Improved Lightweight Mutual Authentication Protocol for RFID Systems

Győző Gódor and Mátyás Antal

Győző Gódor and Mátyás Antal
Department of Telecommunication, Budapest University of Technology and Economics, Magyar Tudósok körútja 2., Budapest, Hungary H-1117, e-mail: godorgy@hit.bme.hu

Abstract The usage of the RFID technology is becoming more and more widespread, however, the systems are quite vulnerable regarding security aspects. The authentication protocols developed for computer networks are not suitable for RFID use, because the tags have very limited memory and computing capacity. New solutions are needed, which requires less operations and fewer messages. The lightweight protocols are one of the currently known methods for this purpose, but the proposed algorithms do not satisfy all of the security requirements. In this paper we introduce our lightweight authentication protocol, which provides prevention against all known attack schemes. We provide a full security analysis and prove the correctness of our solution with formal verification using GNY logics.

1 Introduction

The potential uses of radio frequency identification is nearly endless: logistics, entrance systems, medical applications, measuring traffic, etc. However, radio based identification implies new types of problems - just like in mobile communication and wireless networks - since the radio channel that carries the signals are accessible by everyone who is nearby, thus providing a chance to eavesdrop or attack it. One of the most important problems to solve is to allow the communicating parties to identify and authorize each other in a secure manner, which means they can be sure that the other one is indeed who she says to be.

RFID systems consist of three main functional units: the items to be identified are called tags, the device with which the tags are communicating is called a reader and the back-end, which controls the readers and has access to the database.

Tags are usually very small devices that contain a simple processing unit, a small amount of ROM and/or RAM and a small sized coiled antenna. Tags may be equipped with an own power source, i. e. a battery, in which case

they are so called active tags. In contrast, passive tags, since they do not have any source of power, are capable of communicating only if a reader has initiated a connection beforehand. The amount of energy needed for memory operations, calculations and transmitting the reply message is retrieved from the radio waves emitted by the reader. This means that this type of tags have very limited capabilities regarding the number and complexity of calculations as well as the length of the reply messages. The main goal is to implement a communication protocol between the tag and the reader which allows the parties to exchange data securely, whilst it does not require the tag to be too complex to keep its production cost reasonably low.

The back-end can command a reader to retrieve the identification number stored in an RFID tag. The reader acts very similarly to a simple optical bar code reader in a sense that it forwards every data received from its radio interface to the back-end on a secure, standardized channel, i.e. RS-232 or Ethernet. The identification number read from a tag is processed by the back-end. Its task is to carry out authentication: to check if the ID provided can be found in the database and, depending on the application of RFID, verify if the tag is allowed to use the service, etc. The back-end may be anything from a simple PC to an enormous cluster of servers. When discussing lightweight authentication protocols we suppose the back-end has (at least compared to the tags) unlimited computational capacity. Naturally, in larger systems the back-end may easily become the bottleneck, thus we must take its performance and scalability into consideration as well.

2 Security Requirements of an RFID System

Since the communication between the tags and the reader takes place on a radio channel, practically anyone can eavesdrop the identification number sent by the tags or the requests issued by the reader. Moreover, it is quite possible that an adversary impersonates a reader and requests tag A to give out her identification number. In the possession of that information, he can later personate tag A by demonstrating its ID to a legal reader. If this happened in a supermarket, the worst case would be that someone bought a plasma TV at the price of a soft drink. But if an attacker was able to authorize himself as a government agent and got into a top secret area, it would possibly mean the end of someone's job. Nevertheless it is desirable that a tag gives out its identity only if it has made certain that the reader was authentic and not being an attacker trying to get valuable information. Besides that the reader should authenticate the tag as well.

The messages sent by the tag during the authentication process can be seen by anyone, thus it is very important that an attacker should not be able to gain any valuable information from these. This criteria is fulfilled if

the transmitted data on the channel is completely undistinguishable from random bits.

Since there can be possibly more than thousands of RFID tags in a system, their production price should be kept very low. This leads to the fact that they cannot be equipped with tamperproof memory chips, which would ensure that no data can be read out without permission. But it is desirable that even if the attacker knows the identity of a tampered tag, he will still not be able to find a connection between the previous and current transactions of the tag. This property of tags is called untraceability. Untraceability can be achieved if tags do not reveal their identity to the reader directly, but use a unique identification number in each transaction, which is only known by the reader and themselves.

A proper authentication protocol should ensure that:

- tags' and reader's messages cannot be eavesdropped
- tags are not traceable
- in case a tag is tampered no connection can be found to its previous transactions
- messages cannot be replayed
- man-in-the-middle type of attacks are prevented
- messages cannot be interleaved between different protocol runs
- denial of service attacks are prevented
- no-one can impersonate the tags or the reader
- messages sent by the tags are undistinguishable from random bits

3 Related Work

Protocols that require low computing capacity, such as carrying out simple bit operations and/or calculating one-way hash functions are called lightweight protocols. Because of the relatively low production price of tags running lightweight protocols, they are likely to become popular among consumer goods. There were several solutions, e.g. the Hash Lock Scheme [3] and the Extended Hash-chain Scheme [4] that aimed at fulfilling the security requirements shown in the previous section. The first one fails to prevent the traceability of tags, since the ID is constant between transactions. The latter one does not provide protection against man-in-the-middle attacks, since an adversary can easily stand between the tag and the reader catching and forwarding messages without being noticed. Also the tags need a good random number generator to make random numbers unpredictable which is very hard to implement on a CPU with low capacity.

Luo, Chan and S. Li proposed a new scheme, the Lightweight Mutual Authentication Protocol [1], based on Ohkubo's hash chain [2] and simple XOR operations that provided mutual authentication for the tags and the back-end. The motivation for using mutual authentication is to ensure that

the back-end, and thus the reader itself, is authentic before transferring the tag's ID to the back-end, whilst the back-end can verify whether the tag is an authentic one in the system. To achieve backward privacy, which means that secret values used in previous transactions cannot be recovered even if a tag gets compromised, Ohkubo proposed a hash chain of length n. In the i^{th} transaction the tag sends $a_i = G(s_i)$ to the reader and renews its secret by calculating $s_{i+1} = H(s_i)$, where s_i is determined recursively and G and H are hash functions.

The RFID tag stores an initial s_0 secret value as well as its ID, which can be loaded into its memory during production. To ensure that the tag remains untraceable by a third party who keeps track of the secret values, it is periodically changed after every transaction. For this purpose a s_i secret value is used for the i^{th} transaction as the input for the hash chain and get the a_i output after n rounds of the one-way hash function G in order to maintain backward privacy.

In that scheme the tags have three readable/writeable memory blocks denoted by W_1, W_2 and W_3. Also there is a separate memory block reserved for keeping track of the c_i transaction counter. The back-end stores the s_0 secret value, the current values of the W_1, W_2 and W_3 blocks as well as the tag's ID. The messages of the protocol run can be seen on Figure 1.

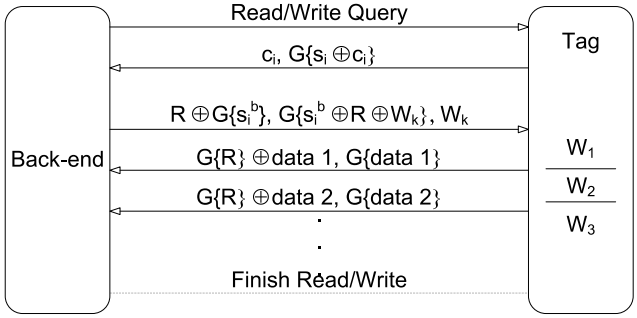

Fig. 1 Luo, Chan and S. Li's Lightweight Mutual Authentication Protocol

The main problem of the proposed scheme is that an adversary can easily replay the second message sent by the tag, thus impersonating it without being noticed. The reason why the protocol does not prevent replay attacks is because the back-end does not use any challenge to ensure the freshness of the tag's response.

Also, because in order to authenticate a tag the back-end needs to calculate hash c_i times for every tag in its database, it can easily become the performance bottleneck in the system. It is desirable to lower the number of required hash operations.

4 Improved Lightweight Mutual Authentication Protocol

Our proposed improvement for the above protocol can be seen on Figure 2.

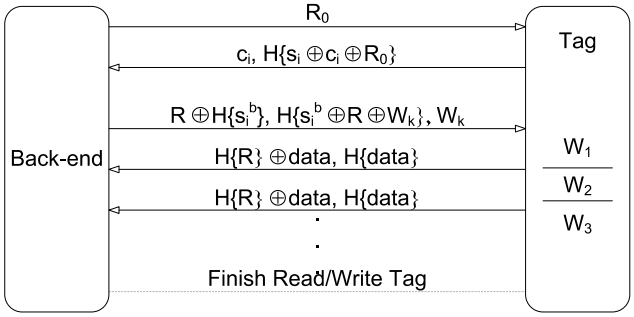

Fig. 2 Improved Lightweight Mutual Authentication Protocol

Instead of Ohkubo's hash chain, we use a simple iterative hash in every transaction. In the beginning let the secret value be s_0, which can be loaded into the tag during production. After a successful authentication, s_{i+1} is calculated by hashing s_i.

In the i^{th} transaction a challenge (R_0) is sent by the back-end, which enables it to verify whether the tag has responded recently. The tag calculates the XOR value of the transaction counter (c_i), the current secret value (s_i) and the back-end's challenge. The tag sends the transaction counter and the hash value of the XOR's output. The challenge is needed to prevent adversaries from replaying the tag's message at a later time.

Given c_i, the back-end does an exhaustive search to find the tag's current secret value in its database, for which $H(s_i^b \oplus c_i \oplus R_0) = H(s_i \oplus c_i \oplus R_0)$, where s_i^b is derived from s_{i-1}^b by hashing. Because of the cryptographic hash function's preimage collision resistant property described in Rogaways's and Shrimpton's work [6], this holds if and only if $s_i^b = s_i$. If it finds a match in the database, it means the tag is authentic.

For mutual authentication the back-end still has to prove it knows the tag's current secret value. Also it is desirable to share a common random session number that can later be used by the parties to send encrypted data. Let this random number be denoted by R. The back-end's response consists of three parts: $R \oplus H(s_i^b)$, $H(s_i^b \oplus R \oplus W_k)$ and W_k.

The tag starts to decipher the message by calculating the XOR of $H(s_i)$ and the first part of the received message ($R \oplus H(s_i^b)$), thus it learns the random number R^t. It authenticates the back-end if and only if both $R^t = R$ and $s_i = s_i^b$ hold.

By the end of the protocol run the parties also agreed on which W_k memory block to read/write in the current transaction. The integrity of W_k is ensured by bounding it with s_i in the second part of the back-end's reply.

Also the tag may transfer encrypted data to the back-end by sending $G(R) \oplus data_i$. In order to ensure data integrity it also sends the hash value of the data. Knowing the R random session number, the back-end can easily decipher the message and check its integrity.

5 Security Analysis of the Protocol

When analyzing protocol security, data confidentiality and integrity, protection against well-known attacks as well as the requirements stated in Section 2 should be taken into consideration.

Data confidentiality and integrity the tags store no valuable system data except for their s_i secret value and the W_1, W_2 and W_3 memory blocks. This means that even if an adversary manages to compromise a tag, she will not be able to obtain any more information than what is stored in its memory. Since the messages sent and received by tags are transferred in hashed form, no information can be learnt by eavesdropping. Data integrity is ensured by adding the hash value of the data to the end of each message.

Tag anonymity the tag's messages are undistinguishable from random bits by a third party, because of the H hash function. Since no private data is sent in clear, tags remain anonymous towards any third party.

Untraceability of tags in the i^{th} transaction $H(s_i \oplus c_i \oplus R_0)$ is sent by the tag, where s_i is changed after a successful read or write and is unknown by third parties. To find a connection between s_i and s_k, where $0 \leq k < i$, an adversary would have to find the preimage of hash function H at least twice, which is known to be a hard problem. This holds even if the attacker manages to tamper a tag and learns the data stored in the tag's memory (eg. the current secret value: s_i).

Prevention of replay attacks the back-end uses a challenge in the first message to ensure the freshness of the tag's response. The adversary would have to wait until the same R_0 is chosen by the back-end for which she already has the response. If R_0 has a large entropy (and it can be assumed because it is generated by the back-end which has enough resources to generate such a R_0 random number), this is very unlikely to happen. The same holds if an attacker tries to replay messages between different protocol runs.

Prevention of man-in-the-middle attacks the second and the third messages of the protocol renders it impossible for an adversary to carry out the attack. Both the secret value and the random session number is transferred in a hashed form or is XOR-ed to a hash value. In order to stand between the two parties and manipulate the messages without being dis-

covered she would have to recover s_i from $H(s_i \oplus c_i \oplus R_0)$, which is not possible because the H hash function is preimage resistant.

Prevention of impersonating the parties the attacker has to know the common secret value s_i in order to impersonate any of the parties. Since the protocol never reveals s_i, and because H is preimage resistant, it is ensured that no-one will be able to impersonate the tag or the reader.

6 Verification of the Protocol's Correctness

6.1 GNY Rules Used For Verification

The postulates used for the proof can be found in [5], the lack of space prevents the authors to communicate them in this paper. Throughout the deduction we always use the same notation as the authors of GNY logics.

In that article ($I3$) postulate does not cover the case when such a formula is hashed which contains the S secret indirectly through an F function. So let us introduce a new postulate for this case and extend the GNY logics with the following:

$$(I3b)\ \frac{P \triangleleft \star H\left(F\left(X,S\right)\right), P \ni F\left(X,S\right), P \models P \overset{S}{\longleftrightarrow} Q, P \models \sharp(F\left(X,S\right))}{P \models Q \mid\sim F\left(X,S\right), P \models Q \mid\sim H\left(F\left(X,S\right)\right)}$$

6.2 Notation

To make the protocol verification more readable, let us introduce the following standard GNY logics notations:

$A : T$
$B : B$
$K_i : s_i$
$K_i' : s_i^b$
$N_a : R_0$
$N_b : R$
$H(X) : G(X)$

6.3 Idealization of messages

The idealized Improved Lightweight Mutual Authentication Protocol's messages can be written in GNY logics in the following form:

$(Ideal1)\ B \rightarrow\ A : \star N_a$
$(Ideal2)\ A \rightarrow\ B : \star c_i, \star H\left(\star F\left(\star F\left(N_a, \star c_i\right), \star K_i\right)\right)$, where $K_i = H^{c_i}(K_0)$
$(Ideal3)\ B \rightarrow\ A : \star F\left(\star N_b, \star H\left(\star K_i'\right)\right), \star H\left(\star F\left(\star F\left(\star W_k, \star N_b\right), \star K_i'\right)\right), \star W_k$

$(\ (Ideal4)\ A \rightarrow\ B : \star F\left(H\left(N_b\right)\right), data), \star H\left(data\right)\ )$

6.4 Assumptions

The following statements are believed to be true prior to the protocol's run:

$(A1)\ B \ni K_0$
$(A2)\ B \ni N_a$
$(A3)\ B \ni N_b$
$(A4)\ B \ni W_k$
$(A5)\ B \models B \overset{K_i}{\longleftrightarrow} A$
$(A6)\ B \models \sharp(N_a)$
$(A7)\ B \models \sharp(N_b)$
$(A8)\ A \ni K_0$
$(A9)\ A \ni c_i$
$(A10)\ A \models B \overset{K_i}{\longleftrightarrow} A$
$(A11)\ A \models \sharp(c_i)$
$(A12)\ A \models B\ \mathrm{Z}\Rightarrow \sharp(N_b)$
$(A13)\ A \models B \models \sharp(N_b)$

6.5 Aims

The main aim of the protocol is to provide mutual authentication. This consists of the following two aims:

1^{st} aim: the back-end ensures that the tag knows the K_i secret.
$(C1)\ B \models A \ni K_i$
2^{nd} aim: the tag ensures that the back-end also knows the K_i secret.
$(C2)\ A \models B \ni K_i$

6.6 Proof

We will deduce the aims from the idealized protocol messages using the assumptions we made. It can be seen during the deduction that no third parties are able to learn the K_i secrets at any time during or after the protocol run.

After receiving the 1^{st} message:

$$(M1)\ A \triangleleft \star N_a$$

Using the $(T1)$ postulate:

$$(D1)\ A \triangleleft N_a$$

From this using the $(P1)$ rule we get:

$$(D2)\ A \ni N_a$$

After receiving the 2^{nd} message:

$$(M2)\ B \triangleleft (\star c_i, \star H(\star F(N_a, \star c_i), \star K_i)))$$

Using rule $(T2)$:

$$(D3)\ B \triangleleft \star c_i$$

Using $(T1)$ postulate:

$$(D4)\ B \triangleleft c_i$$

From this using he $(P1)$ rule we get:

$$(D5)\ B \ni c_i$$

$$(A1)\ B \ni K_0$$

Using $(A1)$ and the $(P4)$ rule c_i times in a row we get:

$$(D6)\ B \ni H^{c_i}(K_0), \text{which means } B \ni K_i$$

$$(A2)\ B \ni N_a$$

From assumption $(A2)$ and statement $(D6)$ using the $(P2)$ possession rule:

$$(D7)\ B \ni F(N_a, c_i)$$

From statements $(D6)$ and $(D7)$ using $(P2)$ postulate:

$$(D8)\ B \ni F(F(N_a, c_i), K_i)$$

$$(A6)\ B \models \sharp(N_a)$$

From $(D7)$ using $(A6)$ and $(F1)$ we get:

$$(D9)\ B \models \sharp(N_a, c_i)$$

From statement $(D9)$ using $(F1)$ freshness rule:

$$(D10)\ B \models \sharp(F(N_a, c_i))$$

From $(D10)$ using $(F1)$ again:

$$(D11)\ B \models \sharp(F(N_a, c_i), K_i)$$

From $(D11)$ statement using $(F1)$ rule once again:

$$(D12)\ B \models \sharp(F(F(N_a, c_i), K_i))$$

From $(D8)$ and $(D12)$ statements using $(F10)$ we get:

$$(D13)\ B \models \sharp(H(F(F(N_a, c_i), K_i)))$$

From message $(M2)$ using $(T2)$ postulate:

$$(D14)\ B \triangleleft \star H(\star F(\star F(N_a, \star c_i), \star K_i))$$

From statements $(D8)$, $(D12)$ and $(D14)$ using the $(A5)$ assumption and the $(I3b)$ postulate with which the GNY logics were extended above (having $X = F(N_a, c_i), S = K_i$ substituted) we get:

$$(D15)\ B \models A \mid\sim F(F(N_a, c_i), K_i)$$

From $(D12)$ and $(D15)$ statements and the $(I6)$ rule:

$$(D16)\ B \models A \ni F\left(F\left(N_a, c_i\right), K_i\right)$$

From $(D2)$ statement using assumption $(A9)$, postulate $(P2)$ and the rationality rule:

$$(D17)\ B \models A \ni F\left(N_a, c_i\right)$$

From statements $(D16)$ and $(D17)$ using the $(P5)$ postulate and the rationality rule:

$$(D18)\ B \models A \ni K_i$$

Notice that statement $(D18)$ is the same as our $(C1)$ aim, which means that the tag is authenticated by the back-end indeed.

After receiving the 3^{rd} message:

$$(M3)\ A \triangleleft \star F\left(\star N_b, \star H\left(\star K_i'\right)\right), \star H\left(\star F\left(\star F\left(\star W_k, \star N_b\right), \star K_i'\right)\right), \star W_k$$

Using the $(T2)$ being told rule:

$$(D19)\ A \triangleleft \star F\left(\star N_b, \star H\left(\star K_i'\right)\right)$$

$$(D20)\ A \triangleleft \star H\left(\star F\left(\star F\left(\star W_k, \star N_b\right), \star K_i'\right)\right)$$

$$(D21)\ A \triangleleft \star W_k$$

$$(A8)\ A \ni K_0$$

From assumption $(A8)$ using the $(P4)$ postulate c_i times in a row we get:

$$(D22)\ A \ni H^{c_i}(K_0), \text{azaz } A \ni K_i$$

From statement $(D22)$ using $(P4)$ again:

$$(D23)\ A \ni H\left(K_i\right)$$

From $(D19)$ and $(D23)$ using the $(P5)$ postulate:

$$(D24)\ A \ni N_b$$

From $(D21)$ using $(T1)$:

$$(D25)\ A \triangleleft W_k$$

From $(D25)$ using the $(P1)$ postulate:

$$(D26)\ A \ni W_k$$

From statements $(D24)$ and $(D26)$ using $(P2)$ we get:

$$(D27)\ A \ni F\left(W_k, N_b\right)$$

From statements $(D22)$ and $(D24)$ using the $(P2)$ postulate again:

$$(D28)\ A \ni F\left(F\left(W_k, N_b\right), K_i\right)$$

$$(A12)\ A \models B\ \mathrm{Z}\Rightarrow \sharp(N_b)$$

$$(A13)\ A \models B \models \sharp(N_b)$$

From assumptions $(A12)$ and $(A13)$ using the $(J1)$ jurisdiction rule:

$$(D29)\ A \models \sharp(N_b)$$

From $(D29)$ using $(F1)$:

$$(D30)\ A \models \sharp(W_k, N_b)$$

From $(D30)$ using $(F1)$ again:

$$(D31)\ A \models \sharp(F(W_k, N_b))$$

From $(D31)$ using $(F1)$ once again:

$$(D32)\ A \models \sharp(F(W_k, N_b), K_i)$$

From statement $(D32)$ using $(F1)$:

$$(D33)\ A \models \sharp(F(F(W_k, N_b), K_i))$$

From statements $(D20)$, $(D28)$ and $(D33)$, the $(A10)$ assumption and the $(I3b)$ postulate from the extended GNY logics (having $X = F(W_k, N_b), S = K_i$ substituted) we get:

$$(D34)\ A \models B \mid\sim F(F(W_k, N_b), K_i)$$

From statements $(D33)$ and $(D34)$ using $(I6)$:

$$(D35)\ A \models B \ni F(F(W_k, N_b), K_i)$$

From assumptions $(A3)$ and $(A4)$ using the $(P2)$ postulate:

$$(D36)\ A \models B \ni F(W_k, N_b)$$

And finally from statements $(D35)$ and $(D36)$ using $(P5)$ we get:

$$(D37)\ A \models B \ni K_i$$

Notice that the $(D37)$ statement is the same as our $(C2)$ aim, which means the back-end is authenticated by the tag. Since both of our aims, $(C1)$ and $(C2)$ are fulfilled, it is proved that the improved protocol provides mutual authentication between the tags and the back-end.

7 Future Work and Conclusions

We have seen that authentication is really important in RFID systems, because of the public radio communication. Passive tags have a very limited CPU capacity, thus it has become obvious that lightweight protocols are the only suitable solutions for this problem. A number of RFID authentication protocols exist, but every one of them had their security issues. We chose an existing protocol that needed some improvement to prevent against well known reply attacks, but was still simple enough to suit passive tags. We analyzed the secureness and then formally verified the correctness of our protocol.

The proposed protocol still has weaknesses that need further studies:

Desynchronization attacks when the 3^{rd} message is blocked by an adversary, the back-end advances in the hash-chain but the tag does not, which leads to different secret values at the two parties. Even if a 4^{th} confirmation message is inserted to let the back-end know that the tag has

successfully run the protocol, it can still be blocked, etc. This is known as The Byzantine Generals Problem [7].

Problems with database error when the back-end's database is corrupted for some reason and it needs to be rolled back to a previous state, tags that were authenticated since the last save will be desynchronized and this will render them completely unusable. Further studies are needed to find a way to let the parties get resynchronized.

The protocol presented in this paper still needs improvements, thus our future work is to eliminate these vulnerabilities. After addressing these issues this work may provide the RFID industry a good starting point in making their products more secure.

Acknowledgements This paper was made in the frame of Mobile Innovation Centre's integrated project Nr. 1.1., supported by the National Office for Research and Technology (Mobile 01/2004 contract).

References

1. Luo, Z., Chan, T., Li, J. S.: A Lightweight Mutual Authentication Protocol for RFID Networks. Proc. of the 2005 IEEE International Conference on e-Business Engineering (ICEBE'05), IEEE (2005)
2. Ohbuko, M., Suzuki, K., Kinoshita, S.: Cryptographic Approach to "Privacy-Friendly" Tag. RFID Privacy Workshop@MIT (2003)
3. Weis, S. A., Sarma, S. E., Rivest, R. L., Engels, D. W.: Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems. Security in Pervasive Computing, LNCS 2802 (2004) 201–212
4. Ohbuko, M., Suzuki, K., Kinoshita, S.: Hash-Chain Based Forward-Secure Privacy Protection Scheme for Low-Cost RFID. Proceedings of the 2004 Symposium on Cryptography and Information Security(SCIS2004), Vol. 1 (Jan. 2004) 719–724
5. Gong, L., Needham, R., Yahalom, R.: Reasoning about belief in cryptographics protocols. Proceedings 1990 IEEE Symposium on Research in Security and Privacy, IEEE Computer Society Press (1990) 234–248
6. Rogaway, P. and Shrimpton, T.: Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In Bimal Roy and Willi Meier, editors, Lecture Notes in Computer Science. Springer-Verlag Heidelberg, (2004) 371–388
7. Lamport, L., Shostak, R., Pease, M.: The Byzantine Generals Problem. ACM Transactions on Programming Languages and Systems, Vol. 4., No. 3 (July 1982) 382–401