Improving the Efficiency of RFID Authentication with Pre-Computation

Kaleb Lee **Juan Manuel González Nieto** **Colin Boyd**

Faculty of Science and Technology
Queensland University of Technology
kaleb.lee@student.qut.edu.au, (j.gonzaleznieto, c.boyd)@qut.edu.au

Abstract

Security of RFID authentication protocols has received considerable interest recently. However, an important aspect of such protocols that has not received as much attention is the efficiency of their communication. In this paper we investigate the efficiency benefits of pre-computation for time-constrained applications in small to medium RFID networks. We also outline a protocol utilizing this mechanism in order to demonstrate the benefits and drawbacks of using this approach. The proposed protocol shows promising results as it is able to offer the security of untraceable protocols whilst only requiring the time comparable to that of more efficient but traceable protocols.

1 Introduction

RFID (Radio Frequency Identification) is a wireless identification method that utilizes the reception of electromagnetic or electrostatic radio waves (Shepard 2005). An RFID system typically consists of three components: RFID tags, RFID readers and a back-end database. Although most RFID networks make use of a backend database, there are some applications of RFID that do not require, or may even preclude, a connection with a backend database (Ahamed et al. 2008); however, in this paper we assume a database to be used. The most common, and widespread type of RFID networks are low-cost passive RFID networks. Such tags are distinguished by their dependency on a reader for power as well as their lack of computational power (Rao S. 2007). Passive RFID has been gaining popularity due to its relatively low cost; passive systems are therefore ideal for a wide range applications including object tracking, supply chain management to traffic tolling.

As the applications of such RFID networks continue to expand, the need for security and privacy also becomes more prominent. Many RFID authentication protocols have been proposed that aim to provide a higher level of security to low-cost RFID networks (Mikko Lehtonen et al. 2006); however, aside from security issues special attention must also be taken to consider the feasibility of these protocols. Although a large amount of work has been previously focussed on overcoming the computational limitations of both the tag and the reader/database (Zhang & Baciu 2008), (Toiruul et al. 2007), little effort has been concentrated in the *time efficiency* of RFID authentication protocols, a very practical constraint.

In this paper we will explore the feasibility of the current types of protocols in terms of their communications efficiency as well as investigate the use of pre-computation as a means to allow protocols to overcome this time barrier. In this paper we show that pre-computation is not only a viable method for improving the efficiency of RFID protocols, but also a practical means for achieving higher levels of security in time-constrained applications. Using this method, we are able to greatly reduce the time required during authentication while at the same time remain with very reasonable computational and memory requirements. However, we also note that the use of pre-computation is most suited for small to medium RFID networks.

2 Motivation and Previous Work

Efficiency and privacy are probably the two most important factors when designing RFID authentication protocols. Protocols have to be efficient in a scalable manner to adapt from small to large networks. RFID tags should also be untraceable in order to protect privacy of users. At the same time protocols need to be computationally efficient for both the database/reader and tag; most research emphasises efficiency of tags as they are much more computationally limited when compared to readers and databases. However, little emphasis has been put on protocols for time constrained applications, a major type of application for RFID.

For instance, consider a scenario where an organization makes use of RFID smart cards for access control. Everyone inside the premises is required to use a smart card to open doors and access various equipment and for the organization to log personnel movement, for example by placing readers on doors it is possible to log access. Another possible scenario might be the tracking of inventory, where tags on items are read by the reader as they pass though a gate or door. Evidently such scenarios require authentication to be completed within a limited timeframe, namely the time which the tag takes to pass though the door/gate. However, under such circumstances it is possible for adversaries to easily track the flow of equipment or inventory by simply placing a RFID reader to eavesdrop communication. This can potentially lead to privacy breaches or financial loss.

Current RFID authentication protocols can be separated into two main categories, stateful and stateless, as outlined by Alomair et al. (Alomair & Poovendran 2010). The two types are defined by their use of states, or more practically the management of their (secret) identifiers. Whereas tags using stateful protocols update their identifiers after every successful authentication session with the help of the reader/database, stateless protocols make use of static constant identifiers but utilize other measures, most commonly pseudorandom number generators, to provide security.

Stateless protocols such as the Randomized Hash-Lock protocol (Weis 2003), and tree-based protocols first proposed by Molnar et. al. (Molnar & Wagner 2004), tend to display stronger security properties compared to *stateful protocols*. Most importantly, nearly all stateless protocols are untraceable. This is because they typically make use of a pseudorandom number generator to randomize their responses, so that an eavesdropper cannot correlate different tag responses. However, this additional security property comes at the expense of scalability and efficiency. As the reader/database now has to perform exhaustive search of all existing identifiers stored in order to authenticate just one tag, it is simply not feasible for networks with a large amount of tags to run the protocol efficiently. Attempts to minimize the computational load on the reader/database using a tree-based hierarchy have been proposed by Molnar et al. (Molnar & Wagner 2004). Although this approach was able to greatly reduce the computational load on the reader/database it also introduced some practical issues as the number of messages that must be exchanged before a tag can be authenticated increases accordingly to the number of tags in the system.

There is a vast number of proposed stateful protocols, particularly relative to the number of stateless protocols, and their efficiency varies greatly. In this paper we will be discussing one of the most efficient types of stateful protocols known as *indexed stateful protocols*; examples include EMAP (Peris-Lopez, Hernandez-Castro, Estevez-Tapiador & Ribagorda 2006), LMAP (Peris-lopez, Hern, Tapiador & Ribagorda 2006), LRMAP (Ha et al. 2007), and a protocol of Dimitriou (Dimitriou 2005). These protocols are both scalable and efficient. They utilize an invertible function to encrypt the tag's identifier and their security is dependent on the function chosen. Such an approach does not require exhaustive search of all existing identifiers at the reader/database, thus they are typically very efficient when compared to stateless protocols and consequently scalable. However, because they do not require responses from tags to be randomized they can therefore be traced by both passive and active adversaries. Identifiers are updated only after each successful authentication, in order to provide partial protection from the tag being traced, but consequently they are susceptible to de-synchronization. De-synchronization occurs when the identifier, or state, stored on a tag does not match the identifier stored on the reader/database, causing authentication to fail.

As reflected by the protocols, scalability, traceability and efficiency has been the main focus of most research. Little attention has been given to the communication efficiency, particularly issues such as the amount of messages required to be exchanged and the time required for each authentication. As there are many time-constrained RFID applications, as mentioned earlier, this can greatly affect the feasibility of protocols. On a side note, it should be recognized that the problem of communication efficiency is much more prominent in stateless protocols than indexed stateful protocols.

Although there are a number of previous works such as these by Li et al. (Li et al. 2010) and Poulopoulos et al. (Poulopoulos et al. 2009) which focus on offering efficient authentication, they typically do not offer protection against tracing. In this paper, we show that the use of pre-computation can not only greatly improve the communication efficiency of RFID protocols, but also allows tags to be untraceable to a certain extent. More recent work by Alomair et al. (Alomair et al. 2010) proposed a protocol that allowed constant-time identification using pre-computation. However, as their proposed protocol makes use of an internal counter whose value is only known by the tag, it requires very large amounts data to be pre-computed before a system can be initialized. We believe our approach is more flexible as it does not require the use of very large pre-computed database and that our approach allows authentication on an on-demand basis ultimately requiring less overall computation.

In summary, the novel contributions of this paper are:

- a method of employing pre-computation in the design of time-efficient RFID protocols;
- analysis of the security properties available using our methods which shows that untraceability can be achieved;
- design of a specific protocol utilizing the method and a comparitive analysis of its efficiency and security.

3 Phases and Time of Authentication Protocols

Typically an RFID protocol involves the exchange of three messages between the reader and tag[1]. As shown in Figure 1, in the first message, a is a broadcast query sent from the reader to initiate a session with the tag. The tag then responds with message b, typically with information required for authentication. The reader finally replies with the third message c both as an acknowledgement to the authentication request as well as other data, such as the new tag secret, required to finish the session.

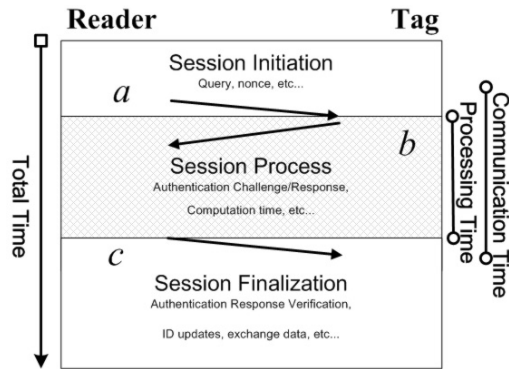

Figure 1: Communication of Typical Authentication Protocols

An authentication session is completed in three phases, *Session Initiation*, *Session Process* and *Session Finalization.*

[1]Most commonly stateless protocols have three message exchanges. Although there are protocols with less than three messages, as well as protocols that require more than three (most notably tree-based protocols) these can be easily adapted into our notation simply by treating the last message as the third message and all messages between the first and third combined to be the second message.

1. *Session initiation* is the phase in which the reader generates any data required for its initiation query, a. This phase commonly involves generating and broadcasting a random nonce. The first phase ends when a tag receives the query, thus entering the next phase.

2. The second phase is known as the *Session Process* phase, during which the tag generates and replies with b to the point where the reader receives b and has completed all operations for c to be transmitted. From the tag's point of view, this phase typically involves encrypting its secret identifier using the received (as well as possibly its own generated) nonce before sending it as b. On receiving b, the reader does what is required to decrypt or verify b, commonly by searching database entries or performing various cryptographic operations. The second phase ends when the reader finishes verifying the identity of the tag, and if required perform any computation necessary for the final message c.

3. The remaining phase, *Session Finalization*, consists of the final message c from the reader as well as any remaining computation for both the reader and tag.

In generalizing protocol sessions into the above phases we can now discuss time accordingly. As shown in Figure 1, we consider three different notions of time, *Communication Time*, *Processing Time*, and *Total time*.

- *Communication Time*, as the name suggests, is the time required for the communication of protocol messages and all processing in-between, i.e. from the start of the reader transmitting message a to the time the tag receives message c.

- *Processing Time* denotes the time from which the tag replies with b to the point before where the reader sends c. This is effectively the time during which the database and reader are to perform the (typically) most demanding operations, the *Session Process* phase.

- *Total time* is the total amount of time spent on the protocol session, i.e. the total time of the three phases combined.

Practically, at a minimum, it is necessary for the tag to be in the range of the reader for the duration of the *Communication Time*. Otherwise the session would not complete leading to a failed session. Therefore in this paper our main aim is to minimize *Communication Time* by offloading time-consuming computations in the *Session Process* phase.

4 Phases and Time of Pre-Computed Protocols

The aim of pre-computation is to offload most of the computation required for the *Session Process* during a protocol session in order to decrease the *Communication Time* of a protocol. Although it is possible for indexed stateful protocols to utilize pre-computation, stateless protocols would see the most improvement, so our example protocol is a stateless protocol.

The phases of pre-computation protocols follow very similarly to that of non-pre-computation protocols, as shown in figure 2. Pre-computed protocols introduces a *Pre-Computation* phase that has to be completed before a session is initiated. During this phase, the reader/database performs the most expensive computational operations that were originally performed during *Session Process* phase; in most cases it would be the hash operations, or cryptographic operations required for exhaustive search. Evidently, not all protocols can utilize pre-computation, only those where such offloading is possible.

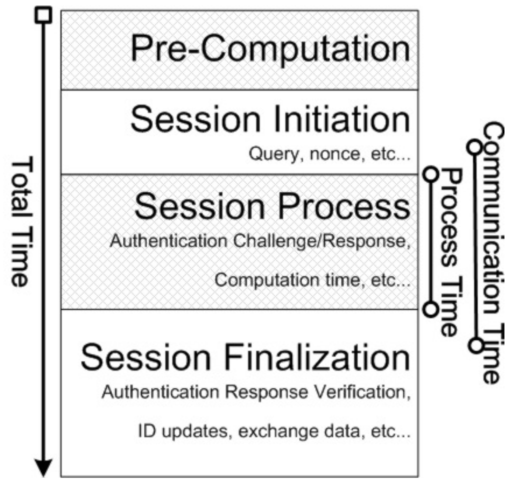

Figure 2: Communication of Pre-Computed Protocols

After most computation has been completed during the *Pre-Computation* phase, the resulting *Session Process* phase requires only minimal computation. In the most extreme examples, such as the protocol example given later in this paper, the reader is only required to perform a search of the pre-computed data, which requires only minimal time. Using such a method, we are able to reduce the *Process Time* dramatically and consequently also *Communication Time*, thus allowing parties to be authenticated within a smaller timeframe.

5 EP-UAP: Efficient Passively-Untraceable Authentication Protocol

In this section, we present a simple example protocol showing how pre-computation can be implemented. This protocol is based on the idea of Randomized Hash-Lock(Weis 2003), and can be considered the pre-computed implementation of the Randomized Hash-Lock protocol. We will be analyzing the efficiency and security of this protocol, and at the same time comparing it to both indexed stateful and stateless protocols, in later sections.

5.1 Notation

The following notation is used to describe the EP-UAP protocol:

H	a one-way hash function
T	RFID Tag
R	RFID Reader
$\|\|$	concatenation operation
ID_{1T}	unique tag identification code, stored in T
ID_{2T}	unique tag identification code, stored in T
R_T	a random nonce, generated by T
R_R	a random nonce, generated by R
ID_{1R}	unique tag identification code, stored in R
ID_{2R}	unique tag identification code, stored in R
m_{TR}	authentication challenge
m_{RT}	authentication challenge response
c_T	authentication challenge check

5.2 EP-UAP Pre-Computation Process

Pre-computation of EP-UAP consists of three steps:

1. Numerous random numbers, $R_{R1}, R_{R4}, R_{R3}, \dots$, are generated.
2. $H(ID_{1R}||R_{Rn})$ are calculated for all existing ID_{1R} using a generated R_{Rn}.
3. Using $H(ID_{1R}||R_{Rn})$ as an index for R_{Rn} all results are stored in the database, after which we say that R_{Rn} is (pre-)computed.

The above process can be repeated until all possible values of R_R are calculated, if resource on the database allows, or can be repeated until a predetermined number of R_R values are pre-computed. For maximum efficiency this process is repeated until all possible values of R_R, however as this can consume a large amount of storage it is only recommended for reasonably small networks.

5.3 EP-UAP Authentication Process

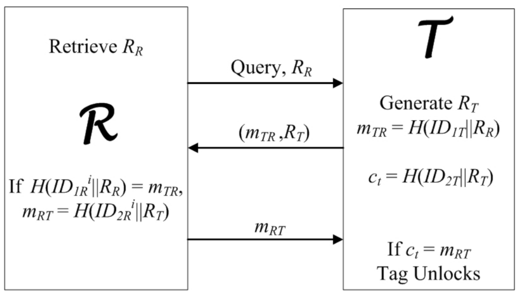

Figure 3: EP-UAP Authentication Process

Authentication between a reader and tag consists of the following three message exchanges, as shown in figure 3:

- In the initial step the reader broadcasts an authenticated R_R along with a communication query.
- After a query and R_R has been received, T generates a fresh random nonce R_T which, in turn, is sent to R along with an authentication challenge m_{TR}, where $m_{TR} = H(ID_{1T}||R_R)$. T would continue to compute the challenge check message c_T, where $c_T = H(ID_{2T}||R_T)$, whilst waiting for R's response.
- Once R receives message (m_{TR}, R_T), it searches for a ID^i_{1R} where $H(ID^i_{1R}||R_R) = m_{TR}$ for the computed R_R.
  - If there exists ID^i_{1R}, where $H(ID^i_{1R}||R_R) = m_{TR}$ the tag would be considered to be authentic and the communicating tag would be identified as ID^i_{1R}. Subsequently the associated ID^i_{2R} would be used to construct the challenge response message m_{RT}, where $m_{RT} = H(ID^i_{2R}||R_T)$. m_{RT} is then sent back to T as a response to m_{TR}. Once T receives m_{RT} it verifies whether $m_{RT} = c_T$.
    - ∗ If $m_{RT} = c_T$, R is deemed as authentic and T would unlock for further communication.
    - ∗ If $m_{RT} \neq c_T$, R is considered to be hostile, consequently T will terminate the current authentication session be ceasing further communication with R until it receives a new query.
  - If there does not exists ID^i_{1R}, where $H(ID^i_{1R}||R_R) = m_{TR}$, T is considered to be hostile, consequently R will terminate the current authentication session, ceasing further communication with T.

5.4 EP-UAP Post-Authentication Process

The use of the post-authentication process is dependent on the security and resource requirements of a database. Three likely scenarios are described below.

- In this scenario resources of the database is limited, and maximum security is not required. Under the given conditions, the number of pre-computed R_R is most likely capped to a predetermined value in order to restrict the use of resources on the database. After each successful authentication between tag and reader/database, all values of $H(ID_{1R},R_{Rn})$ for the given R_R are deleted from the database to allow for a fresh R_R to be pre-computed.
- In this scenario, resource is limited but maximum security is preferred. Under the given conditions, the number of pre-computed R_R is most likely capped to a predetermined value in order to restrict the use of resources on the database. A pool of all possible values of R_R is also created and stored in the database. After each successful authentication between tag and reader/database, all values of $H(ID_{1R},R_{Rn})$ for the given R_R are deleted from the database to allow for a fresh R_R to be pre-computed, the value of the authenticated R_R would be removed from the pool of available R_R values. A fresh R_R is randomly selected from the pool. If there are no available R_R, the pool will be recreated with all possible values of R_R and the previous process would be repeated.
- In this scenario, resource is not and maximum security is preferred. Under the given conditions, the number of pre-computed R_R is not capped. A pool of all possible values of R_R is also created and stored in the database. After each successful authentication between tag and reader/database, all values of $H(ID_{1R},R_{Rn})$ for the given R_R are deleted from the database to allow for a fresh R_R to be pre-computed, the value of the authenticated R_R would be removed from the pool of available R_R values. A fresh R_R is randomly selected from the pool. If there is no available R_R in the pool, the pool will be recreated with all possible values of R_R and the process would be repeated. The first time this occurs this database would have stored all possible values of $H(ID_{1R},R_{Rn})$; thus no additional processing is required for future authentication sessions until new tags are added in which all possible of $H(ID_{1R},R_{Rn})$ would be created for that *(ID)*.

6 Analysis of Protocols

In this section we compare the EP-UAP with three other protocols, a indexed stateful protocol, LRMAP

(Ha et al. 2007), and two stateless protocols, randomized hash-lock (Weis 2003) and a tree-based protocol proposed by Molnar et al. (Molnar & Wagner 2004). As most indexed stateful protocols are of similar efficiency and scalability, any indexed stateful protocol should be able to provide a general guideline regarding their efficiency. Stateless protocols, however, are very different. We have chosen two of the earliest protocols of each class of protocols which set very different efficiency standards.

The randomized hash-lock is one of the first stateless protocols proposed and is of linear time, whereas the tree-based protocol proposed by Molnar et al., one of the first protocols proposed in order to improve the efficiency of linear-time stateless protocols, operates under logarithmic time. The efficiency of both classes of stateless protocols have remained largely within the same range since their introduction (Yousuf & Potdar 2008), thus they should be sufficient as a baseline comparison. Overall the three protocols together should provide a good baseline comparison on the efficiency of common RFID protocols.

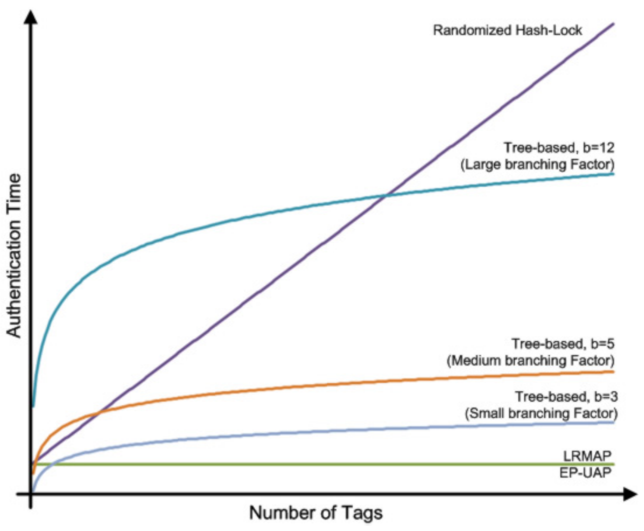

Figure 4: Authentication Time vs Number of Tags

6.1 Efficiency Analysis

First we discuss the efficiency of pre-computed protocols using EP-UAP as an example. We focus only on the *Session Process* phase of the protocols, since if we assume that all protocols are used for the same application, the efficiency of the other phases would mostly remain constant. In this section, we take into account the worst case scenario of each protocol. The following notation is used throughout the remainder of this section.

N:	The total number of tags
H:	Time required to perform one cryptographic operation such as hashing
T:	Time required to transmit one message[2]
b:	Branching factor of a M-ary tree-based protocol

The time required to complete the *Session Process* phase is given by the sum of the total time of computations required on the reader/database and tag as well as the time for messages to be exchanged. Most messages exchanged are in the form of a challenge and response which is considered to be two messages, one from the challenger (typically a tag), and one from the respondent (typically the reader/database). Although practically there would be other factors involved, such as communication between reader and database and the time required to perform searches, these operations are typically not specified by most protocols; hence we assume that these times are constant for all protocols thus omitted from the comparison. A summary of the comparison is given in table 1; a graph of the relationships between the number of tags and time required is also shown in Figure 4.

6.1.1 LRMAP

In a typical scenario, the LRMAP protocol requires a total of 3 hash operations to be performed by the tag and another 3 to be performed by the reader/database, however in the case of de-synchronization the reader/database is required to perform an exhaustive search on all identifiers stored increasing the amount of hash operations required by N resulting in a total of $3 + N$ operations. Note that LRMAP could be considered as a 'hybrid' protocol, where during normal use it is an indexed stateful protocol, but functions similarly to stateless protocols during de-synchronization. Hence here we focus on normal usage scenarios where the tag is assumed to be synchronized. Using LRMAP as an example, we can safely conclude that in typical situations an indexed stateful protocol can offer constant authentication performance, i.e. the time required for authentication is independent of the number of tags or other similar factors.

As expected, regardless of the situation only two messages are required to be exchanged under the LRMAP protocol during *Session Process* phase — one as an authentication challenge from the tag and the other a challenge response from the reader/database. As with nearly all indexed stateful protocols, LRMAP can be used on small to large networks assuming that de-synchronization is rare.

6.1.2 Randomized Hash-Lock

Unlike indexed stateful protocols, stateless protocol do not require identifiers to be updated, neglecting the possibility of de-synchronization. As such the number of hash operations required to be performed by the reader/database under the randomized hash-lock protocol remains at a constant N, with only one hash operation required by the tag. However, since the load of the reader/database increases relative to the total number of tags in the network, the time required can potentially become unrealistically high if there is a large number of tags, hence this type of protocol is feasible on networks with a small number of tags.

Under all situations, two messages are required to be exchanged in the randomized protocol during *Session Process* phase — one as an authentication challenge from the tag and the other a challenge response from the reader/database.

As the randomized hash-lock protocol requires up to N (average of $\frac{N}{2}$) hash operations per authentication, putting a large huge amount of computational load on the reader/database, it is feasible for use only on small networks.

6.1.3 Tree-based Protocols

Tree-based protocols make use of an M-ary tree in order to optimize the amount of computation load on the database. However as we will outline, this

Protocol	Time Required	Network Size
LRMAP	$(6\ [+\ N])H + 2T$	Small - Large
Randomized H-L	$(N)H + 2T$	Small
Tree-based	$(b\ log_b\ N)H + (log_b\ N)T$	Small - Medium
EP-UAP	$3H + 2T$	Small - Medium

Table 1: Efficiency Comparison

approach is not without its drawbacks. Using a typically top-down tree walking approach, there would be a total of $\log_b N$ layers in the tree with each branch having b identifiers. As such, under exhaustive search the total number of hashes required to be performed by the reader/database would be $b \log_b N$. The number of operations required to be performed by the tag would consequently be the same.

Unlike the randomized hash-lock protocol, however, the tree-based protocol authenticates by walking the tree layer by layer, consequently the total number of messages required has also increased to $\log_b N$ which is the total number of layers in the tree. This is a major feasibility concern if a network has a large number of tags but with a small branching factor, as it would require a unrealistic number of messages per authentication. On the other hand, if a network is to have a large branching factor the protocol would require more computation in exchange for less messages per session.

Due to their tree walking nature, tree-based protocols can be used on much larger networks compared to linear-time stateless protocols. The decrease in computational load to b number of hash operations per step is a dramatic decrease when compared to N, keeping the computational load on the reader/database much more manageable.

6.1.4 EP-UAP

As the hash operations required to be performed have already been pre-computed, only the verification of authenticity remains. As such the pre-computed protocol requires only 1 hash operation to be computed by the reader/database, whereas the tag would only have to compute 2, one as an authentication challenge and the other for verification. As expected, *Session Process* phase of EP-UAP is completed after two message exchanges.

This is a dramatic decrease in all areas when compared to the other protocols. Most importantly, the EP-UAP protocol, assuming that the amount of tags in the system remains constant, offers constant-time authentication i.e. the amount of time required per authentication session remains constant regardless of the number of tags, a feat achievable only by indexed stateful protocols.

Unlike other protocols, offloading the computational load during authentication by pre-computing the required operations places a new restriction on these protocols, namely storage. Since pre-computed information must be stored in the database, it might not be feasible if there is a large number of tags on the network. Thus the deployment of pre-computed protocols is limited only in small to medium networks. Nevertheless, we believe this is a worthwhile investment given the vast improvement in efficiency. It should also be noted that this pre-computed 'database' does not have to be stored or computed at a central location; in many situations it would be more beneficial to have multiple independent systems where needed. For example, in smaller networks it is very possible for the computation and data storage to be managed by the reader.

6.2 Security Comparison

Security properties of protocols are more difficult to compare due to the vast amount of factors. Indeed, it should be noted that the EP-UAP, the randomized hash-lock, the tree-based protocol and some indexed stateful protocols such as EMAP and LMAP, are still relatively immature and have been shown to be insecure in various aspects (Avoine et al. 2006), (Wang et al. 2007), (Lu et al. 2009), (Li & Deng 2007). Therefore it would be of limited interest to perform in-depth analysis and/or comparisons at this point. Our purpose is rather to show the basic security limitations of each type of protocol and therefore we have pinpointed one particular important property: traceability.

For simplicity, we regard traceability as the ability for an adversary to distinguish, or identify, a tag based only on its responses[3]. We consider two levels of security in regard to traceability: passive and active. Note that it is possible for protocols to be neither passive nor actively secure.

To clarify each of the levels we first introduce two types of adversaries, passive adversaries and active adversaries. Passive adversaries have the ability to eavesdrop and block communication between a reader and tag but do not have the power to interact with sessions in any way. Active adversaries not only have the ability to eavesdrop sessions but also insert, modify and block messages between the reader and tag. The level of security of a given protocol is determined by the type of adversary it is secure against. If it is secure against passive adversaries then it is *passive secure*; similarly if it is secure against active adversaries then it is considered *active secure.* If a protocol is active secure it is also passive secure, since an active adversary has all the power that a passive adversary possesses (Ding Zhen-hua 2008, Jung-Chun Kao 2006). A summary of the comparison is shown in Table 2.

Protocol	Passive	Active
Indexed Stateful	N	N
Stateless	Y	Y/N
Pre-Computed	Y	N

Table 2: Traceability Comparison

As indexed stateful protocols cannot update states in the absence of an authentic reader and do not have the ability of randomizing their responses, they are both active and passively traceable. By eavesdropping more than one consecutive unsuccessful authentication attempt, it is possible for an adversary to associate the two sessions simply by matching the response sent by the tag as they should be the same.

Stateless protocols, on the other hand, have the ability to randomize their responses using their embedded pseudorandom number generator, so they

[3]Note that there are many different factors that can cause tags to be traceable but are not considered in this paper; these include side-channel information (for instance the time required for authentication or whether authentication was successful), or extraction of secret information.

have the ability to be either passively untraceable or both passive and actively untraceable depending on protocol specification. It is possible for stateless protocols to be actively untraceable if a protocol specifies that authentication challenge is to be randomized independent of the reader's query, else if the challenge is only randomized based on the query from a reader, it can only be passively untraceable. (Note that the particular protocol (LRMAP) we used for comparison is not vulnerable to this as it acts like a stateless protocol after one unsuccessful authentication attempt, but this is an exception rather than norm.)

Aside from traceability under ideal scenarios, tree-based protocols are more susceptible to tracing if some tags have been compromised. Since tree-based protocols authenticate tags by walking from the top of the tree, down through each branch to a specific leaf, if the identifiers of the higher branches are compromised it is possible to distinguish whether or not an uncompromised tag belongs to a specific branch. This attack is studied in further detail by Avoine et al. (Avoine et al. 2006), who also concluded that the impact of such compromises on the logarithmic-time stateless protocol can be minimized by increasing the branching factor of the tree. Note that this attack only affects protocols where identifier are arranged in such manner; other protocols such as stateful protocols and linear-time stateless protocols are not vulnerable to this threat, as all identifiers stored on the tag are unique among all tags in the network.

Unfortunately since pre-computed protocols depend on the reader/database to randomize their response, it is possible for an adversary to impersonate a reader and actively query tags using a previously used nonce. An active adversary can simply reply a previously used random nonce. Nevertheless, it is possible for pre-computed protocols to be actively secure by introducing a nonce generated by the tag and for the database to pre-compute the combination of both. However this would increase the amount of computation required beyond the number of tags in the system, a tradeoff that is difficult to justify.

7 Feasibility of EP-UAP

In this section we discuss the practical feasibility of pre-computation using the EP-UAP protocol as an example. The performance of the protocols will be compared by approximating the time required for one authentication in a network with the same amount of tags. The network used in the comparison is a small-to-medium network with 406900 tags. Tag identifiers are to be 256 bits in length — most implemented tags today only use around 64-bit to 128-bit identifiers and it is safe to assume that this number is only going to increase. It is also assumed that the database can perform around 100 MB of hash operations per second. This approximation is based on a dual-core processor operating at 1.83 GHz released in 2007; more modern processors commonly have more than twice the amount of cores operating at more than twice the frequency and quite possibly utilizing more efficient architectures, hence this estimation should prove useful only as a baseline. We also estimate the time required to transmit one message to be approximately 20ms. The results in this section are computed using the formulas from Table 1 under the above estimations.

As the efficiency of tree-based protocols depends on the branching factor of the tree, b, we give three approximations of tree-based protocols: first with minimum branching factor, from binary tree where $b = 2$, a reasonable branching factor, where $b = 2^5$, and finally a large branching factor (relative to the number of tags), where $b = 2^{12}$.

The results are given in Table 3. As show in the table, indexed stateful protocols such as the LRMAP protocol are expectedly the most efficient requiring a minimum of only 40 ms, whereas the randomized hash-lock protocol requires the longest time. The most interesting of the figures are perhaps for the tree-based protocols. In order to optimize the tree-based protocols, one must find a suitable balance between the amount of computational load on the reader/database, by increasing the branching factor thus increasing the security, and the number of messages required for authentication, a very important factor into determining the time required to authenticate one tag. The EP-UAP protocol is also one of the most efficient protocols requiring only a minimum of 40ms for authentication.

Aside from the authentication, practically we also have to take into account the total time required for computation: process time. Whereas with the other authentication protocols the total time required is around the same range as their process time, this is not the case for pre- protocols. The notion of pre-computation is to minimize the time required for communication session by reallocating the time required for authentication into two periods of time, where most time consuming computations are to be processed before the session begins. Howver, rhe total of the two times remain unchanged. Another limitation of pre-computation protocols is the amount of storage required. Using the data from Table 3 where there are 406900 tags, the database is required to store at least 100 MB of data. This might not be a major problem as most modern systems typically have multiple gigabytes of RAM, but would nevertheless limit rate of authentication, i.e. the number of authentications within a period of time.

One possible issue that limits the feasibility of pre-computation protocols are applications where there could be a continuous high rate of authentication, resulting in the number of required authentications exceeding the number of computations the database is capable of. This issue can be partially eased by either increasing the computational power of the database or increasing the memory, this can be achieved as the systems can be independent of each other. This increase can also be temporary in applications where there are predicted periods of high demand, such as workers coming and leaving work. However, we emphasize that a large network is required for such events to occur, ones that we do not recommend for pre-computation.

8 Conclusion and Future Work

This paper investigates the use of pre-computation as a means to minimize the time required for authentication. By utilizing pre-computation we were able to construct the EP-UAP protocol to demonstrate the benefits as well as outline the drawbacks of pre-computation. We were able to show that by using pre-computation we are able to provide untraceability at a comparable level to stateless protocols whilst maintaining within the efficiency range of indexed stateful protocols, which do not provide any untraceability. However such protocols are only suited for small to medium networks due to possible storage constraints but nevertheless an improvement over some stateless protocols which are only suited for small networks.

As the EP-UAP protocol is designed as a proof-of-concept protocol, and still very immature, further

Protocol	Protocol Type	Computational Time (ms)	Transmission Time (ms)	Process Time (ms)	Total Time (ms)
LRMAP	Stateful *(Indexed)*	1.46×10^{-3}	40	40	40
Rand. Hash-Lock	Stateless *(Linear)*	1000	40	1040	1040
Rand. Hash-Lock (avg)	Stateless *(Linear)*	500	40	540	540
Tree-based ($b = 2^1$)	Stateless *(Log'mic)*	9.1×10^{-3}	380	380	380
Tree-based ($b = 2^5$)	Stateless *(Log'mic)*	2.9×10^{-2}	80	80	80
Tree-based ($b = 2^{12}$)	Stateless *(Log'mic)*	15	40	45	45
EP-UAP	Pre-Computed	4.8×10^{-4}	40	40	1040

Table 3: Comparison of Protocol Running Times

work is required in order to design a provably secure authentication protocol for RFID utilizing pre-computation. This would to allow for a more robust comparison between protocols as well as a more in-depth analysis of the possible advantages or drawbacks of pre-computation.

References

Ahamed, S.I. Hoque E., R. F., F., K. & T., N. (2008), 'Ya-srap: Yet Another Serverless RFID Authentication Protocol', *Intelligent Environments, 2008 IET 4th International Conference on* **NA**(NA), 1 – 8.

Alomair, B., Clark, A., Cuellar, J. & Poovendran, R. (2010), Scalable RFID Systems: a Privacy-Preserving Protocol with Constant-Time Identification, *in* 'the 40th Annual IEEE/IFIP International Conference on Dependable Systems and Networks – DSN'10', IEEE, IEEE Computer Society, Chicago, Illinois, USA.

Alomair, B. & Poovendran, R. (2010), 'Privacy versus Scalability in Radio Frequency Identification Systems', *Computer Communication, Elsevier* .

Avoine, G., Dysli, E. & Oechslin, P. (2006), Reducing Time Complexity in RFID Systems, *in* B. Preneel & S. Tavares, eds, 'Selected Areas in Cryptography', Vol. 3897 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 291–306.

Dimitriou, T. (2005), A Lightweight RFID Protocol to Protect Against Traceability and Cloning Attacks, *in* 'Security and Privacy for Emerging Areas in Communications Networks, 2005. SecureComm 2005. First International Conference on', pp. 59–66.

Ding Zhen-hua, Li Jin-tao, F. B. (2008), 'A Taxonomy Model of RFID Security Threats', *Communication Technology, 2008. ICCT 2008. 11th IEEE International Conference on 10-12 Nov. 2008* **NA**(NA), 765 – 768.

Ha, J., Ha, J., Moon, S. & Boyd, C. (2007), LRMAP: Lightweight and Resynchronous Mutual Authentication Protocol for RFID System, *in* 'ICUCT'06: Proceedings of the 1st international conference on Ubiquitous convergence technology', Springer-Verlag, Berlin, Heidelberg, pp. 80–89.

Jung-Chun Kao, Marculescu, R. (2006), 'Eavesdropping Minimization via Transmission Power Control in Ad-Hoc Wireless Networks', *Sensor and Ad Hoc Communications and Networks, 2006. SECON '06. 2006 3rd Annual IEEE Communications Society on 28-28 Sept. 2006* **2**(NA), 707–714.

Li, J., Wang, Y., Jiao, B. & Xu, Y. (2010), An Authentication Protocol for Secure and Efficient RFID Communication, *in* '2010 International Conference on Logistics Systems and Intelligent Management', Vol. 3, pp. 1648 –1651.

Li, T. & Deng, R. (2007), Vulnerability Analysis of EMAP-An Efficient RFID Mutual Authentication Protocol, *in* 'Availability, Reliability and Security, 2007. ARES 2007. The Second International Conference on', pp. 238 –245.

Lu, L., Han, J., Xiao, R. & Liu, Y. (2009), Action: Breaking the Privacy Barrier for RFID Systems, *in* 'INFOCOM 2009, IEEE', pp. 1953 –1961.

Mikko Lehtonen, T. S., Michahelles, F. & Fleisch, E. (2006), From Identification to Authentication - A Review of RFID Product Authentication Techniques, *in* 'Printed handout of Workshop on RFID Security - RFIDSec 06'.

Molnar, D. & Wagner, D. (2004), Privacy and Security in Library RFID: Issues, Practices, and Architectures, *in* 'CCS '04: Proceedings of the 11th ACM conference on Computer and communications security', ACM, New York, NY, USA, pp. 210–219.

Peris-lopez, P., Hern, J. C., Tapiador, J. M. E. & Ribagorda, A. (2006), Lmap: A Real Lightweight Mutual Authentication Protocol for Low-Cost RFID Tags, *in* 'In: Proc. of 2nd Workshop on RFID Security', Ecrypt, p. 06.

Peris-Lopez, P., Hernandez-Castro, J., Estevez-Tapiador, J. & Ribagorda, A. (2006), Emap: An Efficient Mutual-Authentication Protocol for Low-Cost RFID Tags, *in* R. Meersman, Z. Tari & P. Herrero, eds, 'On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops', Vol. 4277 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 352–361.

Poulopoulos, G., Markantonakis, K. & Mayes, K. (2009), A Secure and Efficient Mutual Authentication Protocol for Low-Cost RFID Systems, *in* '2009 International Conference on Availability, Reliability and Security. ARES '09', pp. 706 –711.

Rao S., Thanthry N., P. R. (2007), 'Rfid Security Threats to Consumers: Hype vs. Reality', *Security Technology, 2007 41st Annual IEEE International Carnahan Conference on 8-11 Oct. 2007* pp. 59 – 63.

Shepard, S. (2005), *RFID Radio Frequency identification*, The McGraw-Hill Companies, Inc.

Toiruul, B., Lee, K. O. & Kim, J. M. (2007), SLAP - a Secure but Light Authentication Protocol for RFID Based on Modular Exponentiation, *in* 'International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, 2007. UBICOMM '07.', pp. 29 –34.

Wang, W., Li, Y., Hu, L. & Lu, L. (2007), Storage-Awareness: RFID Private Authentication based on Sparse Tree, *in* 'Security, Privacy and Trust in Pervasive and Ubiquitous Computing, 2007. SECPerU 2007. Third International Workshop on', pp. 61 – 66.

Weis, S. A. (2003), Security and Privacy in Radio-Frequency Identification Devices, PhD thesis, MIT.

Yousuf, Y. & Potdar, V. (2008), A Survey of RFID Authentication Protocols, *in* 'Advanced Information Networking and Applications - Workshops, 2008. AINAW 2008. 22nd International Conference on', pp. 1346 –1350.

Zhang, X. & Baciu, G. (2008), Low Cost Minimal Mutual Authentication Protocol for RFID, *in* 'IEEE International Conference on Networking, Sensing and Control, 2008. ICNSC 2008.', pp. 620 –624.