This is the very first version of the paper "RFID Security" by Denis Trček and Pekka Jäppinen" that is to be published in RFID and Sensor Networks: Architectures, Protocols, Security and Integrations", Auerbach Publications, Taylor & Francis Group, edited by Zhang Y., Yang L.T. and Chen J.

NON-DETERMINISTIC LIGHTWEIGHT PROTOCOLS FOR SECURITY AND PRIVACY IN RFID ENVIRONMENTS

Denis Trček*, Pekka Jäppinen**
*Department of Communication Systems
»Jožef Stefan« Institute
Jamova 39, 1000 Ljubljana, Slovenia
and
**Lappeenranta University of Technology
Skinnarilankatu 34, 53850 Lappeenranta, Finland

Abstract. The permanent need for lightweight protocols has been recently stimulated by emerging ubiquitous computing in which devices with limited computing resources are going to play an important role. Preserving security and privacy in such environments is not an easy task. This paper therefore provides an extensive overview of the field, and identifies some weaknesses of existing protocols that have not been addressed so far. Based on this, two new non-deterministic cryptographic protocols are presented that are designed to provide security and privacy in environments with devices that have weak resources, most notably radio-frequency identification tags (RFIDs).

Keywords: lightweight cryptographic protocols, non-deterministic protocols, ubiquitous computing, security, privacy.

1. Introduction

Current computing trends are shifting towards wireless communications that enable ubiquitous computing paradigms. Within such environments the majority of devices will be devices with limited computing resources, be it processing power, available storage, or power supply. The main representative among these devices will be radio-frequency identification tags or RFIDs (according to Gartner Group their expected market share is to reach three billion US $ by year 2010 [6], especially due to their wide use in retail [18]).

RFID tags consist of a microchip and an antenna, both encapsulated in polymeric material. The microchip has encoded identification (*ID*) data, and communication takes place on radio-frequencies by electro-magnetic coupling between readers and tags. A reader induces a voltage in the tag's antenna and this provides sufficient power for a tag to respond. In contrast to such tags, powered by coupling and called passive, tags can also have a battery – these are active tags. Passive tags are cheaper, they have an operating perimeter up to 3 meters and a relatively high error rate. On the other hand, active tags, which are more expensive, have an operating perimeter up to a few hundred meters, and lower error rate. Both kinds of tags can be read only, write once-read many, or rewritable.

So RFIDs are soon expected to be main communication devices in ubiquitous computing environments with application areas ranging from retail to health-care systems. Therefore security and privacy are becoming increasingly important, not only from the users' perspective and expectations, but also a from legislative viewpoint. Knowing that RFIDs are weak processing devices, this means a significant challenge for assurance of security and privacy.

This paper presents new protocols for provision of security and support of privacy in RFID environments that meet the above stringent conditions. In the second section, the current status of the field is presented, where existing protocols with known weaknesses are given. In addition, some new weaknesses of existing protocols are described. Based on these lessons, two new lightweight and non-deterministic protocols for RFID environments are presented and analyzed. Conclusions are drawn in the fourth section, and the paper ends with references.

2. Current status of the field

This section first gives an overview of cryptographic primitives related issues (security mechanisms), followed by cryptographic protocols related issues (security services).

2.1. Overview of cryptographic issues

The main barrier to security and privacy implementation is price. Recent reference RFID implementation was expected to have the following characteristics. It was passively powered and had 96 bits of read-only memory that carry the tag's identity (*ID*), which is unique for each tag. Chip operated at speed providing 200 read operations per second. It was estimated that a maximum of 2000 gates could be allocated to security within the economically acceptable range [weis03]. Taking into account Moore's law (and being a bit conservative), the upper limit is now approaching 4000 to 5000 gates.

This still puts stringent requirements on security and privacy protocols, which have to be lightweight. Although many protocols in the literature are claimed to be lightweight, they are based on many hidden assumptions that do not take into account the additional gates that are needed for implementation. For example, they assume that use of one-way hash functions automatically qualifies a protocol to be lightweight. But this does not hold true for the majority of functions like MD-x or SHA-x family [5]. Further, each additional step in a protocol often requires additional dedicated circuitry – steps in security protocols are semantically related and, as a consequence, each additional step results in a more complex algorithm that has to be implemented at the RFID tag.

Therefore only particular crypto-primitives can be chosen, most notably lightweight AES [4] and lightweight DES (DESL) [16]. With lightweight AES, roughly 3400 gate equivalents are used and the circuit is optimized for low-power operation. With DESL, authors claim that a comparable strength to that of AES is achieved, with 45% less chip size, 86% fewer clock cycles and roughly 1800 gates. The latter can encrypt 64-bit plaintext in 144 clock cycles. DESL is particularly appropriate for our purposes. It will be the basis for producing 128 bit long hashed values (various principles of using symmetric block ciphers for one-way hash functions can be found in [19]).

2.2. Overview of crypto-protocols issues

Because of the above constraints, protocols for authentication and privacy should consist of as few steps as possible, where a simple “challenge – response” remains the most desirable architecture. If more rounds in a protocol are needed, then it is preferred that these additional messages are syntactically equivalent, which means that the same circuitry can be used (of course, with a different input). This is also in line with the requirement that serialized computations are preferred over concurrent ones. What matters with passive tags is power consumption per clock cycle (mean power consumption minimization). Therefore concurrent (»parallel«) computations should be replaced by serialized ones [4].

The following threats can be identified in the area of RFID protocols:

- Man-in-the-middle attack is done in a way where an adversary modifies challenges with its own data; an appropriately selected challenge may mislead a tag and / or a reader to believe that they communicate directly one with another, which is not the case, whereas the messages are handed over and modified by an adversary.
- Passive attack is an attack in which enough information can be obtained by simply monitoring the communication between a tag and a reader.
- Active attack is an attack where an adversary is actively involved in the communication and modifies messages (man-in-the-middle is a kind of active attack).
- Reply attack is an attack, where an adversary records exchanged messages and simply reuses later without necessarily knowing what is contained in them or how actually to calculate the content of these messages.
- Relay attack is an attack, where relaying of messages is deployed between a tag and a reader to falsely convince the reader that the tag is in its close proximity so it can act accordingly.
- Malicious reader attack can be of many kinds, but we will concentrate on the unauthorized tracking attack, where a malicious system tracks a tag without necessarily knowing its true identity, but just recognizing it in various places on the basis of its responses to challenges.
- Physical attack is an attack, in which tag's content is read directly from a circuitry instead of using wireless connection.

The last kinds of attacks will not be addressed, because preventing such attacks significantly increases the cost of RFID tags and cannot be justified by the intended applications.

RFID protocols can be divided into single tag protocols and multiple tag protocols. Further, each of these groups can be divided into single round protocols and multiple round protocols. The following multiple round single tag protocols should be mentioned (summarized from [15]):

1. Protocol of Weis, Sarma, Rivest and Engels [20] – a reader and a tag share a secret x. After being triggered by the reader's request, the tag produces a random r and computes a value $(r, (ID \,||\, H(ID)) \oplus f_x(r))$, which is sent to the reader (here “||” denotes concatenation of strings, “⊕” bitwise XOR operation, and “f_x “ a pseudo-random function that uses secret x as a parameter). After verification, the reader replies with the tag's ID. It is evident that exposure of the

plain *ID* in the third step can be problematic, not to mention that replay attacks are trivial, because the first and the second message are cryptographically independent.

2. Protocol of Henrici and Muller [9] – a reader sends a request, after which a tag calculates $H(ID)$ and $H(s \circ ID)$. Next, it sends $H(ID)$, $H(s \circ ID)$ and δs to the reader (" $\circ$ " denotes some chosen operator, "*s*" the serial number of the step, and "δs" the difference between current and previous session number - this difference is equal to 1 when the previous transaction is valid). After receiving this message, the reader computes a new *ID* for the tag ($ID \leftarrow ID \circ r$), updates the database and sends *r* and $H(r \circ s \circ ID)$ to the tag. After receipt, the tag is able to verify the integrity of *r* and is able to calculate the new *ID* ($ID \leftarrow ID \circ r$). Therefore the tag and the reader are supposed to stay in synchronism. However, one problem with this protocol is that an attacker can achieve database desynchronization if XOR is used for " $\circ$ ". In this case, an attacker replaces *r* in the third step by a zero-bits string, and as a result $H(r \oplus s \oplus ID) = H(s \oplus ID)$ is obtained. This value is the same as the value from the second step, which the tag has sent to the reader and which can also be read by attacker. Therefore, when the tag checks messages from the third step, it updates its new *ID* with a value that differs from that calculated by the reader. The result is database desynchronization.
3. Protocol of Ohkubo, Suzuki and Kinoshita [14] – with this protocol, a tag and a reader share two hash functions *G* and *H*, and an initial secret s_i. A reader sends a request to the tag, and this triggers tag to compute a new secret by calculating $H^1(s_i) = H(s_i)$ and storing this new value. At the same time, the tag computes $G^1(s_i) = G(s_i)$ and sends this value to the reader. The back-end database hashes each of the stored secret values and finds a matching pair (ID, $G^1(s_i)$). In the second run, $H^2(s_i) = H(H(s_i))$ and $G^2(s_i) = G(G(s_i))$ are calculated and used, etc. However, this version is vulnerable to replay attacks, because the second message that is sent from the tag to the reader is not linked to the first message. Therefore an adversary can send a request to the tag and record the reply, to use it at some later time to respond to the reader. According to [15], Avoine, Dysli and Oechslin propose a solution, whereby the first message contains a fresh challenge *r*, and the second message is calculated as $G(s_i \oplus r)$ [1].
4. Protocol of Molnar and Wagner [13] – this is an example of a protocol which is supposed to be without flaws. A tag and a reader share a secret *x*. Initially, the reader chooses random r_r and sends it to the tag. The tag chooses random r_t , computes $\sigma_1 = ID \oplus f_x(0, r_r, r_t)$, and sends it to the reader. The reader uses σ_1 to retrieve *ID* by calculating $ID = \sigma_1 \oplus f_x(0, r_r, r_t)$, and then replies with $\sigma_2 = ID \oplus f_x(1, r_r, r_t)$. After receiving this message, the tag checks if *ID* is okay.

However, some shortcomings still exist, even with the latter two protocols, which have not been previously described:

- With regard to the modified Ohkubo, Suzuki and Kinoshita protocol, if an adversary always sends the same challenge, the response from the tag will always be the same. This is a serious problem for privacy, because the tag becomes traceable despite its unknown identity.
- In case of the improved version of the Molnar and Wagner protocol, the shortcoming is related to the third message. What happens if there is no match after the tag receives this message? If the tag is supposed to react, this case certainly means a more complex protocol with additional steps, but these steps

are missing. And without these additional steps, using the protocol as given in [15], this means that an adversary can tweak some bits in the second message and a wrong *ID* is determined by the reader. So the protocol does not assure the integrity of the checked *ID*.

As described in [15], many protocols that deploy XOR function can be successfully attacked by submitting somehow a zero vector (all bits being zero) as an input to computation (see for example the protocol of Henrici and Mueller). The reason is straightforward – any bit sequence XORed with a zero vector produces the same bit sequence. So in such cases it is a wise practice to check that the input is not a zero vector. Similar resonoing applies when a unit vector (all bits are set to 1) is used for XORing – this leads to negation of the input bit sequence.

All the above described protocols are, so to say, deterministic protocols. A different approach has been taken by Hoper and Blum with the HB protocol [10] (its successors are HB+ and HB++ [12, 2]). All HB variants are based on the Learning Parity with Noise (LPN) problem. This problem requires an attacker to calculate a *k*-bit secret *x*, shared between a reader and a tag, after being given several calculations of $b_i = a_i \bullet x \oplus \upsilon_i$, where υ_i (also called noise) is equal to 1 with a probability that takes on values from the interval [0, ½). Since the probability of noise being 1 is strictly less than 0.5, an adversary can challenge a tag with some chosen *a* several times successively. Once *k* equations with linearly independent *a*-s have been obtained, *x* can be recovered by Gaussian elimination. Further, exploitation of this principle is the basis for active attacks, to which HB protocol family is not resistant. There are other weaknesses of this family like man-in-the-middle for HB+, which are described in [7].

Later, some other representatives of single tag protocols followed – an extensive overview with the description of their vulnerabilities is given in [15]. In addition, this paper also covers multiple tag protocols extensively, which are intended for scenarios where the simultaneous presence of two tags in a reader's field is required.

3. New non-deterministic cryptographic protocols

The protocols introduced in this section are suitable for single tag and multiple tag applications. They are non-deterministic (ND protocols), meaning that when a reader gets a response from the tag, the expected values of this response lie in a certain interval. The reader has to check all possible values within this interval to find a match. Such protocols put the majority of the computational workload on the reader and back-end systems. In most cases this is acceptable, especially in the case of RFID architectures where significantly larger computational resources are available at the back-end side (such a principle is common nowadays, especially with digital signatures).

3.1. The first ND protocol

The first protocol goes as follows (see Fig. 1). A reader and a tag share a common secret *x,* and both are able to compute the same hash function *H*. The tag and the reader also share *n* that determines the interval for calculation of random values Δt (this *n* does not need to be secret).

Now the authentication process takes place:

1. The reader challenges the tag with a time-stamp.
2. After receiving the time stamp, the tag optionally checks it against past received values. It is unreasonable to assume that the RFID tag will have autonomous time circuitry and, to prevent replies, former values have to be stored. Due to limited resources, the memory for storing received challenges is FIFO, consisting of, for example, 4 locations; when the 5th value is received, the first value is overwritten. So if the time stamp value is fresh, the tag stores it and computes random Δt from the interval [0, n -1], meaning that it may have n different values. The tag concatenates secret s with ($t + \Delta t$) and hashes the string.
3. The tag sends the result from the previous step to the reader.
4. On receipt of the message from the second step, the reader starts calculations to find a match. For this match to be found, the reader calculates $H(s \| (t \oplus 0))$, ..., $H(s \| (t \oplus (n-1)))$ with s-es being taken from pairs (ID, s) that are stored in the database. If a match is found, the tag is authenticated.

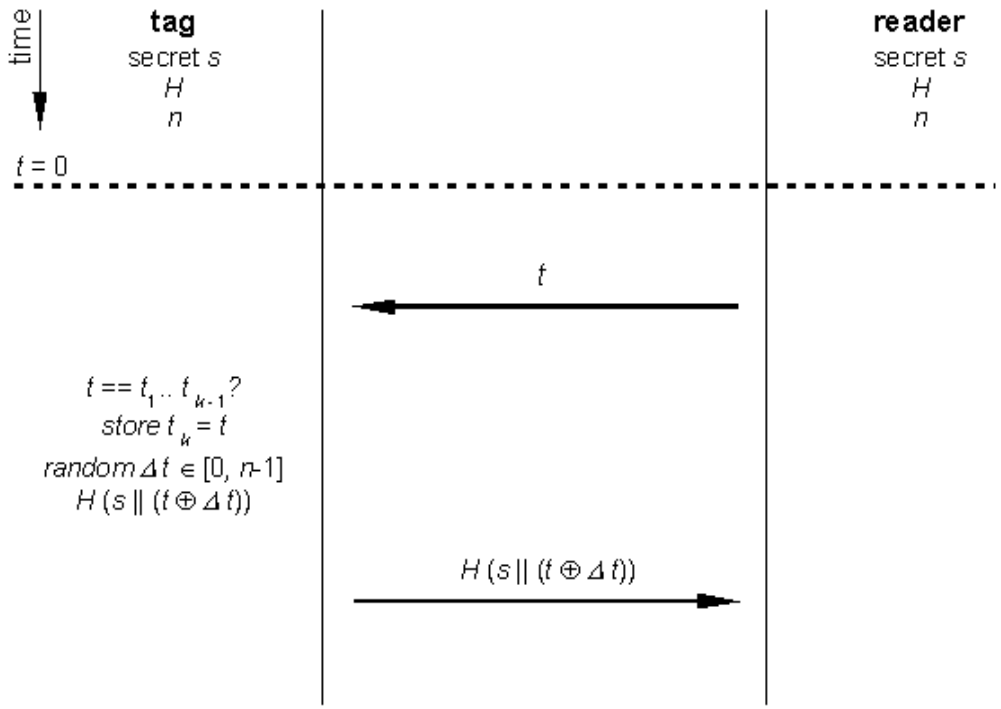

Figure 1: The first ND protocol

Random challenge is optionally checked for freshness in the second step to protect privacy. If a malicious reader constantly uses the same challenge, the responses of the tag will always be the same and the tag will be traceable. Of course, due to limited resources of a tag the list of all stored challenges can be currently relatively short, but available memory will grow in the future and this step can then become obligatory. However, the analysis at the end of this section shows that optional checking of freshness can already be mandatory with available technology, especially if 48 bits are allocated for challenge, which means that eight challenges can be stored in four 96-bit memory locations.

3.2. The second ND protocol

The second protocol goes as follows (see Fig. 2). The reader and the tag are able to compute a hash function *H.* A tag is given a secret *s* that is also known to the reader. Again, the tag and the reader share *n* that determines the interval for calculation of random values Δr.

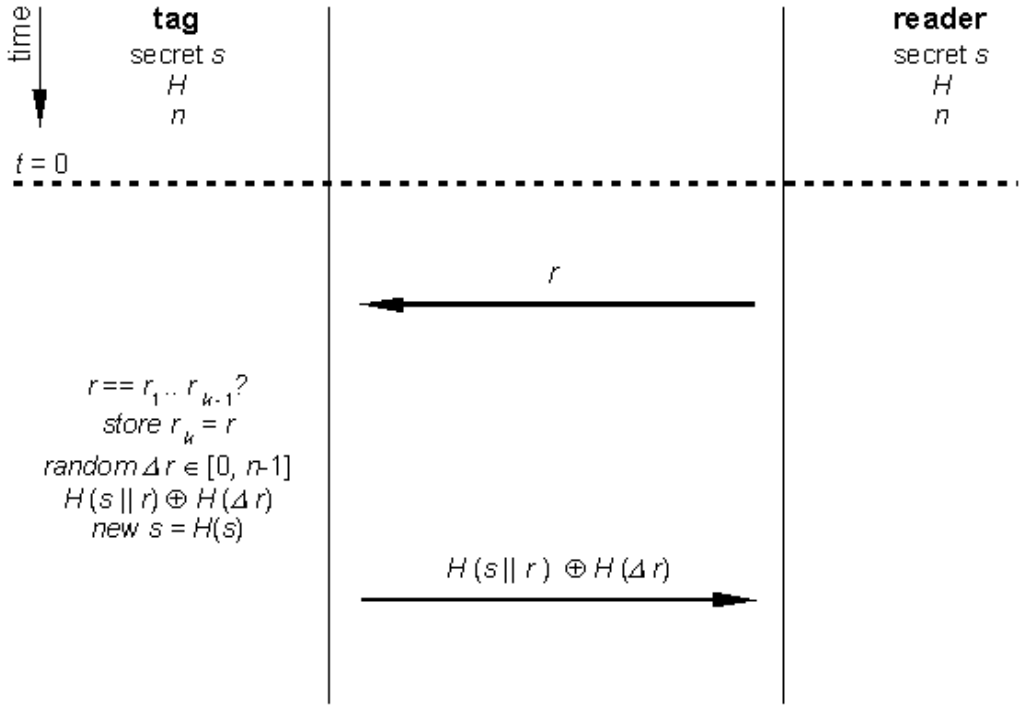

Figure 2: The second ND protocol

Now the authentication process starts taking place:

1. The reader sends the challenge *r* to the tag.
2. On receipt, the tag optionally verifies if the received *r* is on the list of already used challenges. If not, it stores the received challenge, and calculates random Δr. Afterwards, the tag computes *H* (*s* || *r*) and further randomizes this result by XORing it with *H*(Δr). It sends this result to the reader.
3. On receipt of the message from the previous step, the reader calculates *H* (*s* ||*r*) ⊕ *H*(Δr = 0), ..., *H* (*s* ||*r*) ⊕ *H*(Δr = *n* -1) until a match is found that authenticates the tag. Of course, the reader has access to a database with pairs (*ID*, *s*).

This protocol has some important properties that have to be discussed. It is based on a random challenge that is optionally checked by the tag for freshness to protect privacy from malicious readers. Again, because available memory will grow in the future, this step can then become obligatory. Next, after the challenge is checked, it is concatenated with the secret *s* and only then hashed. Applying the hash function to two arguments (*s* and *r*) is often done by XORing *s* and *r* (such an example is the modified protocol of Okhubo, Suzuki and Kinoshita [1]). This can cause problems if *r* consists only of zero-bits, or bits that are all set to 1. In the former case, XOR operation results in *s* itself,

while in the latter case XOR results in negation of *s*. By concatenating arguments (instead of XORing them) this problem is avoided and there is no need for additional circuitry to check whether all bits in a challenge are 0 or 1. Finally, this result is XORed with hashed Δr. This supports multiple tag applications. By offsetting the initial value in circuitry that produces Δr, the two values for Δr will differ, and so will the final result. Thus an attacker will not know that tags are actually responding as twin-tags. The reason for hashing Δr is that the interval of these values is relatively small, e.g. defined by 8 bits. Without hashing, only the last eight bits of $H(s \| r)$ would be affected.

3.3.A brief analysis of ND protocols

Before going into details it should be stated that readers are assumed in our scenarios to belong to a secure environment, as well as the back-end part of a system.

Security and privacy

For the two ND protocols for RFID environments it can be concluded that all the messages are unique, look random to a third party, and are optionally checked for freshness as well as responses (of course, in engineering reality this uniqueness will certainly be limited because of the number of available FIFO memory locations and limited extent of Δt and Δr intervals).

Further, the first message in both protocols is tied cryptographically to the second message. Thus active and passive attacks are prevented. Reply attacks are also prevented. Malicious reader tracking is impossible because of constantly changing messages. However, preventing physical attack remains an open issue, as discussed in the first section.

Further, relay attacks can be prevented by adding a distance-bounding protocol developed by Hancke and Kuhn [8]. This is possible, because the protocols are logically independent – the Hancke and Kuhn protocol is aimed at distance-bounding, but the ND at authentication and privacy.

Consumption of resources

Let us provide some quantitative estimates of the number of (NAND) gates needed for implementation of the above ND protocols:

- Storing one bit requires 5 gates (assuming D flip-flops).
- Due to the fact that block ciphers can be used for cryptographic hashing, our assumption for implementation is DESL that requires approx. 1800 gates [16]. Although using block ciphers for hashing is not efficient for ordinary implementations, in the case of RFIDs it makes sense (dedicated hash functions that are implemented in software are faster on ordinary computing devices, however if they are implemented in hardware they significantly exceed the number of the gates required for DESL).
- Values Δr *and* Δs are generated with an implementation that is suitable for lightweight purposes and that deploys a shift register. With this implementation, the shift register has an XOR feed-back loop, where one input is the output of the shift register, and the other input is the *n*-th bit in the register. The output of the XOR gate is fed into the first storage cell of the shift register. A register with *m* bits can represent 2^m different values, however all zeros would stuck the circuit,

so the actual number of different values is 2^m-1. By choosing an appropriate n, the resulting sequence is pseudo random and of a maximal length, if m is such that $p(x) = x^m + x^n + 1$ is irreducible over GF [11]. A shift register with 4 bits requires approx. 60 gates, with 8 bits approx. 120 gates, and so on. Therefore assuming 8 bits for our implementation 120 gates are needed.

- Optionally k = 4 (optional) n-bit locations are needed for storage of used r and t values, and one location for secret s (ID). Therefore, assuming 96-bit values this means (4+1) *96 = 480 bits and 2400 gates. Note that one n-bit location can store 3 challenges r and t, if these are 32 bits long, so the total number of stored challenges in this case is 12.
- Bitwise XOR requires 4 gates, so XORing 128 bits requires 512 gates (this is the figure for the second protocol, while the first one requires only 8*4=32 gates).

A rough estimate of the total cost, using the above values, would be approx. 4800 gates, where logic gates that are needed for comparisons of freshness of received challenges are not included. So the main cost is contributed by storage cells that are needed for freshness checks. If optional freshness checks are excluded, the number of gates is approx. 2400, which certainly qualifies the above protocols as lightweight. But even with the inclusion of optional steps, ND protocols stay within the limits for RFID implementations.

4. Conclusions

The area of security and privacy of RFID protocols is becoming increasingly important. This is also because of legal requirements, and there have been recent discussions as to whether legislation is needed that specifically addresses RFIDs [17]. Although an official statement has been made by the EU that RFID-tailored legislation will not be introduced, this does not mean that security and privacy for RFIDs is of no concern. Other laws certainly remain applicable to ubiquitous computing environments, the most notable one being the Data Privacy Directive [3].

In this paper we have studied some of the most important protocols for authentication and assurance of privacy. We have described their weaknesses that have been found in the literature so far. In addition, we have described some new weaknesses. Taking this into account, we have introduced two new lightweight cryptographic protocols. These protocols are non-deterministic and require minimal resources on the tag's side. To achieve this, heavier computations are put on the reader / back-end side, which is acceptable in the majority of cases. But because of these basic properties, our ND protocols effectively provide authentication and assure privacy. We have analyzed briefly their resistance to known attacks and we believe that they fulfill their intended use.

5. References

[1] Avoine G., Dysli E., Oechslin P., Reducing Time Complexity in RFID Systems, Proceedings of the 12th Annual Workshop on Selected Areas in Cryptography, pp. 291–306, 2005.

[2] Bringer J, Chabanne H., Dottax E., HB++: A lightweight authentication protocol secure against some attacks, IEEE International Conference on Pervasive Services,Workshop on Security, Privacy and Trust in Pervasive and Ubiquitous Computing, 2006.

[3] European Commission, Privacy and Electronic Communications Directive, 02/58/EC, Official Journal of the European Communities, L201, 31/7/2002, Brussels, 2002.
[4] Feldhofer M., Wolkerstorfer J., Rijmen V., AES implementation on a grain of sand, Information Security, IEE Proceedings, Vol. 152, No. 1, pp. 13 – 20, London, 2005.
[5] Feldhofer M., Rechberger C., A Case Against Currently Used Hash Functions in RFID Protocols, Workshop on RFID Security Security 06, Graz, 2006, http://www.iaik.tugraz.at/aboutus/people/feldhofer/papers/RFIDSec06_slides.pdf.
[6] Gartner Group, RFID Market $3 Billion in 2010, RFID Update, December 13, 2005, http://www.rfidupdate.com/articles/index.php?id=1014.
[7] Gilbert H., Robshaw M., Sibert H., An active attack against HB+ - a provably secure lightweight protocol, IEE Electronic Letters 41(21), pp. 1169–1170, 2005.
[8] Hancke G.P., Kuhn M.G., An RFID distance bounding protocol, Proceedings of the IEEE/Create-Net SecureComm, pp. 67–73, 2005.
[9] Henrici D., Müller P., Hash-based enhancement of location privacy for radio-frequency identification devices using varying identifiers, Proceedings of the 1st International Workshop on Pervasive Computing and Communication Security, pp. 149–153, 2004.
[10] Hopper N.J., Blum M.M., Secure human identification protocols, Advances in Cryptology — ASIACRYPT 01, Lecture Notes in Computer Science, vol. 2248, pp. 52–66 , Springer, 2001.
[11] Horowitz P., Hill W., The art of electronics, Cambridge University Press, New York, 1989.
[12] Juels A., Weis S. A., Authenticating pervasive devices with human protocols, Advanced in Cryptology - CRYPTO'05, Lecture Notes in Computer, vol. 3126, pp. 293–308 Springer, 2005.
[13] Molnar D., Wagner D., Privacy and security in library RFID: issues, practices, and architectures, Proceedings of the 11th ACM conference on Computer and communications security, ACM Press, 2004, pp. 210–219.
[14] Ohkubo M., Suzuki K., Kinoshita S., A cryptographic approach to a 'privacy-friendly' tags, RFID Privacy Workshop, MIT, November 15 2003.
[15] Piramuthu S., Protocols for RFID tag/reader authentication, Decision Support Systems, Vol. 2007, No. 43, pp. 897-914, Elsevier, 2007.
[16] Poschmann A., Leander G., Schramm K., Paar C., New Light-Weight Crypto Algorithms for RFID, Proc. of the IEEE International Symposium on Circuits and Systems - ISCAS 2007, New Orleans, 2007 (forthcoming)
[17] Pritchard S., CeBIT 2007: Europe opts out of RFID regulation, PCPro, Dennise Publishing Ltd., March 15, 2007, http://www.pcpro.co.uk/news/107699/cebit-2007-europe-opts-out-of-rfid-regulation.html.
[18] Roussos G., Enabling RFID in Retail, Computer 39(3), 25-30, Los Alamitos: IEEE, 2006.
[19] Schneier B., Applied cryptography, 2nd edition, John Wiley & Sons, New York,1995.

[20] Weis S.A., Sarma S.E, Rivest R., Engels D.W., Security and privacy aspects of low-cost radio frequency identification systems, Proceedings of the 1st Security in Pervasive Computing, LNCS, vol. 2802, pp. 201–212, Springer, 2004.

Author's biography

Prof. Dr. Denis Trček is principal investigator and research group leader at Jožef Stefan Institute. He has been involved in the field of computer networks and information systems security and privacy for almost twenty years. He has taken part in various European projects, as well as domestic projects in government, banking and insurance sectors. His bibliography includes over one hundred titles, including works published by renowned publishers like Springer and John Wiley. D. Trček has served (and still serves) as a consultant and a member of various international bodies and boards, from editorial to professional ones. He is inventor of a patented family of light-weight cryptographic protocols. His interests include e-business, security, trust management, privacy and human factor modelling.

Pekka Jäppinen received his M.Sc. and D.Sc. degrees in information technology from Lappeenranta University of Technology in 2001 and 2004. He has been working at the Communications Engineering laboratory since 1995. His research interests include short-range wireless communication, communication protocols, personal information management, security and privacy.

Acknowledgements

D. Trček would like to thank to ARRS (Slovenian Research Agency) for financial support of this research (grant number J2-9649). P. Jäppinen would like to thank Nokia Foundation for the Nokia Visiting Researcher grant that made possible the visit to Josef Stefan Institute.