keys. This saves us from the need of exhaustive database searches in order to find identifiers. However, it conflicts with the provision of real location privacy since malicious readers can track a tag based on this hash value until it next is identified by an authentic reader (which then causes a change of the identifier).

Kang and Nyang in 2005 addressed several issues of authentication protocols for RFID [48]. Among others they indicated the risks of unterminated session attacks on tags and denial-of-service attacks on back-end databases. Unterminated session attacks denote authentication sessions that are deliberately not terminated by malicious readers. Because both, the tag and the database, update the stored identifier at the end of an authentication session unterminated session queries result in constant meta-IDs being replied by tags which can be regarded as attack on location privacy. Denial-of-service attacks on back-end databases are based on the fact that most RFID authentication protocols require high computational effort at the database level. Sending numerous malicious queries to a database therefore could lead to overstressing this database.

In 2006 Duc et al. published a paper which proposes an authentication scheme for EPCglobal Class 1 Generation 2 tags [49]. The scheme is based on random nonces and CRC codes. Although their approach does not allow tracing by the use of unterminated session attacks the use of CRC codes instead of hash codes is highly questionable because CRC codes naturally do not provide the same security as hash codes in terms of collision resistance. Another weakness of the protocol is again the O(n) computational effort required at the back-end database in order to identify a tag.

Chatmon et al. in 2006 published a work including two different authentication protocols for low-cost RFID [50]. Both protocols are optimistic which means they are secure and efficient as long as the adversary is passive. If the protocols face an active adversary, like one that implements unterminated session attacks, the security still remains but efficiency gets lost which means the server has to do an exhaustive search for each tag that should be identified but was attacked since the former successful identification. Both algorithms are based on random nonces from tags and reader and a keyed hash function. There difference between the two proposed algorithms is that one of them uses timestamps as nonces additionally to random nonces. In a later paper [51] Burmester (one of the authors of the original paper) et al. proved that both algorithms guarantee anonymity, availability and secure authentication provided that single trusted tags can only participate in one authentication session at a time.

While this section focussed on mutual authentication, tag-to-reader authentication is relevant to anti-counterfeiting solutions to secure the supply chain.

3.4.12. Multilayer Traceability

Avoine and Oechslin in 2005 [52] presented an interesting paper which showed no new solutions but highlighted a delicate topic. According to them the traceability issue of RFID tags is not only restricted to eavesdropping data transfer or actively querying tags but is a problem that needs to be addressed on multiple layers. Former works only considered the application layer on which data like tag identifiers is being transferred. The work of Avoine and Ochslin points out that also the communication layer and the physical layer have to be considered. The communication layer deals with mechanisms that allow readers and tags to communicate like collision avoidance techniques. The physical layer defines the actual wireless communication interface including frequency, modulation and data representation. Avoine and Oechslin indicate ways of how different layers might be used for the tracing of RFID tags.

4. Conclusions

This paper provides a comprehensive overview of state-of-the-art cryptography on the one hand and security and privacy issues in RFID on the other hand. Starting with information technology security it covers desirable security services, attacks and security models in general. Divided into the three groups unkeyed, secret-key and public-key modern cryptographic primitives are presented. The second part focuses on security and privacy concerns arising with large scale introduction of RFID in the supply chain. The most important among these concerns are traceability and cloneability of tags.

Although ordinary cryptographic primitives offer aid to overcome these threats, like authentication mechanisms based on secret-key or public-key primitives, resource constraints impede us from implementing most of the ordinary cryptographic tools. RFID tags are very restricted in available operating power and chip size and unfortunately most of the cryptographic primitives require both. Hence, there is a need for new lightweight cryptographic primitives to be used in RFID technology.

Such lightweight primitives should be based on the well established knowledge on cryptography but tailor the services on the requirements and restraints of RFID. The research survey in this paper shows that researchers work on this topic since around 2003. Different approaches were proposed in the meantime. Most of them are based on the change of tag identifiers somehow combined with authentication schemes, which cover both traceability and cloneability concerns. But also other approaches are presented, not always based on cryptography, like schemes that permanently or temporary silence tags or schemes that block tags from being read by distracting transmissions.

Acknowledgement

The authors of this paper would like to thank Alfio Grasso, Raja Ghosal, Damith Ranasinghe, Kin Seong Leong, Mun Leng Ng and Behnam Jamali for their help.

References

[1] Oswald, E.; Lecture Notes: IT Security; Institute for Applied Information Processing and Communications, Graz University of Technology, Austria; 2005.

[2] Stallings, W.; Network and Internetwork Security: Principles and Practise; Prentice-Hall; New Jersey; 1995.

[3] Menezes, A., van Oorschot, P. and Vanstone, S.; Handbook of Applied Cryptography; CRC Press; Boca Raton; 1997.

[4] Delfs, H. and Knebl, H.; Introduction to Cryptography: Principles and Applications; Springer; Berlin, Heidelberg, New York; 2002.

[5] Oswald, E.; Lecture Notes: Introduction to Information Security; Institute for Applied Information Processing and Communications, Graz University of Technology, Austria; 2004.

[6] Ferguson, N. and Schneier, B.; Practical Cryptography; Wiley Publishing; Indianapolis; 2003.

[7] Klima, V.; Tunnels in Hash Functions: MD5 Collisions Within a Minute; Cryptology ePrint Archive; Retrieved May, 4, 2006 from http://eprint.iacr.org/2006/105.pdf.

[8] Schneier, B.; New Cryptanalytic Results Against SHA-1; Weblog: Schneier on Security; 2005; Retrieved May, 4, 2006 from http://www.schneier.com/blog/archives/2005/08/new_cryptanalyt.html.

[9] Goldreich, O; Foundations of Cryptography: Basic Tools; Cambridge University Press; Cambridge; 2001.

[10] Ranasinghe, D., Lim, D., Devadas, S., Abbott, D. and Cole, P.; Random numbers from metastability and thermal noise; IEE Electronic Letters; Volume 41; Issue 16; Pages 13-14; 2005.

[11] Grimaldi, R.P.; Discrete and Combinatorial Mathematics: An Applied Introduction; 4th ed. pp. 244–248,1998.

[12] AES Lounge; AES Security; Retrieved April, 3, 2006 from http://www.iaik.tu-graz.ac.at/research/krypto/AES/index.php#security.

[13] RSA Laboratories; What is a linear feedback shift register?; Retrieved April, 4, 2006 from http://www.rsasecurity.com/rsalabs/node.asp?id=2175.

[14] RSA Laboratories; What is a stream cipher?; Retrieved April, 4, 2006 from http://www.rsasecurity.com/rsalabs/node.asp?id=2174.

[15] Fluhrer, S.R. and McGrew, D.A.; Statistical Analysis of the Alleged RC4 Keystream Generator; pp19–30, FSE 2000.

[16] RSA Laboratories; What are Message Authentication Codes?, Retrieved April, 5, 2006 from http://www.rsasecurity.com/rsalabs/node.asp?id=2177.

[17] Chang, S., Eberle, H., Gupta, V. and Gura, N.; Elliptic Curve Cryptography – How it Works; Sun Microsystems Laboratories; 2004; Retrieved May, 9, 2006 from http://research.sun.com/projects/crypto/.

[18] Gura, N., Shantz, S., Eberle, H., et al.; An End-to End Systems Approach to Elliptic Curve Cryptography; Sun Microsystems Laboratories; 2002; Retrieved May, 10, 2006 from http://research.sun.com/projects/crypto.

[19] Juels, A. and Pappu, R.; Squealing Euros: Privacy Protection in RFID-Enabled Banknotes; in Financial Cryptography; Springer Lecture Notes in Computer Science; Volume 2742; Pages 103-121; 2003.

[20] Ranasinghe, D., Engels, D. and Cole, P.; Low-Cost RFID Systems: Confronting Security and Privacy; White Paper Series; AutoID Labs; Edition 1; 2005.

[21] EPCglobal; The EPCglobal Network; Retrieved March, 30, 2006 from http://www.epcglobalinc.org/about/EPCglobal_Network.pdf.

[22] EPCglobal; EPC Radio-Frequency Identity Protocols, Class-1 Generation- 2 UHF RFID, Protocol for communications at 860 MHz – 960 MHz, Version 1.0.9; January 2005; Retrieved May, 12, 2006 from http://www.epcglobalinc.org/standards_technology/ratifiedStandards.html.

[23] Sarma, S., Weis, S. and Engels, D.; RFID Systems and Security and Privacy Implications; in Proceedings of the 4th International Workshop on Cryptographic Hardware and Embedded Systems; Springer Lecture Notes in Computer Science; Volume 2523; Pages 454-469; 2003.

[24] Weis, S., Sarma, S., Rivest, R. and Engels, D.; Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems; in Proceedings of the First International Conference on Security in Pervasive Computing; Springer Lecture Notes in Computer Science; Volume 2802; Pages 201-212; 2003.

[25] Avoine, G.; Bibliography on Security and Privacy in RFID Systems; MIT; Cambridge, Massachusetts; Retrieved May, 17, 2006 from http://lasecwww.epfl.ch/~gavoine/rfid/.

[26] Juels, A.; RFID Security and Privacy: A Research Survey; RSA Laboratories; 2005; Retrieved May, 26, 2006 from http://www.rsasecurity.com/rsalabs/staff/bios/ajuels/publications/pdfs/rfid_survey_28_09_05.pdf.

[27] Juels, A., Rivest, R. and Szydlo, M.; The Blocker Tag: Selective Blocking of RFID Tags for Consumer Privacy; in Proceedings of the 10th ACM Conference on Computer and Communications Security; ACM Press; Pages 103-111; 2003.

[28] Rieback, M., Crispo, B. and Tanenbaum A.; Keep on Blockin' in the Free World: Personal Access Control for Low-Cost RFID Tags; in Proceedings of the 13th International Workshop on Security Protocols; Cambridge; 2005.

[29] Juels, A., Syverson, P. and Bailey, D.; High-power Proxies for Enhancing RFID Privacy and Utility; in Proceedings of the 5th Workshop on Privacy Enhancing Technologies; Dubrovnik; 2005.

[30] Rieback, M., Crispo, B. and Tanenbaum A.; RFID Guardian: A Battery-Powered Mobile Device for RFID Privacy Management; in Proceedings of the 10th Australasian Conference on Information Security and Privacy; Springer Lecture Notes in Computer Science; Volume 3574; Pages 184-194; 2005.

[31] Avoine, G.; Adversary Model for Radio Frequency Identification; Technical Report; Security and Cryptography Laboratory, Swiss Federal Institute of Technology; Lausanne; 2005.

[32] Nohara, Y., Inoue, S., Baba, K. and Yasuura, H.; Quantitative Evaluation of Unlinkable ID Matching Schemes; in Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society; ACM Press; Pages 55-60; 2005.

[33] Ohkubo, M., Suzuki, K. and Kinoshita, S.; Cryptographic Approach to 'Privacy-Friendly' Tags; in Proceedings of the RFID Privacy Workshop; MIT; Cambridge, Massachusetts; 2003.

[34] Avoine, G. and Oechslin, P.; A Scalable and Provably Secure Hash-Based RFID Protocol; in Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications Workshops; IEEE Computer Society; Pages 110-114; 2005.

[35] Yeo, S. and Kim, S.; Scaleable and Flexible Privacy Protection Scheme for RFID Systems; in Proceedings of the Second European Workshop on Security and Privacy in Ad-hoc and Sensor Networks; Springer Lecture Notes in Computer Science; Volume 3813; Pages 153-163; 2005.

[36] Golle, P., Jakobsson, M., Juels, A. and Syverson, P.; Universal Re-Encryption for Mixnets; in Topics of Cryptology; Springer Lecture Notes in Computer Science; Volume 2964; Pages 163-178; 2004.

[37] Saito, J., Ryou, J. and Sakurai, K.; Enhancing Privacy of Universal Re-encryption Scheme for RFID Tags; in Proceedings of the International Conference on Embedded and Ubiquitous Computing; Springer Lecture Notes in Computer Science; Volume 3207; Pages 879-890; 2004.

[38] Avoine, G.; Privacy Issues in RFID Banknote Protection Schemes; in Proceedings of the Sixth International Conference on Smart Card Research and Advanced Applications; Kluwer Academic Publishers; Pages 33-48; 2004.

[39] Zhang, X. and King, B.; Integrity Improvements to an RFID Privacy Protection Protocol for Anti-counterfeiting; in Proceedings of the 8th International Conference on Information Security; Springer Lecture Notes in Computer Science; Volume 3650; Pages 474-481; 2005.

[40] Castelluccia, C. and Avoine, G.; Noisy Tags: A Pretty Good Key Exchange Protocol for RFID Tags; in Proceedings of the International Conference on Smart Card Research and Advanced Applications; Springer Lecture Notes in Computer Science; Volume 3928; Pages 289-299; 2006.

[41] Ranasinghe, D., Engels, D. and Cole, P.; Security and Privacy: Modest Proposals for Low-Cost RFID Systems; in Proceedings of the Auto-ID Labs Research Workshop; Zurich; 2004.

[42] Tuyls, P. and Batina, L.; RFID-Tags for Anti-Counterfeiting; in Topics in Cryptology: The Cryptographers' Track at the RSA Conference 2006; Springer Lecture Notes in Computer Science; Volume 3860; Pages 115-131; 2006.

[43] Feldhofer, M., Dominikus, S. and Wolkersdorfer, J.; Strong Authentication for RFID Systems Using the AES Algorithm; in Proceedings of the 6th International Workshop on Cryptographic Hardware and Embedded Systems; Springer Lecture Notes in Computer Science; Volume 3156; Pages 357-370; 2004.

[44] Lim, C. and Korkishko, T.; mCrypton – A Lightweight Block Cipher For Security of Low-Cost RFID Tags and Sensors; in Proceedings of the 6th International Workshop on Information Security Applications; Springer Lecture Notes in Computer Science; Volume 3786; Pages 243-258; 2005.

[45] Juels, A.; Minimalist Cryptography for Low-Cost RFID Tags; in Proceedings of the 4th International Conference on Security in Communication Networks; Springer Lecture Notes in Computer Science; Volume 3352; Pages 149-164; 2004.

[46] Lee, S., Hwang Y., Lee D. and Lim J.; Efficient Authentication for Low-Cost RFID Systems; in Proceedings of the International Conference on Computational Science and Its Applications; Springer Lecture Notes in Computer Science; Volume 3480; Pages 619-627; 2005.

[47] Dimitriou, T.; A Lightweight RFID Protocol to protect against Traceability and Cloning attacks; in Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks; IEEE Communications Society; Pages 59-66; 2005.

[48] Kang, J. and Nyang, D.; RFID Authentication Protocol with Strong Resistance against Traceability and Denial of Service Attacks; in Proceedings of the Second European Workshop on Security and Privacy in Ad hoc and Sensor Networks; Springer Lecture Notes in Computer Science; Volume 3813; Pages 164-175; 2005.

[49] Duc, D., Park J., Lee H. and Kim, K.; Enhancing Security of EPCglobal Gen-2 RFID Tag against Traceability and Cloning; in Proceedings of the Symposium on Cryptography and Information Security; Hiroshima, Japan; 2006.

[50] Chatmon, T., Le, T. and Burmester, M.; Anonymous Authentication with RFID Devices; Technical Report TR-060112; Computer Science Department, Florida State University; 2006.

[51] Burmester, M., Le, T. and Medeiros, B.; Provably Secure Ubiquitous Systems: Universally Composable RFID Authentication Protocols; Cryptology ePrint Archive; Report 2006/131; Retrieved May, 17, 2006 from http://eprint.iacr.org/2006/131.pdf.

[52] Avoine, G. and Oechslin, P.; RFID Traceability: A Multilayer Problem; in Proceedings of the 9th International Conference on Financial Cryptography and Data Security; Springer Lecture Notes in Computer Science; Volume 3570; Pages 125-140; 2005.

A. Terminology

Cipher

General term denoting cryptographic systems being able to encrypt/decrypt messages.

Ciphertext

An encrypted message. The outcome of an encryption operation.

Cryptanalysis

Studies of methods that may be used to reveal the secret of a cryptographic system or to reveal the meaning of an encrypted message without knowing the actual secret.

Cryptographic hash function

A function that transforms a message m of any length into a fixed size string denoted as the hash value h. There are several requirements that have to be fulfilled by cryptographic hash functions. For a detailed discussion please refer to section 2.5.1 of this work.

Cryptographic protocol

A well-defined way of using two or more cryptographic tools in order to gain a security service (security services are explained in section 2.2 of this work).

Cryptographic tool

See primitive.

Denial of Service attack

An attack on an information technology system with the aim of derogating or blocking the availability of the services this system offers. Often, this attack is based on driving the system to its limits so it cannot service further requests.

Digital signature scheme

Cryptographic schemes that provide data origin authentication and means to detect changes of messages. The term digital signature is most often used in the context of public-key cryptography. A similar concept in secret-key cryptography is called message authentication codes (MAC). Please refer to sections 0 and 2.5.3 of this work to learn more about digital signatures and message authentication codes.

Impersonation

The attempt of an attacker adopting one or more characteristics of a trusted party in order to appear like the trusted party to an identifying third party.

Plaintext

An unencrypted message. The outcome of a decryption operation.

Primitive

Basic cryptographic routines or building blocks of cryptographic protocols. Primitives are discussed in section 2.5 of this work.

Pseudo-random number

A number that was generated with the help of a pseudo-random number generator. Pseudo-random number generators are algorithms that generate sequences of numbers that are not truly random but that show properties which are akin to sequences of numbers generated by random number generators. A detailed discussion about randomness is provided in section 2.5.1 of this work.

Random number

A number that was generated with the help of a random number generator. A random number generator is a physical device able to generate sequences of numbers that show no kind of pattern. A detailed discussion about randomness is provided in section 2.5.1 of this work.

Side channel

A term used in the context of cryptanalysis to denote sources of information in actual cryptographic implementations that may help to reveal any useful knowledge in order to follow the aim of cryptanalysis. Examples for side channels are execution time, current consumption, or electromagnetic emanations.

Spoofing

See impersonation.