On the security of one RFID authentication protocol

Kirill Tsaregorodtsev
Researcher at Cryptography laboratory,
JSRPC “Kryptonite”, Moscow, Russia

CTCrypt’2023

КРИПТОНИТ

Outline

1. Introduction
2. Adversarial model
3. Security reduction

Introduction

Security of RFID systems

For RFID systems it is important to find a balance between technical characteristics and security.

Security of RFID systems

For RFID systems it is important to find a balance between technical characteristics and security.
The following security requirement are addressed:

1. **Party authentication** (unilateral/mutual);
2. Confidentiality (of the additional data);
3. Integrity (of the additional data);

Technical issues

1. “Near field” communications (i.e. physical restrictions on the reading range); makes the possibility of relay-attacks less severe.

Technical issues

1. "Near field" communications (i.e. physical restrictions on the reading range); makes the possibility of relay-attacks less severe.
2. Implementing simplest RFID protocols on passive tags without autonomous power sources.

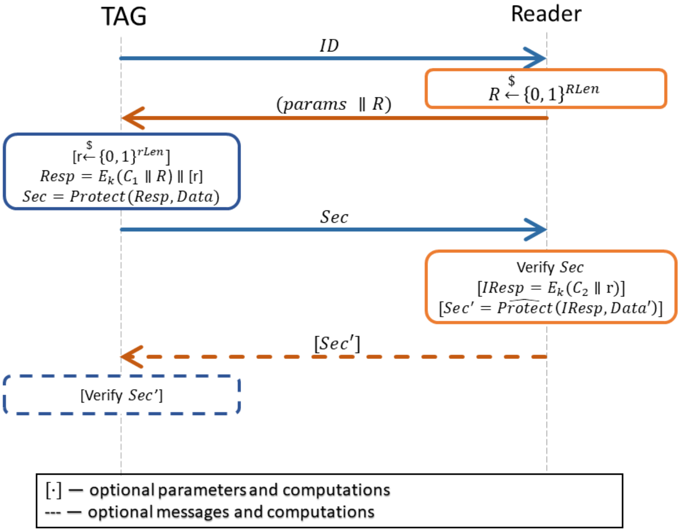
Technical issues

1. "Near field" communications (i.e. physical restrictions on the reading range); makes the possibility of relay-attacks less severe.
2. Implementing simplest RFID protocols on passive tags without autonomous power sources.
3. Protected WORM (write once, read many) memory to store shared secret keys.

Technical issues

1. "Near field" communications (i.e. physical restrictions on the reading range); makes the possibility of relay-attacks less severe.
2. Implementing simplest RFID protocols on passive tags without autonomous power sources.
3. Protected WORM (write once, read many) memory to store shared secret keys.
4. Relatively small gate area; in particular, symmetric-cryptography based protocols are preferable.

RFID authentication protocol



Adversarial model

Authentication protocol with optional data transfer

A triple $\Pi = (\mathbf{InitReader}, \mathbf{InitTag}, \mathbf{Auth})$ of (probabilistic) algorithms.

Authentication protocol with optional data transfer

A triple $\Pi = (\mathbf{InitReader}, \mathbf{InitTag}, \mathbf{Auth})$ of (probabilistic) algorithms.

- Reader initialization $\mathbf{InitReader}$; no input, returns Reader initial state.

Authentication protocol with optional data transfer

A triple $\Pi = (\mathbf{InitReader}, \mathbf{InitTag}, \mathbf{Auth})$ of (probabilistic) algorithms.

- Reader initialization $\mathbf{InitReader}$; no input, returns Reader initial state.
- Tag initialization $\mathbf{InitTag}$; input: a unique Tag ID, current Reader state, returns Tag initial state $state_{ID}$, updated Reader state.

Authentication protocol with optional data transfer

A triple $\Pi = (\mathbf{InitReader}, \mathbf{InitTag}, \mathbf{Auth})$ of (probabilistic) algorithms.

- Reader initialization $\mathbf{InitReader}$; no input, returns Reader initial state.
- Tag initialization $\mathbf{InitTag}$; input: a unique Tag ID, current Reader state, returns Tag initial state $state_{ID}$, updated Reader state.
- Authentication algorithm $\mathbf{Auth}$; input: participant's $state_A$ and a message m to be processed; returns an updated state $state'_A$ and response m'.

AUTH^+ model

Interface: oracles $CreateTag$, $StartReaderSession$, $StartTagSession$, $Send$, $Result$, $SetMessage^b$, $Test^b$:

- $CreateTag(ID)$: creates a new Tag with ID via $\Pi.InitTag$ and updates Reader database;

Interface: oracles *CreateTag*, *StartReaderSession*, *StartTagSession*, *Send*, *Result*, $SetMessage^b$, $Test^b$:

- *CreateTag*(*ID*): creates a new Tag with *ID* via $\Pi.InitTag$ and updates Reader database;
- *StartReaderSession*(*mode*): starts TAM/MAM session on the Reader's side, returns session "pointer" π;
- *StartTagSession*(*ID*): starts session on the Tag's side; returns session "pointer" π;

AUTH^+ model

Interface: oracles $\mathit{CreateTag}$, $\mathit{StartReaderSession}$, $\mathit{StartTagSession}$, Send, Result, $\mathit{SetMessage}^b$, Test^b:

- $\mathit{CreateTag}(\mathit{ID})$: creates a new Tag with ID via $\Pi.\mathit{InitTag}$ and updates Reader database;
- $\mathit{StartReaderSession}(\mathit{mode})$: starts TAM/MAM session on the Reader's side, returns session "pointer" π;
- $\mathit{StartTagSession}(\mathit{ID})$: starts session on the Tag's side; returns session "pointer" π;
- $\mathit{Result}(\pi)$: returns the result of the session π;

AUTH^{+} model

Interface: oracles $\mathit{CreateTag}$, $\mathit{StartReaderSession}$, $\mathit{StartTagSession}$, Send, Result, $\mathit{SetMessage}^b$, Test^b:

- $\mathit{CreateTag}(\mathit{ID})$: creates a new Tag with ID via $\Pi.\mathit{InitTag}$ and updates Reader database;
- $\mathit{StartReaderSession}(\mathit{mode})$: starts $\mathrm{TAM}/\mathrm{MAM}$ session on the Reader's side, returns session "pointer" π;
- $\mathit{StartTagSession}(\mathit{ID})$: starts session on the Tag's side; returns session "pointer" π;
- $\mathit{Result}(\pi)$: returns the result of the session π;
- $\mathit{SetMessage}^b(\pi, M_0, M_1)$: sets additional data to M_b in the session π;

Interface: oracles *CreateTag*, *StartReaderSession*, *StartTagSession*, *Send*, *Result*, $SetMessage^b$, $Test^b$:

- $CreateTag(ID)$: creates a new Tag with ID via $\Pi.InitTag$ and updates Reader database;
- $StartReaderSession(mode)$: starts TAM/MAM session on the Reader's side, returns session "pointer" π;
- $StartTagSession(ID)$: starts session on the Tag's side; returns session "pointer" π;
- $Result(\pi)$: returns the result of the session π;
- $SetMessage^b(\pi, M_0, M_1)$: sets additional data to M_b in the session π;
- $Send(\pi, m)$: sends message m in the session π;

Interface: oracles *CreateTag*, *StartReaderSession*, *StartTagSession*, *Send*, *Result*, $SetMessage^b$, $Test^b$:

- $CreateTag(ID)$: creates a new Tag with ID via $\Pi.InitTag$ and updates Reader database;
- $StartReaderSession(mode)$: starts TAM/MAM session on the Reader's side, returns session "pointer" π;
- $StartTagSession(ID)$: starts session on the Tag's side; returns session "pointer" π;
- $Result(\pi)$: returns the result of the session π;
- $SetMessage^b(\pi, M_0, M_1)$: sets additional data to M_b in the session π;
- $Send(\pi, m)$: sends message m in the session π;
- $Test^b(\pi)$: checks "correctness" of the session π (in case $b \neq 0$).

Interface: oracles $\mathit{CreateTag}$, $\mathit{StartReaderSession}$, $\mathit{StartTagSession}$, Send, Result, $\mathit{SetMessage}^b$, Test^b:

- $\mathit{CreateTag}(\mathit{ID})$: creates a new Tag with ID via $\Pi.\mathit{InitTag}$ and updates Reader database;
- $\mathit{StartReaderSession}(\mathit{mode})$: starts $\mathrm{TAM}/\mathrm{MAM}$ session on the Reader's side, returns session "pointer" π;
- $\mathit{StartTagSession}(\mathit{ID})$: starts session on the Tag's side; returns session "pointer" π;
- $\mathit{Result}(\pi)$: returns the result of the session π;
- $\mathit{SetMessage}^b(\pi, M_0, M_1)$: sets additional data to M_b in the session π;
- $\mathit{Send}(\pi, m)$: sends message m in the session π;
- $\mathit{Test}^b(\pi)$: checks "correctness" of the session π (in case $b \neq 0$).

Goal: guess the bit b.

Adversarial capabilities

- create legitimate tags using $CreateTag$ queries;

Adversarial capabilities

- create legitimate tags using $CreateTag$ queries;
- start sessions of chosen type using $StartReaderSession$ or $StartTagSession$ queries;

Adversarial capabilities

- create legitimate tags using $CreateTag$ queries;
- start sessions of chosen type using $StartReaderSession$ or $StartTagSession$ queries;
- check the result of the session using $Result$ query;

Adversarial capabilities

- create legitimate tags using $CreateTag$ queries;
- start sessions of chosen type using $StartReaderSession$ or $StartTagSession$ queries;
- check the result of the session using $Result$ query;
- send messages to protocol participants using $Send$ query; messages are transmitted within some fixed session π to the session holder (the Reader is able to participate in parallel sessions, but for Tags all sessions are strictly sequential, and the adversary is able to send message only in the current session for the Tag);

Adversarial capabilities-2

- set additional data to be authenticated and/or encrypted using $SetMessage^b$ query; the bit b controls which of the data messages (M_0, M_1) will be processed; in case of AE-sessions it is possible that $M_0 \neq M_1$ (this oracle formalizes the inability of adversary to break the confidentiality of the transmitted data);

Adversarial capabilities-2

- set additional data to be authenticated and/or encrypted using $SetMessage^b$ query; the bit b controls which of the data messages (M_0, M_1) will be processed; in case of AE-sessions it is possible that $M_0 \neq M_1$ (this oracle formalizes the inability of adversary to break the confidentiality of the transmitted data);
- test sessions using $Test^b$ query; if the adversary is able to authenticate without the help of Tag (or Reader), or to forge MAC-value, then it is possible to construct a session π for which there would be no matched session π', i.e., it can be tested using $Test^b(\pi)$ query; the answer helps to guess the bit b.

Adversarial capabilities-2

- set additional data to be authenticated and/or encrypted using $SetMessage^b$ query; the bit b controls which of the data messages (M_0, M_1) will be processed; in case of AE-sessions it is possible that $M_0 \neq M_1$ (this oracle formalizes the inability of adversary to break the confidentiality of the transmitted data);
- test sessions using $Test^b$ query; if the adversary is able to authenticate without the help of Tag (or Reader), or to forge MAC-value, then it is possible to construct a session π for which there would be no matched session π', i.e., it can be tested using $Test^b(\pi)$ query; the answer helps to guess the bit b.

Hence, bit b controls the following properties: confidentiality, secure participant authentication, data integrity within the session, integrity “at the session level”.

More on $Test^b$ oracle

$Test^b(\pi)$

---

if $(b = 0)$ **then**

  return 0

else

  $t_1 \leftarrow Correctness(\pi)$

  $t_2 \leftarrow \mathrm{NOT}(Match(\pi, Sessions))$

  return $(t_1 \& t_2)$

fi

Matched sessions: what and why

- *Match* predicate **binds** two sessions (from the Tag and the Reader "points of view") in one object;

Matched sessions: what and why

- $Match$ predicate **binds** two sessions (from the Tag and the Reader "points of view") in one object;
- formalizes the following logic: if the authentication finished successfully, then the legitimate partner was "alive", i.e. responded properly to the **holder's challenge**...

Matched sessions: what and why

- $Match$ predicate **binds** two sessions (from the Tag and the Reader “points of view”) in one object;
- formalizes the following logic: if the authentication finished successfully, then the legitimate partner was “alive”, i.e. responded properly to the **holder’s challenge**...
- in case of additional data with integrity check:

Matched sessions: what and why

- $Match$ predicate **binds** two sessions (from the Tag and the Reader "points of view") in one object;
- formalizes the following logic: if the authentication finished successfully, then the legitimate partner was "alive", i.e. responded properly to the **holder's challenge**...
- in case of additional data with integrity check:
  - Tag's MAC value σ_1 binds r, R and $Data$,

Matched sessions: what and why

- $Match$ predicate **binds** two sessions (from the Tag and the Reader "points of view") in one object;
- formalizes the following logic: if the authentication finished successfully, then the legitimate partner was "alive", i.e. responded properly to the **holder's challenge**...
- in case of additional data with integrity check:
  - Tag's MAC value σ_1 binds r, R and $Data$,
  - Reader's MAC value σ_2 binds r and $Data'$,

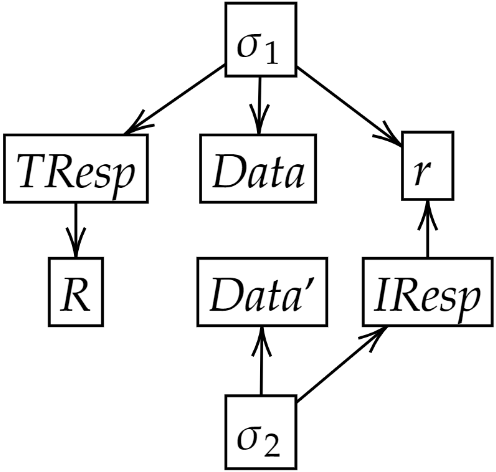
Matched sessions: what and why

- $Match$ predicate **binds** two sessions (from the Tag and the Reader "points of view") in one object;
- formalizes the following logic: if the authentication finished successfully, then the legitimate partner was "alive", i.e. responded properly to the **holder's challenge**...
- in case of additional data with integrity check:
  - Tag's MAC value σ_1 binds r, R and $Data$,
  - Reader's MAC value σ_2 binds r and $Data'$,
  - hence, $Data$ and $Data'$ are **implicitly binded**...

Matched sessions: what and why

- $Match$ predicate **binds** two sessions (from the Tag and the Reader "points of view") in one object;
- formalizes the following logic: if the authentication finished successfully, then the legitimate partner was "alive", i.e. responded properly to the **holder's challenge**...
- in case of additional data with integrity check:
  - Tag's MAC value σ_1 binds r, R and $Data$,
  - Reader's MAC value σ_2 binds r and $Data'$,
  - hence, $Data$ and $Data'$ are **implicitly binded**...
- i.e. if the Reader authentication is correct on the Tag's side, then it is guaranteed that $Data'$ is an answer not only to the Tag's challenge r, but also to the $Data$ message.

MAM-mode with integrity check



Another few words on $Test^b$

- *Match* and *Correctness* filters out "trivial" attacks that shall not be considered as security violations;

Another few words on $Test^b$

- *Match* and *Correctness* filters out "trivial" attacks that shall not be considered as security violations;
- e.g., if the adversary relays all the messages...

Another few words on $Test^b$

- *Match* and *Correctness* filters out “trivial” attacks that shall not be considered as security violations;
- e.g., if the adversary relays all the messages...
- or interrupts the delivery of the last message in the session.

Adversarial model for authentication protocol

Goal: predict the bit b (which additional data is processed inside $SetMessage^b$, whether the check in $Test^b$ is trivial or not).

Adversarial model for authentication protocol

Goal: predict the bit b (which additional data is processed inside $SetMessage^b$, whether the check in $Test^b$ is trivial or not).

Success measure:

$$\mathsf{Adv}_{\Pi}^{\mathrm{auth}^+}(\mathcal{A}) = \mathbb{P}\Big[\mathrm{Exp}_{\Pi}^{\mathrm{AUTH}^+-1}(\mathcal{A}) \to 1\Big] - \mathbb{P}\Big[\mathrm{Exp}_{\Pi}^{\mathrm{AUTH}^+-0}(\mathcal{A}) \to 1\Big],$$

Adversarial model for authentication protocol

Goal: predict the bit b (which additional data is processed inside $SetMessage^b$, whether the check in $Test^b$ is trivial or not).

Success measure:

$$\mathrm{Adv}_{\Pi}^{\mathrm{auth}^+}(\mathcal{A}) = \mathbb{P}\Big[\mathrm{Exp}_{\Pi}^{\mathrm{AUTH}^+-1}(\mathcal{A}) \to 1\Big] - \mathbb{P}\Big[\mathrm{Exp}_{\Pi}^{\mathrm{AUTH}^+-0}(\mathcal{A}) \to 1\Big],$$

Main aim: estimate the maximal advantage $\mathrm{Adv}_{\Pi}^{\mathrm{auth}^+}(\mathcal{A})$; the maximum is taken over the adversarial class with the restrictions on computational complexity of $\mathcal{A}$ and the number of queries (as well as other query characteristics that depend on the the particular application of RFID technology).

Security reduction

Reduction

- Main goal: decompose the model for the whole protocol to the sub-models for the sub-modules of the protocol;

Reduction

- Main goal: decompose the model for the whole protocol to the sub-models for the sub-modules of the protocol;
- Idea: if main "building blocks" are "good", then the whole protocol is "good".

Reduction

- Main goal: decompose the model for the whole protocol to the sub-models for the sub-modules of the protocol;
- Idea: if main "building blocks" are "good", then the whole protocol is "good".
- We were able to decompose AUTH^{+} to the model for confidentiality only ($LOR2$), and the model for the integrity AND authentication ($AUTH'$);

Reduction

- Main goal: decompose the model for the whole protocol to the sub-models for the sub-modules of the protocol;
- Idea: if main "building blocks" are "good", then the whole protocol is "good".
- We were able to decompose AUTH^+ to the model for confidentiality only ($LOR2$), and the model for the integrity AND authentication ($AUTH'$);
- The last one is decomposed further to the integrity-only model (EUF-CMA) and authentication-only model ($Chal$);

Reduction

- Main goal: decompose the model for the whole protocol to the sub-models for the sub-modules of the protocol;
- Idea: if main "building blocks" are "good", then the whole protocol is "good".
- We were able to decompose AUTH^{+} to the model for confidentiality only ($LOR2$), and the model for the integrity AND authentication ($AUTH'$);
- The last one is decomposed further to the integrity-only model ($EUF\text{-}CMA$) and authentication-only model ($Chal$);
- Each sub-model can be studied separately.

Estimating the adversarial advantage

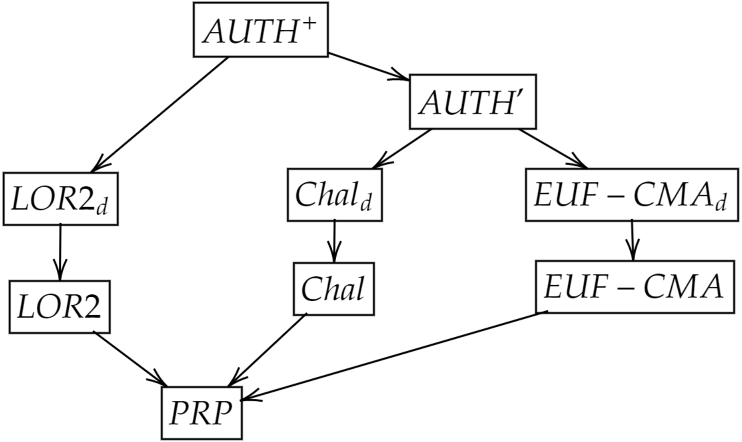


Figure 2: Schematic representation of the proof

Estimating the adversarial advantage

Theorem
The following inequality holds:

$$\mathsf{Adv}_{\Pi}^{\mathrm{AUTH}^+}(t, d, \mathcal{P}, \mathcal{Q}, \mathcal{R}, \Theta, \mathcal{M}, \mathcal{N}, \Phi, \Psi, \widehat{\mathcal{Q}}, \widehat{\mathcal{M}}, \widehat{\Phi}) \leq$$

$$\leq \sum_{i=1}^{d} \mathsf{Adv}^{LOR2}\Big(t + T, q_i, \mu_i, \phi_i, \hat{q}_i, \widehat{\mu}_i, \widehat{\phi}_i\Big) +$$

$$+2 \cdot \sum_{i=1}^{d} \mathsf{Adv}^{\text{EUF-CMA}}\Big(t + T, r_i + q_i + \hat{q}_i, \max\left(\mu_i + 2, \widehat{\mu}_i + 2, \nu_i + 1\right),$$

$$\phi_i + 2 \cdot q_i + \widehat{\phi}_i + 2 \cdot \hat{q}_i + \psi_i + r_i, \theta_i\Big) +$$

$$+2 \cdot \sum_{i=1}^{d} \Big(\mathsf{Adv}^{\mathrm{PRP}}(t + T, p_i + \theta_i) + \frac{\theta_i \cdot (p_i + \theta_i)}{2^{Rlen}} + \frac{\theta_i |Consts|}{|\mathrm{Dom}| - p_i - \theta_i + 1}\Big).$$

Example: simplifying assumptions

Let the following assumptions be true:

- The best estimate for «Magma» block cipher (in)security in sPRP-model is $\frac{q \cdot t}{2^{256}}$;
- the pair of modes $(CBC, \widehat{CBC})$ with random IV is used;
- constant length is 4 bits, challenge length is 60 bits.

Example: simplifying assumptions

Let the following assumptions be true:

- The best estimate for «Magma» block cipher (in)security in sPRP-model is $\frac{q \cdot t}{2^{256}}$;
- the pair of modes $(CBC, \widehat{CBC})$ with random IV is used;
- constant length is 4 bits, challenge length is 60 bits.

Let $p = \max_i p_i$, $\phi = \left(\max_i \phi_i + \max_i \widehat{\phi}_i + \max_i \psi_i\right)$, $\mu = \max_i (\mu_i, \widehat{\mu}_i, \nu_i)$, then it holds that:

Example: simplifying assumptions

Let the following assumptions be true:

- The best estimate for «Magma» block cipher (in)security in sPRP-model is $\frac{q \cdot t}{2^{256}}$;
- the pair of modes $(CBC, \widehat{CBC})$ with random IV is used;
- constant length is 4 bits, challenge length is 60 bits.

Let $p = \max_i p_i$, $\phi = \left(\max_i \phi_i + \max_i \widehat{\phi}_i + \max_i \psi_i\right)$, $\mu = \max_i (\mu_i, \widehat{\mu}_i, \nu_i)$, then it holds that:

$$\mathsf{Adv}_{\Pi}^{\mathrm{AUTH}^+}(t, d, \mathcal{P}, \mathcal{Q}, \mathcal{R}, \Theta, \mathcal{M}, \mathcal{N}, \Phi, \Psi, \widehat{\mathcal{Q}}, \widehat{\mathcal{M}}, \widehat{\Phi}) \leq$$

$$\leq \frac{6d\phi^2}{2^{64} - \phi} + \left(\frac{dp^2}{2^{58}} + \frac{9dp}{2^{63} - p}\right) +$$

$$+dp \cdot \mathsf{Adv}^{\mathrm{EUF\text{-}CMA}}(3p, \mu + 2, 3\phi + 5p) + + \frac{d \cdot (t + T + 2\phi) \cdot (p + \phi)}{2^{255}}.$$

Thank you for your attention!

Author(s):

Anastasiia Chichaeva

Researcher at Cryptography laboratory,
JSRPC "Kryptonite", Moscow, Russia
a.chichaeva@kryptonite.ru

Ekaterina Griboedova

Researcher at Cryptography laboratory,
JSRPC "Kryptonite", Moscow, Russia
e.griboedova@kryptonite.ru

Stepan Davydov

Researcher at Cryptography laboratory,
JSRPC "Kryptonite", Moscow, Russia
s.davydov@kryptonite.ru

Tsaregorodtsev Kirill

Researcher at Cryptography laboratory,
JSRPC "Kryptonite", Moscow, Russia
k.tsaregorodtsev@kryptonite.ru