”ALEXANDRU-IOAN CUZA” UNIVERSITY IAȘI

FACULTY OF COMPUTER SCIENCE

BACHELOR’S THESIS

Consequences of using temporary variables in RFID authentication

proposed by

Andrei-Alexandru Patrașcan

Session: june, 2025

Scientific coordinator

Prof. Dr. Țiplea Ferucio Laurențiu

”ALEXANDRU-IOAN CUZA” UNIVERSITY IAȘI

FACULTY OF COMPUTER SCIENCE

Consequences of using temporary variables in RFID authentication

Andrei-Alexandru Patrașcan

Session: june, 2025

Scientific coordinator

Prof. Dr. Țiplea Ferucio Laurențiu

Avizat,

Îndrumător lucrare de licență,

Prof. Dr. Țiplea Ferucio Laurențiu.

Data: ..... Semnătura: .....

Declarație privind originalitatea conținutului lucrării de licență

Subsemnatul **Patrașcan Andrei-Alexandru** domiciliat în **România, jud. Bacău, com. Filipești, sat Hârlești, str. Zorilor, nr. 180**, născut la data de **22 martie 1999**, identificat prin CNP **1990322046190**, absolvent al Facultății de informatică, **Facultatea de informatică** specializarea **informatică**, promoția 2025, declar pe propria răspundere cunoscând consecințele falsului în declarații în sensul art. 326 din Noul Cod Penal și dispozițiile Legii Educației Naționale nr. 1/2011 art. 143 al. 4 și 5 referitoare la plagiat, că lucrarea de licență cu titlul **Consequences of using temporary variables in RFID authentication** elaborată sub îndrumarea domnului **Prof. Dr. Țiplea Ferucio Laurențiu**, pe care urmează să o susțin în fața comisiei este originală, îmi aparține și îmi asum conținutul său în întregime.

De asemenea, declar că sunt de acord ca lucrarea mea de licență să fie verificată prin orice modalitate legală pentru confirmarea originalității, consimțind inclusiv la introducerea conținutului ei într-o bază de date în acest scop.

Am luat la cunoștință despre faptul că este interzisă comercializarea de lucrări științifice în vederea facilitării falsificării de către cumpărător a calității de autor al unei lucrări de licență, de diplomă sau de disertație și în acest sens, declar pe proprie răspundere că lucrarea de față nu a fost copiată ci reprezintă rodul cercetării pe care am întreprins-o.

Data: ..... Semnătura: .....

Declarație de consimțământ

Prin prezenta declar că sunt de acord ca lucrarea de licență cu titlul **Consequences of using temporary variables in RFID authentication**, codul sursă al programelor și celelalte conținuturi (grafice, multimedia, date de test, etc.) care însoțesc această lucrare să fie utilizate în cadrul Facultății de informatică.

De asemenea, sunt de acord ca Facultatea de informatică de la Universitatea "Alexandru-Ioan Cuza" din Iași, să utilizeze, modifice, reproducă și să distribuie în scopuri necomerciale programele-calculator, format executabil și sursă, realizate de mine în cadrul prezentei lucrări de licență.

Absolvent **Andrei-Alexandru Patrașcan**

Data: ..... Semnătura: .....

Contents

Motivation **2**

Introduction **3**

1 Security and Privacy(Vaudenay's RFID model) **4**

1.1 RFID Schemes 4

1.2 Adversaries and oracles 6

1.3 Classes of adversaries 7

1.4 Security 7

1.5 Privacy 8

2 RFID Authentication **10**

2.1 Primitives used 10

2.2 Primitives' structure 10

2.3 Specific steps of the authentication protocol 12

3 The use of temporary variables **15**

3.1 Defining temporary variables 15

3.2 Temporary State Disclosure and its effects on Privacy 15

3.3 Corruption without temporary state disclosure 19

3.4 The impossibility results for resettable tags 22

3.5 The impossibility results for stateless tags 24

4 Case study **31**

4.1 Initial given protocol attack 31

4.2 Initial given scheme security 32

Conclusions **34**

Bibliography **35**

Motivation

The choice towards studying RFID schemes and their security came for multiple reasons.

Firstly the applications for this fascinating technology are vast, they provide near-instant authentication for multiple tags at the same time meaning it can be used in the medical field, for supply chains, in the automotive and construction industry and where it all started, in a military context. They provide access control from authentication of staff to providing access to ski lifts.

Secondly a typical RFID scheme is an environment where achieving security properties is difficult. The tags come with multiple constraints like processing power and limited memory and some of them can only work when interacting with a reader. This means that the RFID protocols need to strike a balance between lightness of operations and a solid security parameter. The tags are also cheap devices that are not temperproof, meaning new difficulties in developing solid protocols concerning how memory is used.

All these limitations and the possible applications of RFID schemes lead to a challenging but rewarding field of study.

Introduction

In a world that grows more and more a reliability on smart devices the need for secure and private protocols grows with it. One such technology that makes this possible is RFID. RFID is the acronym of Radio Frequency Identification and it designates a wireless technology that uses radio signals aiming to identify objects and persons by the means of devices(tags) attached to them.

During World-War II, the British wanted to distinguish between enemy and their own returning aircraft. To achieve this they placed transponders on their aircrafts which would respond to queries from the base stations. This was called the Identity Friend or Foe(IFF) system, which is considered to be the first use of RFID.

Since then RFID technology has been applied to a number of technical and scientific fields. In medicine RFID tags are used in blood transfusion and analysis. A RFID reader scans the tags attached to blood bags and finds the appropriate one for a specific patient. In the aeronautics industry, RFID tags are used for the supply chain. In the automotive industry, the tags can be attached to parts of a car and tracked during assembly. RFID also has many applications in the construction industry [6], from automated tracking of pipe spools to tracking the location of buried assets.

Chapter 1

Security and Privacy(Vaudenay's RFID model)

1.1 RFID Schemes

A typical RFID system includes three components: a coil (or antenna), a transceiver (with a decoder) and a transponder(a radio-frequency tag) embedded with unique information. The antenna emits radio signals in order to activate the tag and to read or write data on it. Antennas establish the communication between the transceiver and the tags. They can be packaged with the transceiver and the decoder in order to become a reader.

The tags are passive transponders identified by a unique ID. Their technical parameters often are: memory, power supply, shape, size and the presence of a microprocessor. For many applications the tags need to be very small and cheap so their attributes are deeply constrained. A tag most often:

- is passive: because it has no batteries it can operate only when queried by the reader and can only respond for a short time after;
- has limited memory: a few kilobits of memory;
- has limited computational ability, can only perform basic calculations: hash calculations [7], pseudorandom generation [8], symmetric encryption [9]. Some elliptic-curve arithmetic and public-key cryptography may fit, but remain expensive so far;

- provides no physical security: each tag can be opened(corrupted) and thus revealing its memory;

- communicates at up to a fixed distance: the tag has a range of a few meters.

In RFID Moore's law cannot be directly applied. In fact, the main goal of RFID transponders producers is to keep prices down and so the performance takes a secondary, less important role.

The reader is composed by one or more transceivers and a backend processing system(sometimes in the literature the reader denotes just the transceiver). The reader is able to accomodate multiple tags at the same time although a high number of them can lead to substantial delays for authentication.

The server is the component that handles the complex cryptographic calculations and eases the load on readers whenever possible. The traditional protocols included fixed reader positions and assumed secure(wired) communication. However in practical applications the communication between server and reader is also done via Radio-Frequency Signals.

RFID protocols are used most often for two goals: to identify tags(by recovering their unique ID) and to authenticate tags(to make sure a tag is legitimate i.e. registered in the database).

A frequently used and accepted model for security and privacy for RFID schemes is Vaudenay's model, proposed in [2].

RFID schemes: A RFID scheme is formed by a triple:

- a setup scheme for the reader: SetupReader(λ): generates a pair of keys: K_P and K_S for the reader depending on the security parameter λ. The public key is released to the public and the secret key is stored on the reader database.

-a setup scheme for the tag: SetupTag(ID) $\rightarrow$ (K, S): K is the tag secret and S denotes the initial state of the tag. Furthermore if the tag is legitimate the values ID and K are stored on the memory of the reader.

-an interactive protocol between the reader and the tag, at the end of comunication the reader will output a value ($\perp$, correct ID, incorrect ID) indicating succesfull authentication or not and the tag returns ok for a legitimate reader or $\perp$ otherwise. Formally: $Ident[\mathcal{T}_{ID} : S; \mathcal{R} : sk_{\mathcal{R}}, DB; * : pk_{\mathcal{R}}] \rightarrow [\mathcal{T}_{ID} : out_{\mathcal{T}_{ID}}; \mathcal{R} : out_{\mathcal{R}}]$. This is interpreted as: the identification protocol "Ident" starts with tag $\mathcal{T}_{ID}$ from state S, the reader R having the secret key $sk_{\mathcal{R}}$ and the access to database DB, and public key $pk_{\mathcal{R}}$

being shared to both parties. After a succesfull computation it leads to $\mathcal{T}_{ID}$ returning $out_{\mathcal{T}_{ID}}$ and $\mathcal{R}$ yielding $out_{\mathcal{R}}$.

The *correctness* of a RFID scheme means that after each succesfull execution of the interactive protocol between the reader and a legitimate tag, the reader outputs the tag's identify with overwhelming probability. For mutual authentication schemes correctness also includes that the tag outputs ok with overwhelming probability.

1.2 Adversaries and oracles

Adversaries: An adversary is often defined by what action they can perform, i.e. the oracles they can query, the goals of their attack(also known as the game they are playing) and by the way they can interact with a system. An adversary is a p.p.t algorithm.

A tag can be either *drawn* or *free*, this means that it is in reach of the adversary or not. A virtual tag is a temporary identifier for a tag when it is drawn.

Below are the 8 oracles defined in Vaudenay's model:

1. $CreateTag^b$*(ID)*: created a new tag, legitimate(b=1) or not(b=0) with the unique ID. This oracles uses SetupTag(ID) and if the tag is legitimate then the reader updates it's database. The reader searches the database in order to authenticate tags.

2. *DrawTag(*$distr$*)* $\rightarrow (vtag_1, b_1, ..., vtag_n, b_n)$: selects at random n tags following the distribution $distr$ and draws them to be used by the adversary. The oracle provides an array of virtual identifiers ($vtag_1, ..., vtag_n$) as the adversary needs to reference them. and the bits b_k meaning authentic or not.

A hidden table Γ is kept by the oracle containing the association between the temporary identifiers and the real IDs. This will be revealed after an attack experiment to determine if it was succesfull or not.

3. *Free(vtag)*: the vtag is eliminated from the set the attacker works with. This makes it unreachable to the adversary.

4. *Launch* $-> \pi$: the reader launches a new protocol instance π.

5. *SendReader(*m, π*)*$-> (m')$: sends the reader a message m and receives as response the message m' as part of the protocol π.

6. *SendTag(*m, π*)*$-> (m')$: sends the tag a message m and receives as response the message m'. If m is empty the response is the first message transmitted by the tag for a given protocol.

7. *Result(π)*: After the protocol has ended on the reader side, output 1 if the tag was authenticated and 0 otherwise.

8. *Corrupt(vtag)*: Outputs the current internal state of the tag temporary named vtag.

1.3 Classes of adversaries

Depending on how these oracles are used/the access the attacker has, 4 kinds of adversary are defined:

1. *STRONG*: class of adversary who have access to all the oracles, no limitations on how the Corrupt(vtag) oracle is used. Thus a strong adversary can get the state of a tag and still query other oracles for tag $\mathcal{T}$.

2. *DESTRUCTIVE*: adversaries who never use a tag after a Corrupt(vtag) oracle is called i.e. the tag is destroyed.

3. *FORWARD*: adversaries that once they call Corrupt(vtag) oracles can only call the same oracle i.e the corruption of tags are left at the end of the attacking phase.

4. *WEAK*: adversaries who have no access to the Corrupt(vtag) oracle, the internal memory of the tags remains a secret.

Another way to consider adversaries is by their access to the Result(π) oracle. If they have the access, they are considered to be *wide* adversaries, if not, they are called *narrow*. Since the access to the Corrupt() and the Result() oracles is quite orthogonal 4 new distinct adversarial models are distinguished: narrow-strong, narrow-destructive, narrow-forward and narrow-weak.

1.4 Security

The security definition of the Paise-Vaudenay model [11] encompases attacks where the adversary aims to impersonate or forge a legitimate tag $\mathcal{T}$ or the reader $\mathcal{R}$.

Consider an adversary in the class STRONG. They win if there is an instance of the protocol run by the reader where a legitimate tag is authenticated but done so when the specific steps of the protocol are followed out of order(interleaved). If the probability of the adversary to win is negligible then a RFID scheme is considered secure.

For a mutual authentication scheme the same property needs to be respected for the tag to authenticate the reader.

Reader authentication: The definition of reader authentication is based on a security experiment $Exp^{\mathcal{R}-aut}_{\mathcal{A}_{sec}}$ where a strong adversary must impersonate $\mathcal{R}$ to a legitimate tag $\mathcal{T}_{ID}$. To exclude trivial attacks, the attacker $\mathcal{A}_{sec}$ must compute some of the protocol messages, not just forward them from $\mathcal{R}$ to $\mathcal{T}_{ID}$. $Exp^{\mathcal{R}-aut}_{\mathcal{A}_{sec}} = 1$ denotes that $\mathcal{A}_{sec}$ wins the security experiment.

Definition 1: An RFID system achieves reader authentication if for every strong adversary $\mathcal{A}_{sec}$ Pr[$Exp^{\mathcal{R}-aut}_{\mathcal{A}_{sec}} = 1$] is negligible.

1.5 Privacy

Vaudenay's notion of privacy captures anonymity and unlinkability and focuses on the privacy loss in the wireless link. It is based on the existence of a simulator $\mathcal{B}$, called a **blinder**, that can simulate the reader and the tags without knowing their secrets.

The simulation aspect of the blinder catches the case of an adversary that can't use the messages transmitted, they only know how the protocol interacts and the actual data traffic is simulated. By contrasting this attacker with an adversary that eavesdropes, denies or changes messages several conclusions can be drawn about the properties of a protocol.

The privacy definition can be formalized by a privacy experiment $Exp^{prv-b}_{\mathcal{A}_{prv}} = b'$ defined by an adversary of class P, a security parameter s and $b \in_R$ 0,1. In the first phase of the experiment, the reader is initialized (using SetupReader(1^s)). The public key is available to $\mathcal{A}_{prv}$ and $\mathcal{B}$. $\mathcal{A}_{prv}$ interacts with the protocol according to its constraints. If b = 1, all queries to Launch, SendReader, SendTag and Result oracles are handled by $\mathcal{B}$. Moreover, the blinder can surveil all the other oracles. In phase two, the adversary cannot call the oracles any longer but is given the hidden table Γ of the DrawTag oracle. Finally, $\mathcal{A}_{prv}$ returns bit b'.

Informally, $Pr[Exp^{prv-b}_{\mathcal{A}_{prv}}]$ denotes the probability that an adversary $\mathcal{A}_{prv}$ can guess the bit b, meaning whether they interact with the real oracles or through a blinder $\mathcal{B}$.

Definition 2: Let P be one of the adversary classes specified above. An RFID system is P-private if for every adversary $\mathcal{A}_{prv}$ of P there exists a probabilistic polynomial time algorithm $\mathcal{B}$ (blinder) such that the advantage $Adv^{prv}_{\mathcal{A}_{prv}} = |Pr[Exp^{prv-0}_{\mathcal{A}_{prv}} = 1] -$

$Pr[Exp^{prv-1}_{\mathcal{A}_{prv}} = 1]|$ of $\mathcal{A}_{prv}$ is negligible. $\mathcal{B}$ simulates the oracles: Launch, SendReader, SendTag and Result to $\mathcal{A}_{prv}$ without having access to the reader's secret key or database. It is considered the all oracle queries $\mathcal{A}_{prv}$ makes and the responses received are also available to $\mathcal{B}$.

Privacy is defined by the ability of the adversary to infer relations about the ID from the protocol messages. If a scheme is resistent to all P-private adversaries(belonging to P class, with all P specific access) then the scheme achieves P-privacy.

An adversary $\mathcal{A}$ is trivial if there is a negligible difference between the chances of an adversary that interacts with the system to win and the chances of a blinded adversary.

All privacy notions defined in the Paise-Vaudenay Model are linked the following way:

Strong	$\Rightarrow$	Destructive	$\Rightarrow$	Forward	$\Rightarrow$	Weak
$\Downarrow$		$\Downarrow$		$\Downarrow$		$\Downarrow$
Narrow-Strong	$\Rightarrow$	Narrow-Destructive	$\Rightarrow$	Narrow-Forward	$\Rightarrow$	Narrow-Weak

Remark: The following relation is clear: $Weak \subseteq Forward \subseteq Destructive \subseteq Strong$

Chapter 2

RFID Authentication

2.1 Primitives used

The chosen paper[1]'s authentication is presented in the following chapter. The Protocol makes use of a diagonal block key matrix (DBKM) encryption algorithm for it's property of expanding the key size of the encryption without need of extensive resources. It insurres a balance between security and lightness of operations. Another component is the self updating encryption order (SUEO) and it's role is to boost security. Lastly a self updating modulus (SUM) algorithm is deployed to weaken the correlation between the plaintext and the ciphertext.

These three algorithms are used to form the block-order-modulus variable matrix encryption algorithm, dubbed the DBKM-SUEO-SUM algorithm.

2.2 Primitives' structure

[1] proposes 3 corollaries based on block matrix properties and modulo operations that make the base for the primitives.

A. **DBKM**

For two square matrices A_1 and A_2: if $A_1 \times t_1$ mod(p) = c_1 and $A_2 \times t_2$ mod(p) = c_2 exist then:

$$\begin{pmatrix} A_1 & \\ & A_2 \end{pmatrix} \times \begin{pmatrix} t_1 \\ t_2 \end{pmatrix} mod(p) = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}$$

Updating the order of the first matrix along the main diagonal yields multiple key matrices of different sizes thus expanding the key size. For the fact that it uses

2 matrices A_1 and A_2 as its building blocks the tag does not need to keep multiple separate keys in its memory.

B. **SUEO**

For 2 square matrices A_1 and A_2: if $A_1 \times t$ mod(p) = c_1 and $A_2 \times c_1$ mod(p) = c_2 exist and $A_2 \times t$ mod(p) = c_3 and $A_1 \times c_3$ mod(p) = c_4 exist then:

$$c_2 \neq c_4$$

Using the property of non-commutativity of matrices one can change the order of the multiplications to get different ciphertexts. This change is equivalent to having multiple keys for encryption but with this result security can be improved without using additional storage space.

C. **SUM**

For 2 square matrices A and B and $A \times B = I$(identity matrix)mod(p) then:

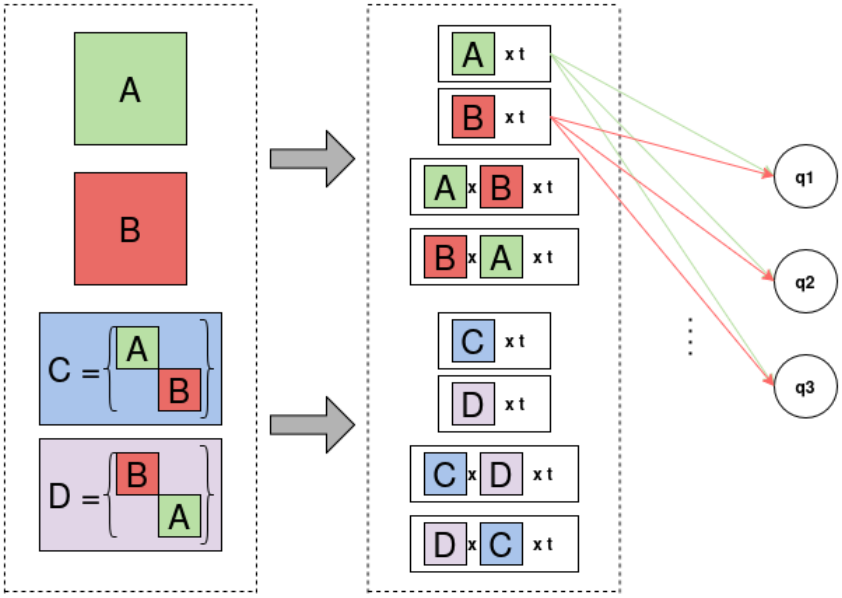
B is the modulus p-inverse matrix of A and the modulus q-inverse of A,

for a q integer divisor of p.

Thus the change in modulus still yields an inverse and the decryption process can still happen but the correlation between the ciphertext and the plaintext is weakened.

D. **DBKM-SUEO-SUM**

The above three components can be used for an encryption and decryption algorithm. The following figure represents an example with two key matrices and three integer divisors of modulus p.



2.3 Specific steps of the authentication protocol

Table 2.1: Notation used in the description of the protocol

Notation	Meaning
N_t	Nonce generated by the tag
N_r	Nonce generated by the reader
S	Secret value
S_d	Secret value that determines the construction of the block key matrix
S_p	Secret value that determines the encryption order
S_c	Secret value that determines the selection of modulus
p	Modulus
q	q $\in$ f(p), f(p) is the set of integer divisors of p
Z	Total number of DBKM index table
N	Total number of key matrices
W_i	Total number of SUEO index table
N_c	Total number of the set f(p)
A, A_{new}	Initial and updated encryption matrices
B, B_{new}	Initial and updated decryption matrices

The notation used in the protocol are shown in table 1. The details of it are are follows:

1. The reader sends a query to the tag. This activates tags without batteries.

2. The tag uses its internal pseudo-random number generator to get nonce N_t. After that the tag uses the encryption matrix A and the modulus p to encrypt $N_t||S$.

3. The tag sends the message E($N_t||S$,A,p) to the reader.

4. The reader decrypts E($N_t||S$,A,p) and obtains the secret S. If S can be queried in its internal database, the reader has authenticated the tag succesfully and accepted nonce N_t, else the protocol is stopped. After that, the reader generates nonce N_r and uses A and p to encrypt $N_r||S$.

5. The reader sends E($N_r||S$,A,p) to the server.

6. The server decrypts E($N_r||S$,A,p) and obtains secret S. A query on S suggests a succesfull authentication of the reader in which case the server accepts N_r. Contrarily the protocol is stopped. The server generates the new secret values S_d, S_p, S_c and uses A and p to encrypt $N_r||S_d||S_p||S_c$.

7. The server sends E($N_r||S_d||S_p||S_c$,A,p) to the reader.

8. The reader decrypts E($N_r||S_d||S_p||S_c$,A,p) and obtains N_r. If the value N_r received matches the saved value then the reader authenticates the server. In this case the new secret values S_d, S_p, S_c are accepted, if not the protocol is stopped. Afterwards the reader uses A and p to encrypt $N_t||S_d||S_p||S_c$.

9. The reader sends E($N_t||S_d||S_p||S_c$,A,p) to the tag.

10. The tag decrypts E($N_t||S_d||S_p||S_c$,A,p) and obtains N_t. If the value N_t received matches the saved value then the tag authenticates the reader. Following that the new secret values S_d, S_p, S_c are accepted, if not the protocol is stopped. Subsequently fresh protocol parameters are calculted: mod(S_d, Z) to determine the construction of diagonal block key matrix, mod(S_p, W_i) to determine the encryption order, mod(S_c, N_c) to determine the choice of the new modulus. With the new key matrix A_{new}, new modulus q and encryption order, the tag encrypts $N_t + 1||ID$.

11. The tag sends E($N_t + 1||ID$,A_{new},q) to the reader.

12. Reader calculates mod(S_d,Z), mod(S_p,W_i), mod(S_c,N_c) itself. Then it decrypts E($N_t + 1||ID$,A_{new},q) using the newly calculted B_{new} and q. If the value N_t matches the saved value, ID is obtained. The reader uses A_{new} and q to encrypt $N_r + 1||ID$.

13. The reader sends E($N_r + 1||ID$,A_{new},q) to the server.

14. Server calculates mod(S_d,Z), mod(S_p,W_i), mod(S_c,N_c) itself. It decrypts E($N_r + 1||ID$,A_{new},q) using the newly calculted B_{new} and q. If the value N_r matches the saved value, the server obtains ID.

The two way DBKM-SUEO-SUM-RFID authentication protocol

Server	**Reader**	**Tag**
A, B, p, f(p), S, Z, W_i	*A, B, p, f(p),N_r, S, Z, W_i*	*A, B, p, f(p),N_t, S, Z, W_i*
		1.Query →
		2.Generates N_t
		E(N_t||S,A,p)
	← 3.E(N_t||S,A,p)	
	4.D(E(N_t||S,A,p),B,p)	
	Generates N_r	
	E(N_t||S,A,p)	
← 5.E(N_r||S,A,p)		
6.D(E(N_r||S,A,p),B,p)		
Generates S_d, S_p, S_c		
E(N_r||S_d||S_p||S_c,A,p)		
→ 7.E(N_r||S_d||S_p||S_c,A,p)		
	8.D(E(N_r||S_d||S_p||S_c,A,p),B,p)	
	E(N_t||S_d||S_p||S_c, A, p)	
		→ 9.E(N_t||S_d||S_p||S_c,A,p)
		10.D(E(N_t||S_d||S_p||S_c,A,p),B,p)
		Calculates mod(S_d,Z)
		Calculates mod(S_p,W_i)
		Calculates mod(S_c,N_c)
		E(N_t+1||ID,A_{new},q)
	← 11.E(N_t+1||ID,A_{new},q)	
	12.Calculates mod(S_d,Z)	
	Calculates mod(S_p,W_i)	
	Calculates mod(S_c,N_c)	
	D(E(N_t+1||ID,A_{new},q),B_{new},q)	
← 13.E(N_r+1||ID,A_{new},q)	E(N_r+1||ID,A_{new},q)	
14.Calculates mod(S_d,Z)		
Calculates mod(S_p,W_i)		
Calculates mod(S_c,N_c)		
D(E(N_r+1||ID,A_{new},q),		
B_{new},q)		
Obtains ID		

Chapter 3

The use of temporary variables

3.1 Defining temporary variables

A tag's memory can be classified in two ways: **permanent**/internal or **temporary**/volatile. The permanent memory contains the state values of the tag, while the temporary memory is composed by temporary/volatile variables used in calculations. Temporary variables can be further split into local temporary variables: used for a single step of the protocol and global temporary variables: used to store values for a future step of the protocol.

As the definitions imply, the global temporary variables have a longer life-time than local temporary variables. They are used to store information that is later verified against received values to aid in the process of authentication. The use of them needs to be done with great care as corruption of tags can occur and by that an adversary can access information that would thwart privacy.

3.2 Temporary State Disclosure and its effects on Privacy

Types of corruption: Under tag corruption the temporary tag state is disclosed depending on the concrete scenario. In general it is under one of the following cases:

1. corruption discloses *the full state*: the permanent and the temporary memory (this is refered as "Temporary State Disclosure")

2. corruption discloses *just the permanent state*, leaving the values in the temporary memory still a secret.

The vulnerability through corruption has been highlighted for the first time in [10]. Later [3] extends the definition to cover PUF-based RFID protocols as well.

The subsequent section outlines the results of [10] and analyzes their relevance to Vaudenay's model.

Theorem 1: *There is no RFID system in Vaudenay's model that achieves both reader authentication and narrow-forward privacy under temporary state disclosure.*

This result is based on the following lemma:

Lemma 1: If every narrow-forward adversary $\mathcal{A}_{prv}$ has a blinder $\mathcal{B}$ such that $Adv^{prv}_{\mathcal{A}_{prv}}$ is negligible then $\mathcal{B}$ can be used for a blinded adversary $\mathcal{A}^{\mathcal{B}}_{sec}$ such that $Pr[Exp^{\mathcal{R}-aut}_{\mathcal{A}^{B}_{sec}} = 1]$ is non-negligible.

Proof for Theorem 1: The definition of privacy requires the existence of a blinder such that there is no difference between an adversary that uses the protocol messages and a blinded adversary $\mathcal{A}^{\mathcal{B}}$. This is equivalent to an adversary that cannot distinguish between $\mathcal{B}$ and the real oracles. Lemma 1 states that there is such a blinder that can impersonate the reader with non-negligible probability. Thus, $\mathcal{B}$ contradicts reader authentication(Definition 1).

The result of Theorem 1 and the proof for Lemma 1 are deducted in the following:

Step 1. There is assumed that there is a scheme that achieves both reader authentication and narrow-forward privacy. This will lead to a contradiction.

Step 2. Narrow-forward privacy means $\mathcal{A}_{prv}$ and blinded $\mathcal{A}^{\mathcal{B}}_{prv}$ adversaries with no discernible advantage between them. $\mathcal{A}^{\mathcal{B}}_{prv}$ simulates the Launch, SendReader and SendTag oracles. The blinded adversary is used for an attacker for reader authentication, $\mathcal{A}^{\mathcal{B}}_{sec}$.

Step 3. The construction of $\mathcal{A}^{\mathcal{B}}_{sec}$ is shown in Algorithm 1 and the win condition for $\mathcal{A}^{\mathcal{B}}_{sec}$ is established(Eq. 1).

The construction for a blinded adversary against reader authentication is:

Algorithm 1 Adversary $\mathcal{A}_{sec}^{\mathcal{B}}$ against reader authentication

1: CreateTag(ID)
2: $vtag \leftarrow$ DrawTag(ID)
3: $\pi \leftarrow$ Launch() ▷ simulated by $\mathcal{B}$
4: $m_1 \leftarrow SendReader(-, \pi)$ ▷ simulated by $\mathcal{B}$
5: i $\leftarrow 1$
6: **while** $i < stepsOfProtocol$ **do**
7: $\quad m_{2i} \leftarrow SendTag(m_{2i-1}, vtag)$ ▷ simulated by $\mathcal{B}$
8: $\quad m_{2i+1} \leftarrow SendReader(m_{2i}, \pi)$ ▷ simulated by $\mathcal{B}$
9: $\quad i \leftarrow i + 1$
10: **end while**
11: $out_{\mathcal{T}_{ID}} \leftarrow SendTag(m_{final}, vtag)$ ▷ computed by $\mathcal{T}_{ID}$

$\mathcal{A}_{sec}^{\mathcal{B}}$ uses the blinder as a black box to simulate the messages between the reader and the tag. $\mathcal{A}_{sec}$ succeds if $\mathcal{T}_{ID}$ accepts $\mathcal{B}$ as reader $\mathcal{R}$. Formally:

$$Pr[Exp_{\mathcal{A}_{sec}^{\mathcal{B}}}^{\mathcal{R}-aut} = 1] = Pr[Ident[\mathcal{T}_{ID} : S_0^{\mathcal{T}_{ID}}; \mathcal{A}_{sec}^{\mathcal{B}} : -; * : pk_R] \rightarrow [\mathcal{T}_{ID} : real; \mathcal{A}_{sec}^{\mathcal{B}} : .]]$$

(Equation 1) where $S_0^{\mathcal{T}_{ID}}$ denotes the initial tag state.

Step 4. A proof that $\mathcal{A}_{sec}^{\mathcal{B}}$ has non-negligible odds to win the security game i.e. Eq. 1 is non-negligible. This is a proof by contradiction as well. Equation 1 being negligible means a non-negligible $p_\perp$ that $out_{\mathcal{T}_{ID}} = \perp$. The consequence is that $\mathcal{A}_{prv}$ has a non-negligible advantage in distinguishing the real oracles from the simulated ones, contradicting privacy.

The construction of $\mathcal{A}_{prv}$ is shown in Algorithm 2.

Algorithm 2 $\mathcal{A}_{prv}$ against narrow-forward privacy

CreateTag(ID)
$vtag \leftarrow$ DrawTag(ID)
$\pi \leftarrow$ Launch()
$m_1 \leftarrow SendReader(-, \pi)$
i $\leftarrow 1$
while $i < stepsOfProtocol$ **do**
    $m_{2i} \leftarrow SendTag(m_{2i-1}, vtag)$
    $m_{2i+1} \leftarrow SendReader(m_{2i}, \pi)$
    $i \leftarrow i + 1$
end while
$S^{\mathcal{T}_{ID}} \leftarrow$ Corrupt(vtag)
$out_{\mathcal{T}_{ID}} \leftarrow \mathcal{T}_{ID}(S^{\mathcal{T}_{ID}}, m_{final})$
if $out_{\mathcal{T}_{ID}} =$ real **then**
    return 0
else
    return 1
end if

The adversary for privacy $\mathcal{A}_{prv}$ corrupts $\mathcal{T}_{ID}$ just before it receives the last message, m_{final}. Next, $\mathcal{A}_{prv}$ performs the computation $\mathcal{T}_{ID}$ would have done for m_{final}. A result of $out_{\mathcal{T}_{ID}} = real$ denotes that the protocol interacted with the real oracles(and the algorithm returns 0), otherwise the calls were make through $\mathcal{B}$(and the algorithm returns 1).

$\mathcal{A}_{prv}$ is a narrow-forward adversary because during the experiment there were no Result(π) calls and once a Corrupt(vtag) was called no other oracle was queried.

What remains is to assess the odds of $\mathcal{A}_{prv}$ discerning between real/simulated oracles.

Firstly, for the case in which the adversary interacts with the real system/oracles, it follows from correctness that probability p_{real} of $out_{\mathcal{T}_{ID}}$ is overwhelming, meaning $Pr[Exp^{prv-0}_{\mathcal{A}_{prv}} = 1] = 1 - p_{real}$ is negligible. Secondly we consider the case with the blinder. From the assumption $\mathcal{B}$ generates the last message such that $out_{\mathcal{T}_{ID}} = \perp$ with non-negligible probability $p_\perp$. Thus $Pr[Exp^{prv-1}_{\mathcal{A}_{prv}} = 1] = p_\perp$ is non-negligible. It follows that $Adv^{prv}_{\mathcal{A}_{prv}} = |1 - p_{real} - p_\perp|$ is non-negligible, thus a contradiction.

The contradiction means that the assuption was incorrect and $\mathcal{A}_{sec}^{\mathcal{B}}$ has non-negligible odds to win the security game.

Step 5. This adversary yields $Pr[Exp_{\mathcal{A}_{sec}^{B}}^{\mathcal{R}-aut} = 1]$ non-negligible meaning that reader authentication is disproved. This means that the assumption at step 1 is false which means Theorem 1 is correct.

3.3 Corruption without temporary state disclosure

Another result from [10] is presented in this chapter. The result obtained above shows there are more assumptions to evaluate than previously considered in the context of tag corruption. A natural direction to follow up on is what level of privacy can be achieved if temporary state disclosure is not a factor. Thus, we study the case in which corruption discloses *just the permanent state*, leaving global and local temporary variables secret.

The previous result is built upon the fact that $\mathcal{A}_{prv}$ can learn the temporary state of a tag during the Ident Protocol. This permits $\mathcal{A}_{prv}$ to verify the last message of the protocol with the tag and due to reader authentication can distinguish between real oracles and a blinded environment. However if the corruption yields only the persistent state, the previous result might not hold.

Theorem 2: *There is no RFID system in Vaudenay's model that achieves both reader authentication and narrow-strong privacy under permanent state disclosure.*

The result of Theorem 2 is deducted in the following:

Step 1. There is assumed that there is a scheme that achieves both reader authentication and narrow-strong privacy. This will lead to a contradiction.

Step 2. Narrow-strong privacy means $\mathcal{A}_{prv}$ and blinded $\mathcal{A}_{prv}^{\mathcal{B}}$ adversaries with no discernible advantage between them. $\mathcal{A}_{prv}^{\mathcal{B}}$ simulates the Launch, SendReader and SendTag oracles. The blinded adversary is used for an attacker for reader authentication, $\mathcal{A}_{sec}^{\mathcal{B}}$.

Step 3. The construction of $\mathcal{A}_{sec}^{\mathcal{B}}$ is shown in Algorithm 3 and the win condition for $\mathcal{A}_{sec}^{\mathcal{B}}$ is established(Eq. 1).

The construction for a blinded adversary against reader authentication is:

Algorithm 3 Adversary $\mathcal{A}_{sec}^{\mathcal{B}}$ against reader authentication

1: CreateTag(ID)
2: $vtag \leftarrow$ DrawTag(ID)
3: $S_0^{\mathcal{T}_{ID}} \leftarrow Corrupt(vtag)$
4: $\pi \leftarrow$ Launch() ▷ simulated by $\mathcal{B}$
5: $m_1 \leftarrow SendReader(-, \pi)$ ▷ simulated by $\mathcal{B}$
6: i $\leftarrow 1$
7: **while** $i < stepsOfProtocol$ **do**
8: $\quad m_{2i} \leftarrow SendTag(m_{2i-1}, vtag)$ ▷ simulated by $\mathcal{B}$
9: $\quad m_{2i+1} \leftarrow SendReader(m_{2i}, \pi)$ ▷ simulated by $\mathcal{B}$
10: $\quad i \leftarrow i + 1$
11: **end while**
12: $out_{\mathcal{T}_{ID}} \leftarrow SendTag(m_{final}, vtag)$ ▷ computed by $\mathcal{T}_{ID}$

$\mathcal{A}_{sec}$ succeds if $\mathcal{T}_{ID}$ accepts m_{final} and returns $out_{\mathcal{T}_{ID}} = real$. It means that $\mathcal{T}_{ID}$ accepts $\mathcal{B}$ as reader $\mathcal{R}$. Formally:

$$Pr[Exp_{\mathcal{A}_{sec}^{B}}^{\mathcal{R}-aut} = 1] = Pr[Ident[\mathcal{T}_{ID} : S_0^{\mathcal{T}_{ID}}; \mathcal{A}_{sec}^{\mathcal{B}} : -; * : pk_R] \rightarrow [\mathcal{T}_{ID} : real; \mathcal{A}_{sec}^{\mathcal{B}} : .]]$$

(Equation 2) where $S_0^{\mathcal{T}_{ID}}$ denotes the initial tag state.

Step 4. A proof that $\mathcal{A}_{sec}^{\mathcal{B}}$ has non-negligible odds to win the security game i.e. Eq. 2 is non-negligible. This is a proof by contradiction as well. Equation 2 being negligible means a non-negligible probability that $out_{\mathcal{T}_{ID}} = \perp$. Let p_t be that probability. The consequence is that $\mathcal{A}_{prv}$ has a non-negligible advantage in distinguishing the real oracles from the simulated ones, contradicting privacy.

The construction of $\mathcal{A}_{prv}$ is shown in Algorithm 4.

Algorithm 4 $\mathcal{A}_{prv}$ against narrow-strong privacy

1: CreateTag(ID)
2: $vtag \leftarrow$ DrawTag(ID)
3: $S_0^{\mathcal{T}_{ID}} \leftarrow Corrupt(vtag)$
4: $\pi \leftarrow$ Launch()
5: $m_1 \leftarrow SendReader(-, \pi)$
6: $t \in \{1, ..., q_t\}$
7: i $\leftarrow 1$
8: **while** $i < t$ **do**
9: $\quad (S_{i+1}^{\mathcal{T}_{ID}}, m_{2i}) \leftarrow \mathcal{T}_{ID}(S_i^{\mathcal{T}_{ID}}, m_{2i-1})$
10: $\quad m_{2i+1} \leftarrow SendReader(m_{2i}, \pi)$
11: $\quad i \leftarrow i + 1$
12: **end while**
13: $out_{\mathcal{T}_{ID}} \leftarrow \mathcal{T}_{ID}(S_t^{\mathcal{T}_{ID}}, m_{final})$
14: **if** $out_{\mathcal{T}_{ID}} =$ real **then**
15: $\quad$ **return** 0
16: **else**
17: $\quad$ **return** 1
18: **end if**

Notation:

$S_0^{\mathcal{T}_{ID}}$ denotes the initial(0) state S of tag $\mathcal{T}_{ID}$

q_t denotes the (expected) number of SendTag queries specified bt the Ident protocol

$\mathcal{T}_{ID}(state, message)$ denotes the computation done by $\mathcal{T}_{ID}$ when given message on specified state.

$\mathcal{A}_{prv}$ is a narrow-strong adversary because during the experiment there were no Result(π) calls and after Corrupt(vtag) was called the adversary continued to use the tag.

In this context the Corrupt(vtag) call(line 3) yields only the persistent state $S_0^{\mathcal{T}_{ID}}$ of $\mathcal{T}_{ID}$. $\mathcal{A}_{prv}$ guesses t(step 6) and simulates the tag in the Ident protocol until the reader returns m_{final}. $\mathcal{A}_{prv}$ performs the computation for $\mathcal{T}_{ID}$ with the last message m and return 0 for the case it interacted with the real oracles or 1 otherwise.

Next we study what $Adv^{prv}_{\mathcal{A}_{prv}}$ $\mathcal{A}_{prv}$ has when p_t is assumed non-negligible. For

the case of the real oracles, it follows from correctness that $out_{\mathcal{T}_{ID}} = real$ with overwhelming probability p_{real}. This means that $Pr[Exp^{prv-0}_{\mathcal{A}_{prv}} = 1] = 1 - p_{real}$ is negligible. Secondly we consider the case with the blinder. From the assumption $\mathcal{B}$ generates the last message such that $out_{\mathcal{T}_{ID}} = \perp$ with non-negligible probability p_t. Furthermore $\mathcal{A}_{prv}$ guesses number of tag messages t with probability $1/q_t$. Thus $Pr[Exp^{prv-1}_{\mathcal{A}_{prv}} = 1] \geq p_t/q_t$. The advantage becomes $Adv^{prv}_{\mathcal{A}_{prv}} \geq |1 - p_{real} - p_t/q_t|$. Due to correctness p_{real} is overwhelming, the assumption gives p_t non-negligible and q_t is polynomially bounded. It results in a non-negligible advantage which contradicts narrow-strong privacy.

The contradiction means that the assuption was incorrect and $\mathcal{A}^{\mathcal{B}}_{sec}$ has non-negligible odds to win the security game.

Step 5. This adversary yields $Pr[Exp^{\mathcal{R}-aut}_{\mathcal{A}^{\mathcal{B}}_{sec}} = 1]$ non-negligible meaning that reader authentication is disproved. This means that the assumption at step 1 is false which means Theorem 2 is correct.

3.4 The impossibility results for resettable tags

It is known that standard security notions do no longer work when the adversary can reset the internal state of the tags. Reset attacks have been motivated in particular by the use of smart cards since in specific cases they go to their initial state when disconnected from power. Subsequently they compute with the same randomness they already used bofore. Reset attacks are always possible when the adversary controls the environment.

To cover the case of reset attacks, a new oracle is introduced: Reset(vtag) to Vaudenay's model. This oracle allows the adversary to reset the state of the vtag to its original values and its randomness as well. It is assumed that it can be computed in polynomial time. As for the Corrupt(vtag) oracle the Reset(vtag) is also not carried out by the blinder $\mathcal{B}$ by merely observed by it.

Theorem 3: *For an adversary that is allowed to call the Reset oracle no privacy can be achieved in Vaudenay's model.*

To demonstrate the theorem an experiment is constructed to show that a narrow-weak adversary $\mathcal{A}_{prv}$ can distinguish with a non-negligible advantage between an interaction with the real oracles and one done through a blinder $\mathcal{B}$. This is shown in algorithm 5.

$\mathcal{A}_{prv}$ eavesdropes a complete execution of the protocol between $vtag_0$ and $\mathcal{R}$(lines

6-10). Let τ_0 be the transcript of the protocol, including the messages sent both by the reader and the tag.

To simulate $\mathcal{R}$ adversary $\mathcal{A}_{prv}$ uses the messages of τ_0 to simulate a new interaction with $vtag_1$, obtaining τ_1. If the same tag was drawn both times then $\mathcal{A}_{prv}$ expects the transcripts to match. A blinder would not know the result of DrawTag()(line 13) and would have the guess which tag was selected.

Next, the $Adv^{prv}_{\mathcal{A}_{prv}}$ is studied. For the case in which the attack uses the real oracles the reset of the tag alwats yields the same transcript a second time. The messages of the reader are the same by design of the attack and those of the tag also match because of the reset. Thus $\mathcal{A}_{prv}$ succeds with probability $1 - \epsilon(l)$ where ϵ is a negligible function and l is the security parameter. Formally:

$Pr[Exp^{prv-0}_{\mathcal{A}_{prv}} = 1] = Pr[(\Gamma[vtag_0] = \Gamma[vtag_1]) \wedge out] + Pr[(\Gamma[vtag_0] \neq \Gamma[vtag_1]) \wedge \neg out] = \frac{1}{2} * 1 + \frac{1}{2} * (1 - \epsilon(l)) = 1 - epsilon(l)/2$.

For the case of the blinder, $\mathcal{B}$ can only guess the tag selected by the second DrawTag call. Thus:

$Pr[Exp^{prv-1}_{\mathcal{A}_{prv}} = 1] = Pr[(\Gamma[vtag_0] = \Gamma[vtag_1]) \wedge out] + Pr[(\Gamma[vtag_0] \neq \Gamma[vtag_1]) \wedge \neg out] = \frac{1}{2} * \frac{1}{2} + \frac{1}{2} * frac12 = frac12$.

The difference between the two cases leads to a non-neglijible probability and therefore the privacy definition is contradicted, proving theorem 3.

Algorithm 5 $\mathcal{A}_{prv}$ against narrow-weak privacy

1: CreateTag(ID_0)
2: CreateTag(ID_1)
3: $vtag_0 \leftarrow$ DrawTag($ID_k, Pr(k) = 1/2, k \in \{0,1\}$)
4: $m_1 \leftarrow SendReader(-, \pi)$
5: i $\leftarrow 1$
6: **while** $i < stepsOfProtocol$ **do**
7: $\quad m_{2i} \leftarrow SendTag(m_{2i-1}, vtag_0)$
8: $\quad m_{2i+1} \leftarrow SendReader(m_{2i}, \pi)$
9: $\quad i \leftarrow i + 1$
10: **end while**
11: $Reset(vtag_0)$
12: $Free(vtag_0)$
13: $vtag_1 \leftarrow$ DrawTag($ID_k, Pr(k) = 1/2, k \in \{0,1\}$)
14: i $\leftarrow 1$
15: **while** $i < stepsOfTag$ **do**
16: $\quad m_{2i} \leftarrow SendTag(m_{2i-1}, vtag_1)$
17: $\quad i \leftarrow i + 1$
18: **end while**
19: **if** $\tau_0 = \tau_1$ **then**
20: $\quad out \leftarrow 1$
21: **else**
22: $\quad out \leftarrow 0$
23: **end if**
24: $return(\Gamma[vtag_0] = \Gamma[vtag_1] \wedge out) \vee (\Gamma[vtag_0] \neq \Gamma[vtag_1] \wedge \neg out)$

3.5 The impossibility results for stateless tags

This section covers the results for stateless tags, meaning tags that cannot update their persistent state. This will lead to the conclusion that destructive privacy is not possible in Vaudenay's model using such tags.

In a stateless RFID scheme the Free(vtag) oracle erases any temporary information stored on the tag. Thus the scheme will make use of tags that have the same persistent state throughout the protocol and the temporary memory is used only when

comunicating with a reader, when the tag is powered by it.

Algorithm 6 presents how an adversary would interact with a scheme using stateless tags.

Algorithm 6 $\mathcal{A}_{prv}$ against narrow-forward privacy with stateless tags

CreateTag(ID)
$vtag \leftarrow$ DrawTag(ID)
Free(vtag)
$vtag \leftarrow$ DrawTag(ID)
$t \in \{1, ..., q_t\}$
$m_1 \leftarrow SendReader(-, \pi)$
i $\leftarrow 1$
while $i < t$ **do**
    $m_{2i} \leftarrow SendTag(m_{2i-1}, vtag)$
    $m_{2i+1} \leftarrow SendReader(m_{2i}, \pi)$
    $i \leftarrow i + 1$
end while
$S \leftarrow$ Corrupt(vtag)
if $temporary\ state\ in\ S = empty$ **then**
    return 1
end if

[10] leaves us with the following lemma:

Lemma 2: The temporary state of tags is always empty in any stateless narrow-forward RFID scheme.

This is justified through contradiction, it is assumed the temporary state is non-empty. The fact is a tag would use temporary variables if it would communicate with the real oracles. If the protocol would be blinded then all SendTag() calls are simulated, meaning no temporary variables. An adversary would be able to distinguish between the two cases after a Corrupt() oracle query, shown in algorithm 6. This contradicts narrow-forward privacy, meaning Lemma 2 holds.

Lemma 2 formally means: $(S^{\mathcal{T}_{ID}}, \cdot) \leftarrow \mathcal{T}^i_{ID}(S^{\mathcal{T}_{ID}}, \cdot)$. Thus the scheme will work with a fixed persistent state and an empty temporary state in order to reach higher than weak privacy.

Theorem 4: *There is no stateless RFID scheme that achieves destructive privacy.*

Destructive privacy implies forward privacy so both must be true for a scheme at the same time. For our stateless scheme it is proved in the following that is not the case. The proof is by contradiction so we start with the presumption of privacy and arrive at a conclusion that is false.

A destructive privacy scheme implies the existence of a blinder $\mathcal{B}$ that can distinguish between the real oracles and their simulation. Using this blinder, $\mathcal{B}_D$, a narrow-forward adversary $\mathcal{A}_{prv}^{\mathcal{B}_D}$ can be constructed. This adversary will be able to thwart forward privacy, hence the contradiction.

The scheme makes use of stateless tags, meaning the persistent memory remains the same. Let $\mathcal{T}^i$ be the computation done by a tag at the i-th SendTag() query.

$\mathcal{A}_{prv}$ is defined in algorithm 7.

Algorithm 7 $\mathcal{A}_{prv}$ against destructive privacy

1: CreateTag(ID_0)
2: CreateTag(ID_1)
3: $vtag_0 \leftarrow$ DrawTag($ID_k, Pr(k) = 1/2, k \in \{0, 1\}$)
4: $\pi \leftarrow$ Launch()
5: $m_1 \leftarrow SendReader(-, \pi)$
6: $j_{\mathcal{R}} \in \{1, \ldots, q_{\mathcal{R}}\}$
7: i $\leftarrow 1$
8: **while** $i < j_{\mathcal{R}}$ **do**
9: $\quad m_{2i} \leftarrow SendTag(m_{2i-1}, vtag_0)$
10: $\quad m_{2i+1} \leftarrow SendReader(m_{2i}, \pi)$
11: $\quad i \leftarrow i + 1$
12: **end while**
13: $\mathcal{S}^{\mathcal{T}_{ID}} \leftarrow Corrupt(vtag)$
14: $b \in \{0, 1\}$
15: **if** $b = 1$ **then**
16: $\quad m_{2j_{\mathcal{R}}} \leftarrow \mathcal{T}^{j_{\mathcal{R}}}(\mathcal{S}^{\mathcal{T}_{ID}}, m_{2j_{\mathcal{R}}-1})$
17: **else**
18: $\quad$ pick a state with the same distribution as CreateTag()
19: $\quad \mathcal{S}^{\mathcal{T}_{ID}} \leftarrow \mathcal{S}$
20: $\quad m_{2j_{\mathcal{R}}} \leftarrow \mathcal{T}^{j_{\mathcal{R}}}(\mathcal{S}^{\mathcal{T}_{ID}}, m_{2j_{\mathcal{R}}-1})$
21: **end if**
22: $m_{2j_{\mathcal{R}}+1} \leftarrow SendReader(m_{2j_{\mathcal{R}}}, \pi)$
23: i $\leftarrow j_{\mathcal{R}} + 1$
24: **while** $i < q_{\mathcal{R}}$ **do**
25: $\quad m_{2i} \leftarrow \mathcal{T}^{i}(\mathcal{S}^{\mathcal{T}_{ID}}, m_{2i-1})$
26: $\quad m_{2i+1} \leftarrow SendReader(m_{2i}, \pi)$
27: $\quad i \leftarrow i + 1$
28: **end while**
29: $return(Result(\pi) \wedge b) \vee (\neg Result(\pi) \wedge \neg b)$

The idea is for the adversary $\mathcal{A}_{prv}$ to forward the messages between the tag and the reader up to a set round $j_{\mathcal{R}}$. After that $\mathcal{A}_{prv}$ corrupts $\mathcal{T}$ and gets its state. Even though the adversary is destructive, they can compute the remaining tag steps of the

protocol using the obtained state and the protocol algorithm. $\mathcal{A}_{prv}$ also picks a state at random, with the same distribution as creating a new tag. Then it picks randomly from the two states and continues the protocol until the end. When the randomly selected state is the one obtained by the Corrupt oracle, due to correctness the Result() query will yield 1 and for the other case the reader will reject it.

Algorithm 7 shows that there is an adversary that can distinguish if a tag has changed its state or not.

The definition for privacy states that there is a blinder $\mathcal{B}_D$ such that $Adv^{prv}_{\mathcal{A}_{prv}} = |Pr[Exp^{prv-0}_{\mathcal{A}_{prv}} = 1] - Pr[Exp^{prv-1}_{\mathcal{A}_{prv}} = 1]| = \epsilon(l)$ for a neglijible function ϵ. If such a $\mathcal{B}_D$ exists then it can simulate the tag and the reader for the first $j_{\mathcal{R}}$ rounds and after can simulate the reader until the protocol terminates. Two discrete phases can be observed. Thus $\mathcal{B}_D$ can answer Result() queries as well as the reader and by extension can distinguish if the messages have the same origin or not.

Next, it is shown how a narrow-forward adversary can use $\mathcal{B}_D$ to distinguish between the blinder $\mathcal{B}$ and the real oracles. This disproves narrow-forward privacy and in turn destructive privacy. $\mathcal{A}_{prv}$ uses the capabilities of $\mathcal{B}_D$ by feeding it information similarly to algorithm 7. This is described in the following, algorithm 8.

Algorithm 8 $\mathcal{A}_{prv}^{\mathcal{B}_D}$ against narrow-forward privacy

1: CreateTag(ID_0) ▷ shown to $\mathcal{B}_D \rightarrow 1$
2: CreateTag(ID_1) ▷ shown to $\mathcal{B}_D \rightarrow 2$
3: $vtag_0 \leftarrow$ DrawTag($ID_k, Pr(k) = 1/2, k \in \{0,1\}$) ▷ shown to $\mathcal{B}_D \rightarrow 3$
4: $\pi \leftarrow$ Launch() ▷ simulated by $\mathcal{B}_D \rightarrow 4$
5: $m_1 \leftarrow SendReader(-, \pi)$ ▷ simulated by $\mathcal{B}_D \rightarrow 5$
6: $j_{\mathcal{R}} \in \{1, \ldots, q_{\mathcal{R}}\}$
7: i $\leftarrow 1$
8: **while** $i < j_{\mathcal{R}}$ **do**
9: $\quad m_{2i} \leftarrow SendTag(m_{2i-1}, vtag_0)$ ▷ simulated by $\mathcal{B}_D \rightarrow 9$
10: $\quad m_{2i+1} \leftarrow SendReader(m_{2i}, \pi)$ ▷ simulated by $\mathcal{B}_D \rightarrow 10$
11: $\quad i \leftarrow i + 1$
12: **end while**
13: $m_{2j_{\mathcal{R}}} \leftarrow SendTag(m_{2j_{\mathcal{R}}-1}, vtag_0)$ ▷ computed by $vtag_0$ *with real oracles or blinder*
14: $Free(vtag_0)$
15: $vtag_1 \leftarrow$ DrawTag($ID_k, Pr(k) = 1/2, k \in \{0,1\}$)
16: $\mathcal{S}^{\mathcal{T}_{ID}} \leftarrow Corrupt(vtag_1)$ ▷ shown to $\mathcal{B}_D \rightarrow 13$
17: $m_{2j_{\mathcal{R}}+1} \leftarrow SendReader(m_{2j_{\mathcal{R}}}, \pi)$ ▷ simulated by $\mathcal{B}_D \rightarrow 22$
18: $i \leftarrow j_{\mathcal{R}} + 1$
19: **while** $i < q_{\mathcal{R}}$ **do**
20: $\quad m_{2i} \leftarrow \mathcal{T}^i(\mathcal{S}^{\mathcal{T}_{ID}}, m_{2i-1})$ ▷ computed by $\mathcal{A}_{prv}^{\mathcal{B}_D}$
21: $\quad m_{2i+1} \leftarrow SendReader(m_{2i}, \pi)$ ▷ simulated by $\mathcal{B}_D \rightarrow 26$
22: $\quad i \leftarrow i + 1$
23: **end while**
24: $b \leftarrow Result(\pi)$ ▷ simulated by $\mathcal{B}_D \rightarrow 29$
25: $return(\Gamma[vtag_0] = \Gamma[vtag_1] \wedge b) \vee (\Gamma[vtag_0] \neq \Gamma[vtag_1] \wedge \neg b)$

The adversary makes $\mathcal{B}_D$ to compute the first $j_{\mathcal{R}}$ rounds of the protocol(8-12). After, $\mathcal{A}_{prv}^{\mathcal{B}_D}$ queries the SendTag() oracle with message $m_{2j_{\mathcal{R}}-1}$ given by $\mathcal{B}_D$. This is the tag side of the protocol for step $j_{\mathcal{R}}$. Next the adversary frees $vtag_0$ and draws a new tag: $vtag_1$. Steps 13-15 are not shown to $\mathcal{B}_D$. Next $\mathcal{A}_{prv}^{\mathcal{B}_D}$ obtains the state of the new tag $vtag_1$ by corrupting it. This is shown to $\mathcal{B}_D$. $\mathcal{A}_{prv}^{\mathcal{B}_D}$ sends $\mathcal{B}_D$ the message $m_{2j_{\mathcal{R}}}$ (line 13). This message has been computed either with the real oracles or via blinder $\mathcal{B}$. $\mathcal{B}_D$ expects this message to be the product of the Corrupt() oracle(line 12, algorithm 7).

Until now the algorithm has computed $j_{\mathcal{R}}$ steps with a tag and 2 options discerned themselves, to continue with the original tag or choose another. The selected tag will be subjected to the experiment done by $\mathcal{B}_D$.

After the protocol enters phase 2 of $\mathcal{B}_D$, it plays the reader side until the end. By hypothesis $\mathcal{B}_D$ can distinguish if the messages received in the second phase match the behaviour of the tag from the first phase, in which case Result() will output 1 or not, and Result() will output 0.

What remains is to calculate the advantage $Adv^{prv}_{\mathcal{A}^{\mathcal{B}_D}_{prv}}$ if $\mathcal{B}_D$ exists.

Firstly the case in which the adversary interacts with the real oracles, i.e. uses the real SendTag() oracle (line 13 of algorithm 8). If $\Gamma(vtag_0) = \Gamma(vtag_1)$ then by the capabilities of $\mathcal{B}_D$ the Result query returns 1 with overwhelming probability, which leads to $Adv^{prv}_{\mathcal{A}^{\mathcal{B}_D}_{prv}}$ returning 1 by the same probability. Even though $\mathcal{B}_D$ receives the state $\mathcal{S}^{\mathcal{T}_{ID}}$ after the first phase by the property of Lemma 2, the state is the same as in the beggining of the protocol. If $\Gamma(vtag_0) \neq \Gamma(vtag_1)$ then the messages in the first phase were computed to the state of one vtag and the rest were computed to the state of a vtag identifying another tag. This is not what $\mathcal{B}_D$ expects and it could wrongly output 1. Let p be the probability of $\mathcal{B}_D$ outputing 0 after $j_{\mathcal{R}}$ messages computed with a random state and the rest computed by an arbitrary state $\mathcal{S}^{\mathcal{T}_{ID}}$. Thus $Pr[Exp^{prv-0}_{\mathcal{A}^{\mathcal{B}_D}_{prv}} = 1] = \frac{1}{2} * (1 - \epsilon(l)) + \frac{1}{2} * p \leq \frac{(1+p)}{2}$.

Next, we study the case where $\mathcal{A}_{prv}$ interacts with a blinder $\mathcal{B}$. For the both cases: $\Gamma(vtag_0) = \Gamma(vtag_1)$ and $\Gamma(vtag_0) \neq \Gamma(vtag_1)$ the output of the SendTag() query computed by $\mathcal{B}_D$ is by way of a random state that is with overwhelming probability different from $vtag_0$ or $vtag_1$. Thus in both cases $j_{\mathcal{R}}$ messages are computed with a random state different than $\mathcal{S}^{\mathcal{T}_{ID}}$ and the rest computed by an arbitrary state $\mathcal{S}^{\mathcal{T}_{ID}}$. Therefore $Pr[Exp^{prv-1}_{\mathcal{A}^{\mathcal{B}_D}_{prv}} = 1] = \frac{1}{2} * (1 - p) + \frac{1}{2} * p = \frac{(1-p)}{2} + \frac{p}{2} = \frac{1}{2}$. What follows is that $Adv^{prv}_{\mathcal{A}^{\mathcal{B}_D}_{prv}} \leq |\frac{(1+p)}{2} - \frac{1}{2}| = \frac{p}{2}$. Given the way in which probability p is defined and the fact that $\mathcal{B}_D$ is able to distinguish if a state changed through a protocol instance then p is non-negligible and so is $Adv^{prv}_{\mathcal{A}^{\mathcal{B}_D}_{prv}}$, concluding the proof.

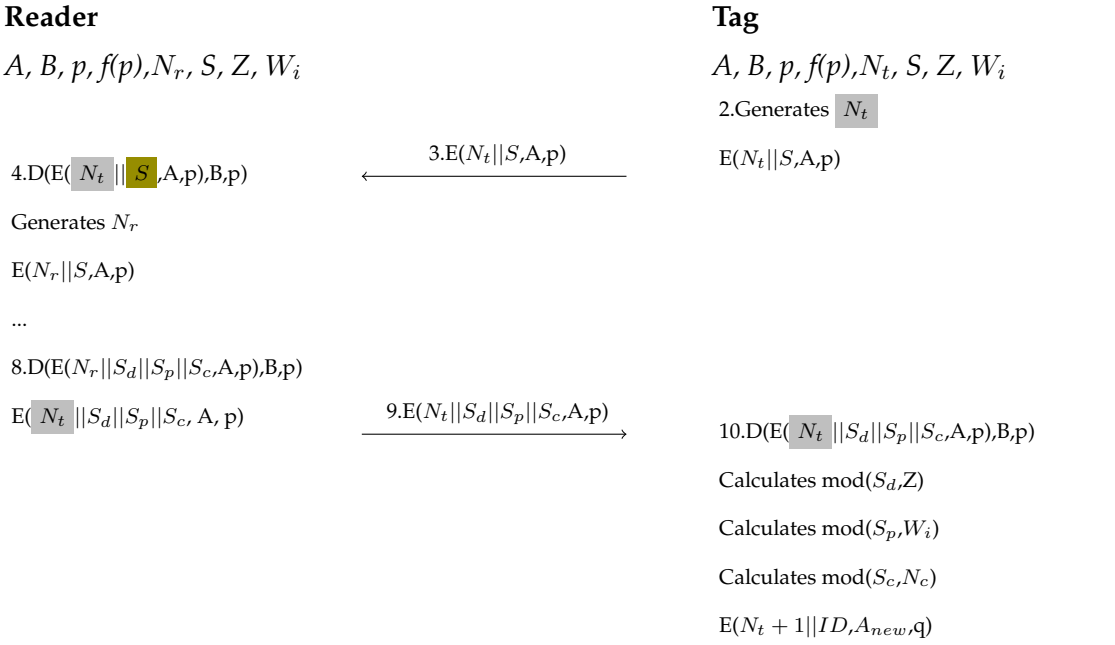
Chapter 4

Case study

4.1 Initial given protocol attack

For the authentication proposed in [1], the tag stores in it's permanent memory the values: *A, B, p, f(p), S, Z,* W_i. The global temporary memory will hold the value N_t used to authenticate the reader. It is generated at step 2 and held until just after step 10.

The two way DBKM-SUEO-SUM-RFID protocol: use of temporary variable N_t



The adversary can interact with the protocol the following way:

1. Adversary waits for a legitimate tag to respond to a reader query(step 3 of the protocol): E(N_t||S,A,p). This can be achieved by also sending a query to the tag, compeling it to compute a nonce and sending E(N_t||S,A,p).

2. Adversary intercepts the response of the reader(step 9 of the protocol)to the tag: E($N_t||S_d||S_p||S_c$,A,p)

3. Having the access to the tag, the adversary corrupts the tag and gets it's internal state. By doing that the adversary now knows A, B, p, f(p), S, Z, W_i and the nonce N_t.

4. Adversary decrypts E($N_t||S_d||S_p||S_c$,A,p) using the values B and p. Now the adversary has access to S_d, S_p, S_c and computes the values A_{new} and q.

5. Using the newly obtained A_{new} and q, adversary increments N_t, appends ID(found in the tag's memory when the corruption occured) and encrypts.

6. Adversary sends the message to the reader whom can not descern that the tag has been tampered with.

4.2 Initial given scheme security

Table 4.1: The security properties claimed by [1]

Protocol	Claimed
Mutual authentication	Yes
Location tracking	Yes
DoS	Yes
Impersonation attack	Yes
Man-in-the-middle attack	Yes
Replay attack	Yes
De-synchronization	Yes
Forward secrecy	Yes

However many of the security claims do not hold in the context of tag corruption. An important note is that an adversary can corrupt a tag during the execution of the protocol, being it earlier or later in its computation.

In the case of mutual authentication, an adversary can use the Corrupt() oracle to obtain secret S. By calling the CreateTag() oracle and using value S illegitimate tags can be authenticated by a reader. Thus tag authentication is countered. For reader authentication the adversary can corrupt multiple tags and store combinations of matrices A, B and modulus p and with an illegitimate reader be able to succesfully authenticate

legitimate tags. This is possible by being able to decrypt (step 4 of the protocol) and obtain nonce N_t which can be returned to the tag (step 9 of the protocol).

In the case of location tracking an adversary can corrupt a tag and infer relations or straight out obtain secret value S, meaning they can link it with a particular location. Strong adversaries can corrupt the tags and release them back. By sending a query(step 1 of the protocol) and breaking reader authentication the adversary can further track the tag in case it was moved.

Conclusions

The use of temporary variables needs to be done with great care, especially when the desired properties for a specific application are both security and privacy.

Bibliography

[1] Yan Wang, Ruiqi Liu, Tong Gao, Feng Shu, Xuemei Lei, Guan Gui, Fellow, IEEE, Jiangzhou Wang, Fellow, IEEE, "A Novel RFID Authentication Protocol Based on A Block-Order-Modulus Variable Matrix Encryption Algorithm"

[2] Serge Vaudenay, "On Privacy Models for RFID" 2007

[3] Ferucio Laurenţiu Ţiplea, "Lessons to be Learned for a Good Design of Private RFID Schemes"

[4] V Shoup, "Sequences of games: a tool for taming complexity in security proofs", cryptology eprint archive, 2004•eprint.iacr.org

[5] S Bocchetti, "Security and privacy in rfid protocols", Jul y, 2006•Citeseer

[6] Konstantinos Domdouzis, Bimal Kumar, Chimay Anumba, "Radio-Frequency Identification (RFID) applications: A brief introduction", Advanced Engineering Informatics, 2007 - Elsevier

[7] M. Feldhofer, C. Rechberger, "A Case against Currently used Hash Functions in RFID Protocols", On the Move to Meaningful Internet Systems 2006: OTM'06 Workshops, including the First International Workshop on Information Security (IS'06), Montpelier, France, Lecture Notes in Computer Science 4277, pp. 372-381, Springer-Verlag, 2006

[8] M.J.B. Robshaw, "Searching for Compact Algorithms: CGEN", First International Conference on Cryptology in Vietnam (Vietcrypt'06), Hanoi, Vietnam, Lecture Notes in Computer Science 4341, pp. 37-49, Springer-Verlag, 2006

[9] M. Feldhofer, S. Dominikus, J. Wolkerstorfer. Strong Authentication for RFID Systems using the AES Algorithm. In Workshop on Cryptographic Hardware and Em-

bedded Systems (CHES'04), Boston, MA, USA, Lecture Notes in Computer Science 3156, pp. 357-370, Springer-Verlag, 2004

[10] Frederik Armknecht, Ahmad-Reza Sadeghi, Alessandra Scafuro, Ivan Visconti, Christian Wachsmann, "Impossibility Results for RFID Privacy Notions"

[11] Paise, R.I., Vaudenay, S.: "Mutual authentication in RFID: Security and privacy", Proc. of ASIACCS. pp. 292-299. ACM Press (2008)