The Energy Budget for Wireless Security: Extended Version*

Dave Singelée[1], Stefaan Seys[1], Lejla Batina[1,2], and Ingrid Verbauwhede[1]

[1] K.U.Leuven ESAT-COSIC and iMinds
Kasteelpark Arenberg 10, B-3001 Leuven-Heverlee, Belgium
firstname.lastname@esat.kuleuven.be

[2] Radboud University Nijmegen, CS Dept./Digital Security group
Heyendaalseweg 135, 6525 AJ Nijmegen, The Netherlands
lejla@cs.ru.nl

Abstract. Due to the numerous security and privacy risks, applications deployed in wireless networks require strong cryptographic protection. Reducing the energy cost of cryptographic algorithms and protocols that run on wireless embedded devices, is a crucial requirement when developing security and privacy solutions for wireless networks. The goal of this work is to give an insight to the global energy cost of secure wireless communications. We will compare the energy cost of different wireless standards and a wide range of cryptographic primitives. To illustrate these numbers, we will evaluate the energy consumption of several authentication schemes for RFID. The results show that both computation and communication cost are important factors in the energy budget, and clearly connected to the security and privacy properties of the wireless applications.

Keywords: Wireless Security, RFID, Authentication, Cryptography, Energy Consumption.

1 Introduction

Today, an increasing number of wireless embedded systems – such as ambient intelligence, the Internet of Things, smart dust, e-health applications – are used in our daily life. Various security and privacy risks arise in these environments. To tackle these threats, there is a clear need for cryptographic algorithms and protocols. However, adding these security and privacy solutions to wireless applications also has a cost. Since most wireless embedded devices typically have an extremely limited power, energy and area budget, conventional cryptographic mechanisms are not directly applicable to wireless networks. Therefore, various efficient, lightweight cryptographic solutions have been proposed for wireless networks, to achieve the desired security and privacy properties.

When cryptographic protocols are run on wireless embedded devices, this requires energy from the battery, which is limited. Therefore it is important to explore the global energy

* This article extends the work that has been presented in [36].

cost of secure wireless communications. The two most important factors contributing to the energy cost are the 'computation' cost and the 'communication' cost. There is a clear trade-off between these two components. Some protocols rely on light-weight computations on the resource-constrained device, but require a lot of communication with a reader, server or terminal. Other protocols maybe require heavy public-key operations but only need a small amount of communication. The goal of this article is to get some insight into this computation/communication trade-off, and provide a more holistic overview of the actual cost of carrying out security protocols in a wireless environment. This article extends the work that has been presented in [36].

The rest of this article is structured as follows. Sect. 2 demonstrates the complex ambiguity of the cost of deploying security techniques in a wireless system. Various cost requirements and trade-offs are presented. In Sect. 3, we give an overview of the communication cost of different wireless standards. In Sect. 4, we discuss the computation cost of different cryptographic algorithms. We mainly focus on common cryptographic primitives such as a symmetric block cipher, a symmetric stream cipher, a cryptographic hash function, or an asymmetric cryptographic primitive. To illustrate these cost figures, Sect. 5 describes four RFID authentication protocols which are solely based on these common cryptographic primitives, and evaluates the energy cost of these protocols. We conclude the article in Sect. 6.

2 Costs versus benefits

Cost as well as benefits are difficult to measure concepts. One of the main requirements when designing an embedded system is to minimize the cost. However, there are many aspects to cost. For example, one can try to minimize the area, such as the number of gates or the number of transistors. This is a typical approach for hardware designs. However, in software design, one will measure the memory footprint. Another cost component for (security) implementations is the execution time needed for carrying out cryptographic primitives. One could also measure the design time and design effort, since this also influences the cost of developing the embedded system. Another important requirement is to minimize the energy and/or power consumption of the implementation. Finally, one should also take into account the cost of physical security, i.e. the cost of preventing physical attacks such as side-channel and fault injection attacks.

It is impossible to minimize all these cost requirements simultaneously as there will always be various design trade-offs. For example, one can decrease the clock frequency to minimize the power consumption. However, this will have a negative impact on the execution time of the algorithm. Furthermore, this could also result in an increased energy consumption too, since it takes longer to complete the necessary operations.

The example showed above already demonstrates that there is an intrinsic difference between power and energy. In the literature, there is often confusion between these two different

cost requirements.Wireless sensor nodes are typically battery operated devices. Therefore, the total amount of energy taken from the battery is important. On the other hand, a passive RFID tag consists of a small chip that is powered during the communication with a reader. The power available to the RFID chip is very limited, while the energy supply is virtually unlimited. These two cases are intrinsically different in terms of implementation strategies as well as for the cost issues. Namely, the peak power consumed by a tag is the main concern in the RFID scenario, while it influences the total energy only partially in the wireless sensor node case.

When evaluating the cost of wireless security protocols, one can identify two main components: communication cost and computation cost. The communication cost is associated with the wireless transmission and reception of data. It includes all the radio parts, the digital and analog circuits to process, transmit and receive the information bits. Computation cost is associated with the execution of the cryptographic algorithms on the embedded platform.

The discussion above clearly shows that one first has to thoroughly study the setting and the wireless system in which the security measures will be deployed. Some techniques can be implemented with a small footprint, but are rather slow and energy consuming. Other methods could be optimal in terms of energy, but require a large number of gates to be implemented or consume a large peak power. One has to specify which implementation requirements are important, and which cost parameters are only secondary. Depending on the wireless system, the most optimal choice could be one particular cryptographic algorithm, while this algorithm could be unfavorable for other wireless systems. Note that in this discussion, we did not yet take into account the security and privacy properties of the cryptographic algorithm. Increasing the security and/or privacy protection will most likely also increase the cost. This makes the cost evaluation of various cryptographic operations on wireless embedded devices even more complex. We will illustrate this more in detail in Sect. 5.

It is important to fix the target platform when comparing the energy cost of various techniques and wireless standards. The difference in platform and implementation options has a significant influence on the energy cost. For example, the energy difference between a hardware and a software implementation of the same cryptographic algorithm can vary over many orders of magnitude. This is illustrated in table 1, which shows various AES implementations [34]. If we compare these numbers, one can notice that the performance difference between an ASIC implementation in 45 nm CMOS and an implementation in Java on an embedded micro-processor is more than 8 orders of magnitude. That is why one mainly uses ASIC, FPGA or low level assembly implementations for applications which require low energy.

In this article, our target device is a wireless embedded device, typically a sensor node, that is fed by a battery. Therefore we particularly focus on the energy cost. Our unit of measurement is the amount of energy needed to transmit, receive, and/or compute one bit of

	Throughput (Mbps)	Power (mW)	Performance (Gbps/W)
45 *nm* CMOS [26]	54,272	125	434
FPGA (Virtex 2) [1, 35]	1,320	490	2.7
ASM StrongARM [32]	31	240	0.13
ASM Pentium 3 [25, 35]	648	41.4 E3	15 E-3
C Embedded Sparc	0.133	120	1.1 E-3
Java Embedded Sparc	450 E-6	120	3.7 E-6

Table 1: Performance and power comparison of various AES implementations.

information, typically expressed in J/bit. The energy numbers given in the rest this article are used to give an indication of the cost and to provide insight into the alternatives.

3 Communication cost of different wireless standards

The energy cost of wireless communication is a significant component in the total energy budget. In this section, we have compared different wireless standards which are currently in use and which are aimed mostly at "low power" applications. The different wireless standards are GSM, Wifi, 802.11, Bluetooth, Zigbee, RFID and a state-of-the-art Body Area Network (BAN).

Table 2 shows that the numbers vary substantially for different standards. Wireless LAN (802.11G) has a TX (transmit) and RX (receive) power usage that is orders of magnitude higher than the other standards. Clearly this is because WLAN was designed for high speed throughput and low latency, while the other standards in the table were designed for low power budgets. Nevertheless, when comparing energy per bit, WLAN has the same performance as Bluetooth. Zigbee has a rather high energy per bit requirement in order to achieve its range of about 75m. When comparing Bluetooth classic with the low energy version, we can see that the power consumption has clearly dropped, but with the penalty of a lower throughput. This results in virtually no change in energy per bit. Finally, we can see that the ultra low power design of Vidojkovic, et al. [43], achieves the same performance as the widely used Nordic RF module with a much smaller energy and power budget.

4 Cost of cryptographic operations on embedded platforms

In this section, we will compare the energy cost of various types of cryptographic primitives implemented on an embedded, resource-constrained device. As a block-cipher algorithm we choose AES, the main standard for encryption. It has been already evaluated in several previous studies [17, 28]. Furthermore, we also give the cost of KATAN [9], a symmetric

	Range	Throughput	Freq.	Power		Energy/bit	
				TX	RX	TX	RX
	(m)	(kbps)	(GHz)	(mW)	(mW)	(nJ/bit)	(nJ/bit)
802.11G[18]	30–100	54,000	2.4	2,300	1,900	42.59	35.19
Zigbee[39]	75	250	2.4	46.44	33.30	185.76	133.20
NFC/RFID[31]	0.2	424	13.56 E-3	60.00	60.00	141.51	141.51
Bluetooth classic[38]	30	2,100	2.4	99.90	67.50	47.57	32.14
Bluetooth low energy[40]	5	1,000	2.4	48.00	39.20	48.00	39.20
Nordic RF[30]	5	1,000	2.4	21.47	25.65	10.74	12.83
BAN[43]	5	1,000	2.4	2.60	0.73	2.60	0.73

Table 2: Performance and energy comparison of wireless standards.

block cipher particularly designed for low-end devices such as RFID tags. As a public-key primitive, Elliptic Curve Cryptography (ECC) has proven to be the best choice for constrained environments ever since its invention in the mid 80's. As a consequence, a majority of research on compact low-power public-key hardware architectures for RFID is dedicated to ECC [15, 24]. To complete the study we also choose one cryptographic hash function and one stream cipher. For the hash function we look at the SHA3 competition and compare the cost of some of the finalists [21]. Unfortunately, as a result of the heavy security requirements from NIST, each of these finalists is quite large and power hungry. As stream cipher we select Trivium [8], a light weight version of the eStream portfolio. Finally, we also give the cost of generating an n-bit random number. This is also an important building block of a cryptographic system, since most cryptographic algorithms and protocols rely on the use of random numbers.

4.1 Symmetric block ciphers

There are various designs of symmetric block ciphers, among which some target low-cost embedded devices. In this section, we give the cost and performance numbers for AES, PRESENT and KATAN implementations on an ASIC platform.

There exist several very compact hardware implementation of AES that are suitable for low-cost applications [10, 12, 29]. We give detailed numbers for the most compact one of Moradi et al. [29] of AES-128, which occupies only 2400 GEs and requires 226 cycles for encryption.

In the recent past several lightweight block ciphers were proposed in the literature that aim (almost exclusively) at low-cost devices. As an example, we give the numbers for KATAN [9] and PRESENT [4]. KATAN is a block cipher that belongs to a family of small and efficient hardware-oriented block ciphers. KATAN ciphers include KATAN32, KATAN48, and KATAN64. All three ciphers use 80-bit keys and have a different block size (KATANn has an n-bit block size). All three block ciphers are highly compact, but we only show the cipher

with the smallest block size as this results in the smallest circuit area (of only 802 GEs). The relevant numbers are given in Table 3. Although the libraries used for the two design are not the same, 0.18 μm and 0.13 μm technology respectively, which effects power consumption, it is evident that both ciphers are suitable for the envisioned low-cost applications. The somewhat higher energy per bit for PRESENT is mainly due to the number of cycles (more than double compared to AES and KATAN) and the high power consumption.

	Tech. (μm)	Throughput (kbps)	Freq. (MHz)	Area (GEs)	Latency (# cycles)	Power (μ W)	Energy per bit (pJ/bit)
AES-128 [29]	0.18	56.64	0.1	2400	226	3.7	65.33
KATAN32 [20]	0.13	12.5	0.1	802	256	0.38	30.48
PRESENT [33]	0.18	11.4	0.1	1075	563	2.52	221.73

Table 3: Performance and energy numbers of AES, KATAN and PRESENT on ASIC.

4.2 Symmetric stream cipher: Trivium

Stream ciphers are very important symmetric-key primitives that are often used to secure wireless communications. The eSTREAM project (as a part of ECRYPT NoE project 2004-2008) had as main goal to identify a portfolio of new stream ciphers, that were divided into two groups: software and hardware (efficient).

One of the remaining three hardware candidates of the project is a stream cipher Trivium [8]. Table 4 gives the main characteristics of Trivium when deployed for RFID applications, i.e. on an ASIC platform using 0.13 μm standard cell libraries. The numbers were given in the work of Good and Benaissa [14]. Among several implementation variants, we show the most compact one and the one with the lowest energy per bit.

	Throughput (Mbps)	Freq. (MHz)	Area (GEs)	Latency (# cycles)	Power (μ W)	Energy per bit (pJ/bit)
Trivium [14]	0.10	0.1	2599	13,140	5.54	55.36
Triviumx64 [14]	6.40	0.1	4921	240	14.31	2.23

Table 4: Performance and energy numbers of Trivium on ASIC (0.13 μm synthesis).

4.3 Elliptic Curve Cryptography

In the past, there has been a substantial amount of papers investigating public-key cryptography for very constrained environments. Most of the papers considered Elliptic Curve Cryptography (ECC) as the most suitable public-key primitive, some examples are [15, 24].

In Table 5, we give the relevant numbers for two ECC implementations on an ASIC. The first implementation is based on an architecture presented in [24]. The proposed processor for ECC over GF(2^{163}) is able to perform EC scalar multiplications as well as general modular arithmetic (additions and multiplications) which are needed for cryptographic protocols. The circuit is fabricated in a 0.13 μm CMOS technology. Another state-of-the-art solution is the silicon chip ECCon, an Elliptic Curve Cryptography processor for application in Radio-Frequency Identification. The chip is produced in a 0.18 μm CMOS technology and the architecture is detailed in [15]. Both chips are using the same binary field for the underlying arithmetic and the size (for both) is around 15 kgates. We notice that although the numbers for power are similar, there is a big difference in the energy numbers. This is due to the performance issues, i.e. the difference in the number of cycles.

	Technology (μm)	Latency (# cycles)	Frequency (KHz)	Power (μ W)	Energy per bit (nJ/bit)
ECC [24]	0.13	86,244	100	7.3	38.59
ECC [15]	0.18	296,000	106	8.57	146.82

Table 5: Performance and energy numbers of ECC on ASIC.

As a platform comparison, the work presented in [22] presents a lightweight implementation of the elliptic curve Diffie-Hellman (ECDH) key exchange for ZigBee-compliant sensor nodes equipped with an ATmega128 processor running the TinyOS operating system. This is a popular platform for embedded security applications. For one point multiplication on this platform, the minimal energy consumption is estimated as 17.04 mJ (when the point is fixed). Although, the authors have investigated an implementation over a 192-bits long prime field, the difference is still more than 2 orders of magnitude.

4.4 Cryptographic hash function

Cryptographic hash functions are widely used in cryptographic algorithms (e.g., digital signatures) and authentication protocols. There is a long list of cryptographic hash functions, although many have been found to be vulnerable and are no longer used in practice. As an example for hash functions, we consider SHA256 and the last five candidates of the NIST SHA3

competition for designing a new hash function. They were all evaluated on a fixed hardware platform, i.e. a SASEBO board. All the results were obtained under the same conditions and synthesized in 90 nm CMOS technology. The results are given in Table 6 and more details can be found in [21].

	Throughput (@ 250 MHz) (Mbps)	Energy per bit (pJ/bit)
SHA256	2000	2
Blake	6000	2.5
Grøstl	13000	2.5
JH	4600	2
Keccak	15000	1
Skein	6700	6

Table 6: SHA3 ASIC (90 nm) synthesis [21].

4.5 Random number generator

Many cryptographic primitives and protocols require the generation of true random numbers. Various ASIC implementations of cryptographic random number generators have been described in the literature. One of the first low-energy designs was proposed by Bucci et al. [7], who integrated a high-speed random number generator on a smartcard IC. Brederlow et al. [5] presented a random number generator design that utilizes the noise produced by single-oxide traps in small area MOSFETs, in combination with built-in redundancy. To improve the robustness to undesired deterministic noise, an important feature in the design of random number generators, Tokunaga et al. [41] introduced a metastability-based random number generator. Matsumoto et al. [27] proposed a very compact random number generator based on the use of SiN MOSFETs. The current state-of-the-art random number generator design was proposed by Srinivasan et al. [37]. They have implemented a high-throughput random number generator in 45 nm CMOS technology. The numbers of these different ASIC designs are shown in Table 7.

When observing these numbers, one can observe that the random number generator implemented by Srinivasan et al. [37] is significantly better than the other designs shown in Table 7. However, the libraries used for the various designs are not the same (45 nm compared to 120-250 μm for the other ones). This largely affects the energy consumption, which usually scales with the square of the technology node. Note that although all these random number generators are suitable for low-cost applications, they were not always particularly designed to have a low energy consumption. For example, some random number generators

were mainly designed to have a smaller circuit area (such as [27]), or to have a better robustness against environmental noise and supply voltage variations (such as [41]). This explains the variance in the energy numbers of the different implementations.

Designing a random number generator on a microcontroller is a challenging and rather unexplored research domain. Nevertheless, some attempts have already been made. For example, Hlaváč, Lórencz and Hadáček have presented a design to generate true random numbers on an Atmel AVR microcontroller, using external oscillators [16]. However, this solution mainly aims to improve the bit rate, which was relatively low in previously proposed designs, and does not really focus on low energy or power consumption.

	Technology (nm)	Area μm^2	Throughput (Mbps)	Power (mW)	Energy per bit (pJ/bit)
Bucci et al. [7]	180	15,824	10	2.3	230
Brederlow et al. [5]	120	9,000	0.2	50 E-3	250
Tokunaga et al. [41]	130	36,000	0.2	1	5,000
Matsumoto et al. [27]	250	1,200	2	1.9	950
Srinivasan et al. [37]	45	1,024	4000	2.26	0.57

Table 7: Performance and energy numbers of random number generators on ASIC.

5 Energy cost of RFID authentication protocols

Most RFID systems require the deployment of an authentication protocol, in which the RFID tag authenticates itself to a reader and/or server. Depending on the cryptographic building blocks being used in the authentication protocol, various security and privacy requirements can be met. Some protocols proposed in the literature are designed to merely offer tag-to-server authentication, while others also offer mutual authentication, or provide various means of privacy protection. The protocols also differ in terms of scalability and key management complexity. Of course, these security and/or privacy properties have an influence on the energy cost of these protocols.

To illustrate the energy figures given in the previous sections, we will now evaluate the energy cost of some RFID authentication protocols. We made a selection of four RFID authentication protocols that rely solely on one cryptographic building block: a cryptographic hash function, a symmetric block cipher, a symmetric stream cipher, or an asymmetric cryptographic primitive. The energy cost of these primitives has been discussed in the previous section. We will now describe each of the four RFID authentication protocols more in detail, give an overview of their main properties and compute their energy cost. Next, we will

compare the communication-computation trade-off in the energy cost of the protocols. Note that we discuss these authentication protocols merely for illustrative reasons. We do not want to make any recommendations on deploying (or not using) a particular RFID authentication protocol.

Before we can compute the energy cost, we first need to make some general assumptions. Since the protocols are deployed in an RFID system, we assume that RFID is used as a communication standard. To generate random numbers, which is required in each of the protocols, we use the design of Bucci et al. [7]. Furthermore, we assume that the protocols are implemented on an active RFID tag, which is operated by a battery.

5.1 Basic zero-knowledge device authentication protocol of Engbert et al.

Protocol description Engberg, Harning and Jensen were one of the first to propose an RFID authentication protocol [11]. They presented a modular zero-knowledge protocol which relies exclusively on the use of a cryptographic hash function. The authors also discuss how their protocol can be extended to offer advanced security and privacy protection. In this paper, we limit ourself to the basic authentication protocol. During a protocol run, the reader sends an authenticated request to a particular tag, upon which the latter authenticates itself to the reader. Therefore, the protocol is designed to search for a specific RFID tag. Such type of authentication protocols are denoted by *RFID search protocols* in the literature.

The protocol works as follows. The reader and RFID tag share a unique symmetric key K. To search for a particular RFID tag, the reader first generates two random numbers N_1 and N_2. These numbers are then combined with the key K in an authentication request, which is sent to the tag. When receiving this request, the tag first combines the nonce N_1 and the key K to recover the nonce N_2. Next, it uses this nonce and the key K to check the last part of the message. If these checks are successful, the request is authorized and the tag computes a hash function on the *XOR* of the nonces N_1 and N_2 and the key K. This response is sent to the reader, which then verifies its correctness.

Properties The protocol described in Fig. 1 is a search protocol, where RFID tags only reply to authorized requests. If the last part of the authentication request is incorrect, the tag does not respond. Note that RFID search protocols are not designed to identify all tags within the reader's communication range efficiently. The reader already has to know which tag it is looking for, since it has to include the shared key K in its authentication request. Identifying a random tag using a "conventional" symmetric challenge-response protocol would require on average $n/2$ protocol runs, where n denotes the number of tags in the RFID system. This approach would become impractical when the number of tags is large.

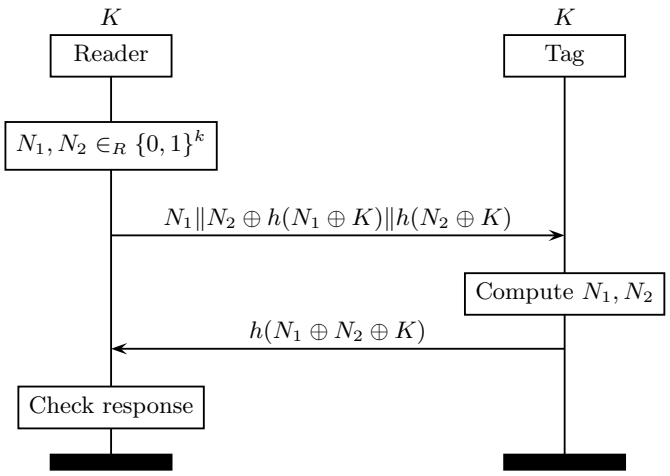

Fig. 1: RFID search protocol based on cryptographic hash function [11]

The basic protocol provides unilateral authentication, since the tag authenticates itself to the reader. To prevent replay attacks, the tag could store the current value of the nonce N_1. Any authentication request that contains a nonce N_1 that is lower or equal to the value stored in the tag's memory, is ignored. Otherwise, the tag updates the value N_1 in its memory. The authors suggest to use the current time (e.g., in seconds) as nonce N_1. This mechanism ensures that the request is fresh and originates from the reader. It also prevents tracking attack, since an attacker cannot replay a request to track a particular tag. The basic protocol does not offer any "advanced" privacy protection. However, one could extend the protocol by requiring the key K to be updated by applying a cryptographic hash function, after each successful run of the protocol. This would ensure forward privacy[3]. However, this approach requires careful attention to key synchronization.

Energy cost The search protocol described in Fig. 1 has a security parameter k. Both the key size and the bitlength of the random nonces are equal to this parameter. In this paper, we assume that k equals 128. Furthermore, we also have to choose which hash function will be used in the protocol. Let us assume that the SHA256 algorithm or the SHA-3 candidate

[3] We refer to [42] for a formal definition of forward privacy.

JH is used. Since the output of these hash functions is too large, one needs to truncate it to 128 bits. By making these assumptions, we can compute the energy cost of the protocol, which is equal to $72,454$ nJ. Of this cost, the computation cost is equal to 768 pJ, and the communication cost to $72,453$ nJ.

5.2 ISO 9798-2 mutual entity authentication protocol based on AES

Protocol description Most RFID authentication protocols are challenge-response protocols which use symmetric and/or asymmetric cryptographic primitives. Protocols for symmetric challenge-response techniques based on encryption are defined in the ISO/IEC 9798-2 standard [19]. Feldhofer et al. [13] proposed to employ these symmetric challenge-response protocols in the context of RFID networks, using the symmetric block cipher AES. The standard both defines a unilateral and a mutual authentication protocol. The latter is shown in Fig. 2 and works as follows: The reader sends a random number r_B to the tag. The tag encrypts r_B and a self-generated random number r_A with the shared key K and sends it to the reader. The latter decrypts the message, checks if r_B is correct, and gets r_A. Next, the reader changes the sequence of the random numbers, encrypts it with K, and sends it to the tag. The tag checks the result and verifies the identity of the reader. The unilateral authentication protocol, in which the tag authenticates itself to the reader, works similar. Instead of three rounds, it contains only two steps. The reader sends a random nonce, and the tag responds with the encryption of this challenge.

Properties The protocol shown in Fig. 2 is designed to provide mutual entity authentication. It offers privacy protection against an adversary which does not know the secret key K (this is denoted by *weak privacy* in the privacy framework of Vaudenay [42]). The adversary needs to decrypt the second protocol message to link various protocol runs to the same tag. The unilateral entity authentication protocol does not offer any privacy protection. Replaying a challenge results in a tag replying with the same nonce, which can be used to track the tag. For privacy reasons, one also has to take care of how to use the symmetric block cipher (e.g., AES) in the protocol. If the first output block of the cipher only depends on the key K and the challenge r_A, then reusing the same challenge would result in the first output block being equal. This can then be used by an attacker to trace a tag. One should however note that the authentication protocols defined in the ISO/IEC 9798-2 standard are not specifically designed to be used in RFID networks. Therefore, providing privacy protection was not one of the design requirements. The mutual entity authentication protocol is not scalable. On average, the reader has to carry out $n/2$ decryptions in the second protocol step, where n denotes the number of tags in the RFID system.

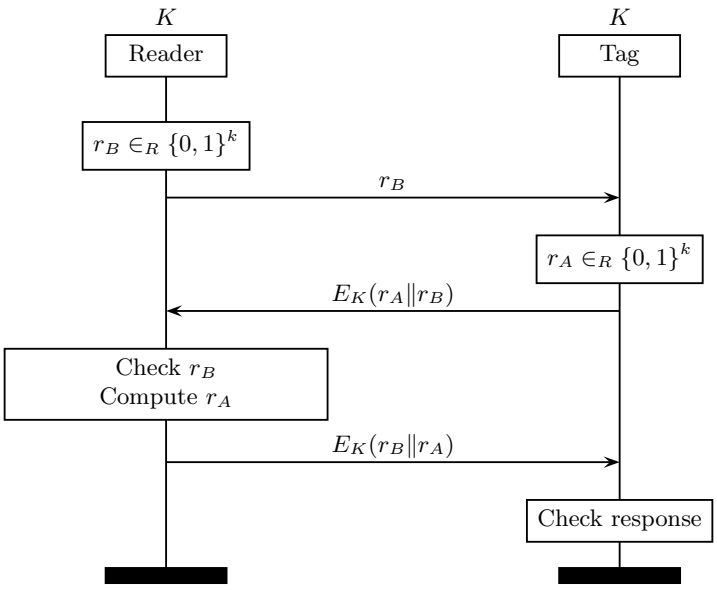

Fig. 2: Mutual entity authentication using a symmetric block cipher [19]

Energy cost The mutual entity authentication protocol described above has a security parameter k. The size of the random nonces r_A and r_B is equal to k bits, while the key size is equal to $2k$ bits. In this paper, we assume that k equals 64. We can then compute the energy cost of the protocol, which is equal to $45,314$ nJ. This cost encompasses a computation cost of 31.444 nJ and a communication cost of $45,283$ nJ.

5.3 PEPS protocol of Billet et al.

Protocol description Billet, Etrog and Gilbert recently proposed the PEPS protocol [3]. This privacy-preserving RFID authentication protocol relies exclusively on the use of a stream cipher. Since the protocol makes use of a symmetric primitive, the reader and RFID tag need to share a secret key K. During the authentication protocol, a secure stream cipher G is used. This stream cipher takes as input an initial value IV and the key K. The bitlength of IV is n, and the size of the key K is k bits. The stream cipher G is used to produce a key stream sequence $G(K, IV)$ of length $m = 2l + k$, where l represents the length of

the authentication responses of the protocol. The key stream $G(K, IV)$ is viewed as the concatenation $G_t(K, IV)\|G_r(K, IV)\|G_s(K, IV)$ of three subsequences of respective lengths l, l, and k. Thus G_t and G_r produce l-bit sequences while G_s produces a k-bit sequence.

The protocol, which is shown in Fig. 3 works as follows [3]. First the reader randomly generates an authentication challenge a of length $n/2$ bits and sends it to the tag. At the receipt of a, the tag (whose current key value is denoted by K) randomly generates a $n/2$-bit number b, derives the initial value $IV = a\|b$, and computes $G(K, IV)$ using the stream cipher G. Then it sends $G_t(K, IV)$ to the reader. The reader authenticates the tag by searching a tag index i and a key $K' \in \{K_i, K_{i,new}\}$ such that $G_t(K', IV) = G_t(K, IV)$. If the reader finds such an index i then the tag is considered as successfully authenticated as tag T_i, otherwise the authentication protocol has failed. If the tag has been authenticated as tag T_i, the reader updates the current key pair associated with tag T_i to $(K', G_s(K', IV))$. Next the reader computes the response $G_r(K', IV)$ and sends it back to the tag. At the receipt of the reader's answer, the tag checks whether $G_r(K', IV) = G_r(K, IV)$. If this equality holds, it replaces its current key value K by $G_s(K, IV)$. If the reader is not successfully authenticated, it keeps its current key value.

Properties The PEPS protocol provides mutual entity authentication under the assumption that the underlying stream cipher is secure. In each protocol run, the tag updates its secret key K when the reader is successfully authenticated. As a result, the protocol offers forward privacy under the assumption that the maximum number of authentications an adversary can disturb is not too large. This is denoted by *almost forward private*. More details can be found in [3]. As many other RFID protocols based on symmetric cryptographic primitives, it is not scalable. On average, the reader has to compute $n/2$ outputs of the stream cipher, where n denotes the number of tags in the RFID system.

Energy cost As was already described above, the PEPS protocol has various security parameters: n, k and l. By carefully selecting the appropriate values for these parameters, one can tune the security and privacy properties of the protocol. For simplicity reasons, we assume that $n = k = l = 128$. We can then compute the energy cost of the protocol, which is equal to $72,489$ nJ. This cost encompasses a computation cost of 35.978 nJ and a communication cost of $72,453$ nJ.

5.4 ECC-based Randomized Schnorr protocol

Protocol description Recently, various RFID authentication protocols (such as [23]) have been proposed that rely exclusively on the use of Elliptic Curve Cryptography (ECC), since these offer some interesting security and privacy properties. One of these protocols is the

Randomized Schnorr protocol [6], proposed by Bringer, Chabanne and Icart. Before we will describe the protocol, let us first introduce some notation. The reader and the tag each have a private key, which are denoted by y and x respectively. The key size is typically 163 bits. We denote P as the base point on a Elliptic Curve, and the public key of the reader as $Y = yP$. In this equation, yP denotes the point derived by the point multiplication operation in the Elliptic Curve group. Each tag has a unique public key xP. One should note that, although the name suggests that it can be publicly known, a tag should not reveal its public key during the execution of the protocol, as this would cause tracking attacks.

The protocol is shown in Fig. 4 and works as follows. The protocols starts by the tag generating two random numbers r_1 and r_2. Next, it computes two points on the elliptic curve: $T_1 = r_1P$ and $T_2 = r_2Y$, and sends them to the reader. When receiving these points, the reader responds with a random challenge c. Next, the tag computes the response v using the challenge c, the random numbers r_1 and r_2, and its private keys x. This response is sent back to the reader, to prove the tag's identity. The reader can check the response,using its private key y, by performing the following computation:

$$c^{-1}[vP - T_1 - y^{-1}T_2] =?xP$$

If the correct private key x is used, the output of the computation is equal to the tag's public key X.

Properties The Randomized Schnorr protocol provides unilateral authentication, as the tag authenticates itself to the reader. The security properties of the protocol are based on the security of the Schnorr scheme against active impersonation attacks under the OMDL assumption. The latter was proved in [2]. Bringer, Chabanne and Icart show that a relevant active adversary against the Randomized Schorr protocol can be transformed into a relevant active adversary against the Schnorr scheme [6]. The protocol also offers narrow-strong privacy in the theoretical privacy framework of Vaudenay [42], under the assumption that the Decisional Diffie-Hellman (DDH) is hard to solve. It does not offer any protection against a wide attacker [23], which has access to the result of the authentication protocol in the reader (accept or reject the tag's authentication claim). The Randomized Schnorr protocol is scalable, as the reader computes the public key of the tag to verify an authentication claim.

Energy cost The Randomized Schnorr protocol has a security parameter k. We assume that this parameter is equal to 163. Furthermore, we propose to slightly modify the protocol to improve the communication cost. In the Randomized Schnorr protocol, the tag needs to send two points on the elliptic curve. Instead of sending both the x and y coordinate of a point, we assume that the tag only sends the 163-bit x coordinate and 1 bit of the y coordinate. As

a result, the tag only needs to send 164 instead of 326 bits. By making these assumptions, we can compute the energy cost of the protocol, which is equal to 105, 203 nJ. This cost encompasses a computation cost of 12, 655 nJ and a communication cost of 92, 548 nJ.

5.5 Performance comparison

The energy cost of the four RFID authentication protocols is summarized in table 8. An important observation is that for all these protocols, the communication cost is significantly larger than the computation cost. Therefore, to optimize the energy cost, one must reduce the communication overhead.

	Cryptographic primitive	Key size (bits)	Nonce size (bits)	Energy cost (nJ)	Communication cost		Computation cost	
					Absolute (nJ))	Relative %	Absolute (nJ)	Relative %
Engbert et al.[11]	SHA-256	128	128	72,454	72,453	99.999	0.77	1.03 E-3
ISO 9798-2[19]	AES	128	64	45,314	45,283	99.93	31.44	0.07
PEPS[3]	Trivium	128	64	72,489	72,453	99.95	35.98	0.05
Randomized Schnorr[6]	ECC	163	163	105,203	92,548	87.97	12,655	12.03

Table 8: Energy comparison of the authentication protocols using RFID radio.

If a communication technology with a lower energy cost would have been used, the trade-off between computation and communication cost would be completely different. To illustrate this more in detail, let us assume that instead of RFID, the devices would communicate via a very low-cost radio that is also used in Body Area Networks (BANs) [43]. In that scenario, the computation cost becomes a more important factor, as is shown in table 9. However, except for the authentication protocol based on ECC, the communication cost remains the largest component in the total energy cost.

	Cryptographic primitive	Key size (bits)	Nonce size (bits)	Energy cost (nJ)	Communication cost		Computation cost	
					Absolute (nJ))	Relative %	Absolute (nJ)	Relative %
Engbert et al.[11]	SHA-256	128	128	613.89	613.12	99.87	0.77	0.13
ISO 9798-2[19]	AES	128	64	504.40	472.96	93.77	31.44	6.23
PEPS[3]	Trivium	128	64	675.34	639.36	94.67	35.98	5.33
Randomized Schnorr[6]	ECC	163	163	14,051	1,396	9.93	12,655	90.07

Table 9: Energy comparison of the authentication protocols using BAN radio.

6 Conclusions

Body Area networks, sensor networks, RFID, NFC and many more wireless technologies will offer novel experiences to the human in the Future Internet of Things. All of these applications carry security and privacy risks. Many novel cryptographic algorithms and protocols with varying security and privacy properties have been proposed to solve these issues.

In this article, we aim at connecting these security and privacy properties with the implementation constraints, more specifically with the limited power and energy budgets of many applications. By taking both the communication and computation cost into account, we aim at providing a more holistic insight into the actual cost of security protocols. The trade-off between the security and privacy properties, the energy cost of cryptographic computations and wireless communication, has been illustrated by evaluating the total energy cost of various RFID authentication schemes. These numbers have demonstrated that for all these protocols, the communication cost is significantly larger than the computation cost. This clearly shows that one primarily needs to reduce the former to decrease the total energy cost in the wireless system.

Acknowledgments

This work was supported in part by the Research Council KU Leuven: GOA TENSE (GOA/11/007), by the Flemish Government, FWO G.0550.12N, G.00130.13N, FWO G.0876.14N, by the Hercules Foundation AKUL/11/19, and by the European Commission through the ICT programme through Horizon 2020 research and innovation programme under grant agreement 644052 HECTOR. The authors would also like to thank Vladimir Rožić for his useful input on random number generators.

References

1. Amphion. www.amphion.com
2. M. Bellare, and A. Palacio. GQ and Schnorr Identification Schemes: Proofs of Security Against Impersonation under Active and Concurrent Attacks. In M. Yung (Ed.), *Advances in Cryptology (CRYPTO'02), Lecture Notes in Computer Science*, volume 2442, pages 162–177. Springer-Verlag, 2002.
3. O. Billet, J. Etrog, and H. Gilbert. Lightweight Privacy Preserving Authentication for RFID Using a Stream Cipher. In S. Hong, and T. Iwata (Eds.), *17th International Workshop on Fast Software Encryption (FSE '10), Lecture Notes in Computer Science*, volume 6147, pages 55–74. Springer-Verlag, 2010.
4. A. Bogdanov, L. Knudsen, G. Leander, C. Paar, A. Poschmann, M. Robshaw, Y. Seurin, and C. Vikkelsoe. PRESENT: An Ultra-Lightweight Block Cipher. In P. Paillier, and I. Verbauwhede (Eds.), *Proceedings of the 9th International Workshop on Cryptographic Hardware and Embedded Systems (CHES '07), Lecture Notes in Computer Science*, volume 4727, pages 450–466. Springer-Verlag, 2007.

5. R. Brederlow, R. Prakash, C. Paulus, and R. Thewes. A Low-Power True Random Number Generator using Random Telegraph Noise of Single Oxide-Traps. In *Proceedings of the 53rd Solid-State Circuits Conference (ISSCC '06)*, pages 1666–1675, IEEE International, 2006.
6. J. Bringer, H. Chabanne, and T. Icart. Cryptanalysis of EC-RAC, a RFID Identification Protocol. In M. Franklin, L. Hui, and D. Wong (Eds.), *International Conference on Cryptology and Network Security (CANS'08), Lecture Notes in Computer Science*, volume 5339, pages 149–161. Springer-Verlag, 2008.
7. M. Bucci, L. Germani, R. Luzzi, A. Trifiletti, and M. Varanonuovo. A High-Speed Oscillator-Based Truly Random Number Source for Cryptographic Applications on a Smart Card IC. In *IEEE Transactions on Computer*, volume 52(4), pages 403–409, April 2003.
8. C. De Cannière. Trivium: A Stream Cipher Construction Inspired by Block Cipher Design Principles. In Sokratis K. Katsikas, Javier Lopez, Michael Backes, Stefanos Gritzalis, and Bart Preneel (Eds.), *Proceedings of the 9th International Conference on Information Security (ISC '06), Lecture Notes in Computer Science*, volume 4176, pages 171–186. Springer-Verlag, 2006.
9. C. De Cannière, O. Dunkelman, and M. Knezevic. KATAN and KTANTAN - A Family of Small and Efficient Hardware-Oriented Block Ciphers. In C. Clavier, and K. Gaj (Eds.), *Proceedings of the 11th International Workshop on Cryptographic Hardware and Embedded Systems (CHES '09), Lecture Notes in Computer Science*, volume 5747, pages 272–288. Springer-Verlag, 2009.
10. D. Canright. A Very Compact S-Box for AES. In J.R. Rao and B. Sunar (Eds.), *Proceedings of the 7th International Workshop on Cryptographic Hardware and Embedded Systems (CHES '05), Lecture Notes in Computer Science*, volume 3659, pages 441–455. Springer-Verlag, 2005.
11. S.J. Engberg, M.B. Harning, and C.D. Jensen. Zero-knowledge Device Authentication: Privacy & Security Enhanced RFID preserving Business Value and Consumer Convenience. In *Second Annual Conference on Privacy, Security and Trust (PST '04)*, pages 89–101. 2004.
12. M. Feldhofer, J. Wolkerstorfer, and V. Rijmen. AES Implementation on a Grain of Sand. In *IEE Proceedings of Information Security*, volume 152(1), pages 13–20. 2005.
13. M. Feldhofer, S. Dominikus, and J. Wolkerstorfer. Strong Authentication for RFID Systems using the AES Algorithm. In M. Joye, and J. J. Quisquater (Eds.), *Proceedings of the 6th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'04), Lecture Notes in Computer Science*, volume 3156, pages 357–370. Springer-Verlag, 2004.
14. T. Good and M. Benaissa Hardware performance of eStream phase-III stream cipher candidates. In *Proceedings of SASC 2008 - The State of the Art of Stream Ciphers*, Lausanne, Switzerland, February 13-14, 2008, pages 163–174, 2008.
15. D. Hein, J. Wolkerstorfer, and N. Felber. ECC is Ready for RFID - A Proof in Silicon. In R. Avanzi, L. Keliher, and F. Sica (Eds.), *Selected Areas in Cryptography, Lecture Notes in Computer Science*, volume 5381, pages 401–413. Springer-Verlag, 2009.
16. J. Hlaváč, R. Lórencz, and M. Hadáček. True Random Number Generation on an Atmel AVR Microcontroller. In *Proceedings of the 2nd International Conference on Computer Engineering and Technology (ICCET '10)*, pages 493–495, IEEE International, 2010.
17. A. Hodjat, and I. Verbauwhede. The Energy Cost of Embedded Security for Wireless Sensor Networks. In G. Griffin, T. La Porta, and S. Phoha (Eds.), *Sensor Network Operations*, John Wiley & Sons, pages 510–522, 2006.
18. Intel. Intel Wireless WiFi Link 5300 Module quick specs datasheet. 2008. Online: `http://h18000.www1.hp.com/products/quickspecs/13085_na/13085_na.PDF`
19. ISO/IEC 9798-2. Information Technology – Security Techniques – Entity Authentication – Part 2: Mechanisms Using Symmetric Encipherment Algorithms, 1999.
20. M. Knezevic. Efficient Hardware Implementations of Cryptographic primitives. Ph.D. thesis, Katholieke Universiteit Leuven, Belgium, 208 pages, 2011.

21. M. Knezevic, K. Kobayashi, J. Ikegami, S. Matsuo, A. Satoh, U. Kocabas, J. Fan, T. Katashita, T. Sugawara, K. Sakiyama, I. Verbauwhede, K. Ohta, N. Homma, and T. Aoki. Fair and Consistent Hardware Evaluation of Fourteen Round Two SHA-3 Candidates. To appear in *IEEE Transactions on VLSI*, 13 pages, 2011.
22. C. Lederer, R. Mader, M. Koschuch, J. Großschädl, A. Szekely, S. Tillich. Energy-Efficient Implementation of ECDH Key Exchange for Wireless Sensor Networks. In *Information Security Theory and Practices — WISTP 2009*, pages 112–127. September 2009.
23. Y. K. Lee, L. Batina, D. Singelée, and I. Verbauwhede. Low-Cost Untraceable Authentication Protocols for RFID (extended version). In S. Wetzel, C. N. Rotaru, and F. Stajano (Eds.), *Proceedings of the 3rd ACM Conference on Wireless Network Security (WiSec '10)*, pages 55–64. ACM, 2010.
24. Y. K. Lee, K. Sakiyama, L. Batina, and I. Verbauwhede. Elliptic Curve Based Security Processor for RFID. In *IEEE Transactions on Computer*, volume 57(11), pages 1514–1527, November 2008.
25. H. Lipmaa. AES Candidates: A survey of implementations. http://www.tcs.hut.fi/ helger/aes/rijndael.html.
26. S.K. Mathew, F. Sheikh, M. Kounavis, S. Gueron, A. Agarwal, S.K. Hsu, H. Kaul, M.A. Anders, and R. K. Krishnamurthy. 53 Gbps Native $GF(2^4)^2$ Composite-Field AES-Encrypt/Decrypt Accelerator for Content-Protection in 45nm High-Performance Micro-Processors. In *IEEE Journal of Solid-State Circuits*, volume 46(4), pages 767–776, April 2011.
27. M. Matsumoto, S. Yasuda, R. Ohba, K. Ikegami, T. Tanamoto, and S. Fujita. $1200\mu m^2$ Physical Random-Number Generators Based on SiN MOSFET for Secure Smart-Card Application. In *Proceedings of the 55th Solid-State Circuits Conference (ISSCC '08)*, pages 414–624, IEEE International, 2008.
28. G. de Meulenaer, F. Gosset, F.-X. Standaert, and O. Pereira. On the Energy Cost of Communications and Cryptography in Wireless Sensor Networks, (extended version). In *IEEE Int. Workshop on Security and Privacy in Wireless and Mobile Computing, Networking and Communications (SecPriWiMob '08)*, pages 580–585. IEEE, October 2008.
29. A. Moradi, A. Poschmann, S. Ling, C. Paar, H. Wang. Pushing the Limits: A Very Compact and a Threshold Implementation of AES. In K. Patterson (Ed.), *Advances in Cryptology (EUROCRYPT 2011), Lecture Notes in Computer Science*, volume 6632, pages 69–88, Springer-Verlag, 2011.
30. Nordic Semiconductors. nRF24L01 Single Chip 2.4GHz Transceiver Product Specification. 2007. Online: `http://www.nordicsemi.com/eng/content/download/2730/34105/file/nRF24L01_Product_Specification_v2_0.pdf`
31. NXP Semiconductors. NXP NFC controller PN544 for mobile phones and portable equipment. 2010. Online: `http://www.nxp.com/acrobat_download2/literature/9397/75016890.pdf`
32. D.A. Osvik, J.W. Bos, D. Stefan, and D. Canright. Fast software AES encryption. In S. Hong, and T. Iwata (Eds.),*Proceedings of the 17th international conference on Fast software encryption (FSE'10), Lecture Notes in Computer Science*, volume 6147, pages 75–93. Springer-Verlag, 2010.
33. C. Rolfes, A. Poschmann, G. Leander, and C. Paar. Ultra-Lightweight Implementations for Smart Devices - Security for 1000 Gate Equivalents. In *proceedings of the 8th IFIP WG 8.8/11.2 International Conference on Smart Card Research and Advanced Applications (CARDIS '08), Lecture Notes in Computer Science*, volume 5189, pages 89–103, Springer-Verlag, 2008.
34. P. Schaumont, and I. Verbauwhede. Domain specific codesign for embedded security. In *Computer*, volume 36(4), pages 68–74, 2003.
35. P. Schaumont, and I. Verbauwhede. Domain Specific Tools and Methods for Application in Security Processor Design. In *Kluwer Journal for Design Automation of Embedded Systems*, volume 7(4), pages 365–383, 2002.
36. D. Singelée, S. Seys, L. Batina, and I. Verbauwhede. The Communication and Computation Cost of Wireless Security – Extended Abstract. In G. Tsudik, and N. Asokan (Eds.), *Proceedings of the 4th ACM Conference on Wireless Network Security (WiSec '11)*, pages 1–3. ACM, 2011.

37. S. Srinivasan, S. Mathew, V. Erraguntla, and R. Krishnamurthy. A 4Gbps 0.57pJ/bit Process-Voltage-Temperature Variation Tolerant All-Digital True Random Number Generator in 45nm CMOS. In *Proceedings of the 22nd International Conference on VLSI Design (VLSI Design '09)*, pages 301–306, IEEE International, 2009.
38. Texas Instruments. Texas Instruments Bluetooth BRF6100 and BRF6150 Product bulletin. 2004. Online: `http://focus.ti.com/pdfs/wtbu/TI_brf6100_6150.pdf`
39. Texas Instruments. Texas Instruments CC2520 2.4 GHz IEEE 802.15.4/Zigbee RF Transceiver datasheet. 2007. Online: `http://www.ti.com/lit/ds/symlink/cc2520.pdf`
40. Texas Instruments. Texas Instruments CC2540 2.4-GHz Bluetooth low energy System-on-Chip datasheet. 2011. Online: `http://www.ti.com/lit/ds/symlink/cc2540.pdf`
41. C. Tokunaga, D. Blaauw, and T. Mudge. True Random Number Generator with a Metastability-Based Quality Control. In *Proceedings of the 54th Solid-State Circuits Conference (ISSCC '07)*, pages 404–611, IEEE International, 2007.
42. S. Vaudenay. On privacy models for RFID. In K. Kurosawa (Ed.), *Advances in Cryptology (ASIACRYPT'07), Lecture Notes in Computer Science*, volume 4833, pages 68–87. Springer-Verlag, 2007.
43. M. Vidojkovic, H. Xiongchuan, P. Harpe, S. Rampu, Zhou Cui, Li Huang, K. Imamura, B. Busze, F. Bouwens, M. Konijnenburg, J. Santana, A. Breeschoten, J. Huisken, G. Dolmans, and H. de Groot. A 2.4GHz ULP OOK Single-Chip Transceiver for Healthcare Applications. In *Proceedings of the 58th Solid-State Circuits Conference (ISSCC 2011)*, pages 458–460, IEEE International, 2011.

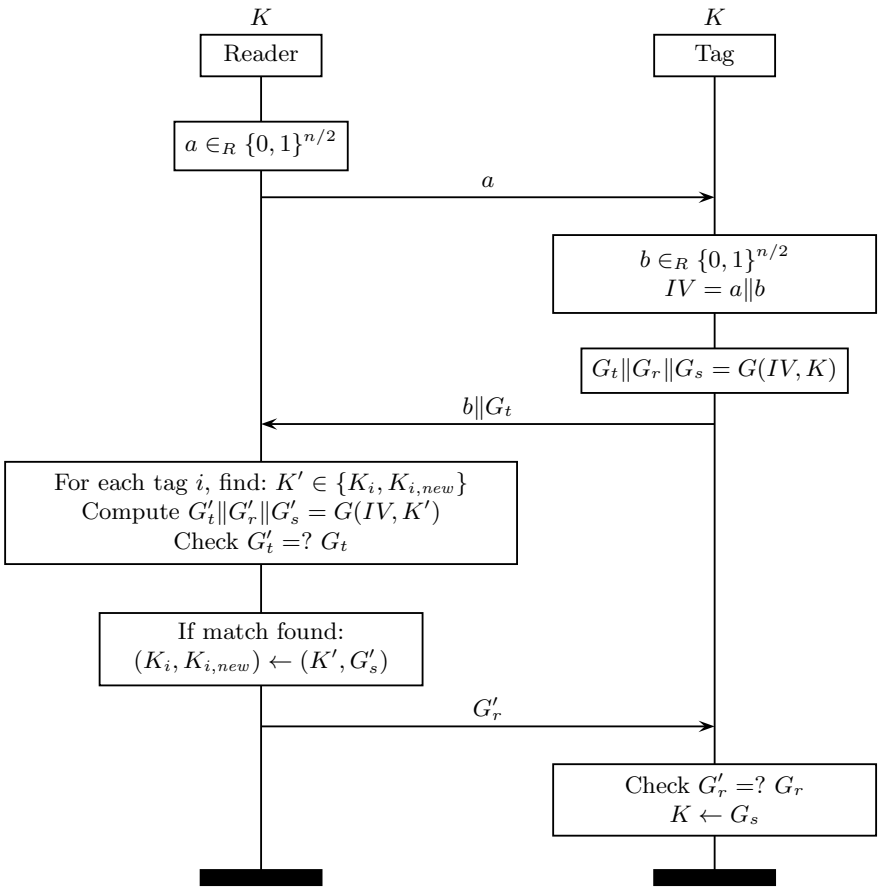

Fig. 3: Stream cipher based PEPS authentication protocol [3]

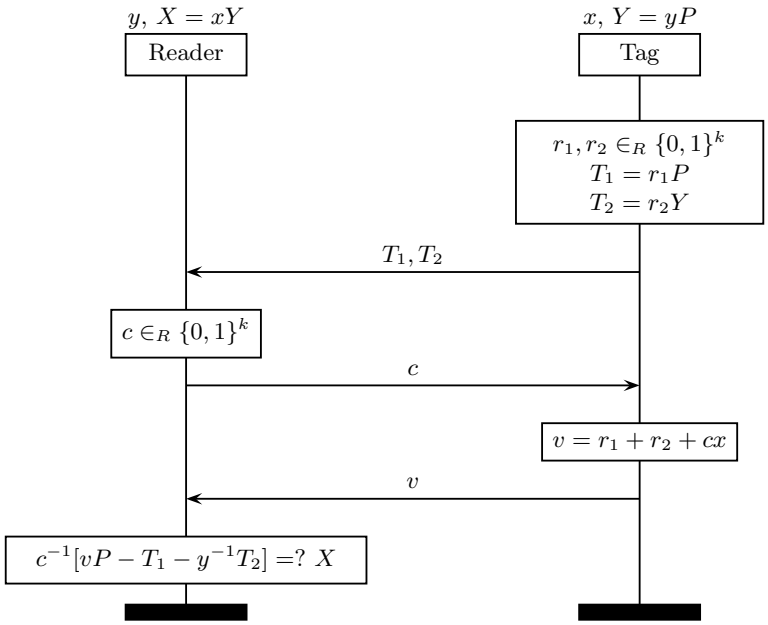

Fig. 4: ECC-based Randomized Schnorr protocol [6]