

Reducing Time Complexity in RFID Systems

Gildas Avoine Etienne Dysli Philippe Oechslin

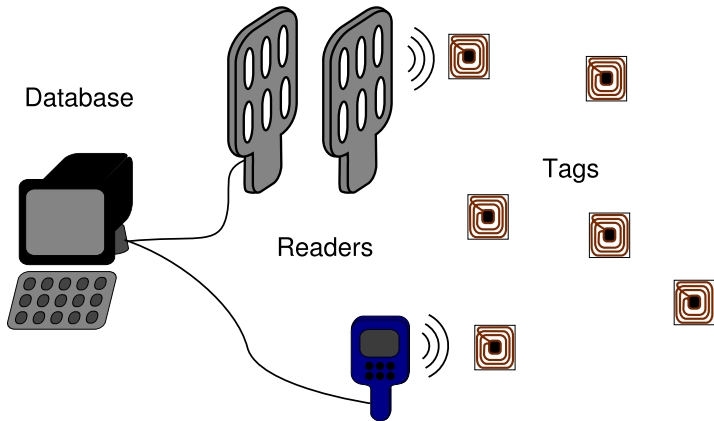
EPFL, Lausanne, Switzerland



- 1 Introduction
- 2 Description and Attack of CR/MW
- 3 Improvement of OSK Using a Time-Memory Trade-Off
- 4 Conclusion

- 1 Introduction
 - RFID Systems
 - Security and Privacy Issues
 - Cryptography for RFID Tags
 - Existing Challenge-Response Protocols

RFID Systems



Classical attacks

- denial of service
- impersonation of tags
- channel eavesdropping

Tags are vulnerable because they lack computational power and storage capacity.

Traceability

Given two tag-reader interactions, an adversary should not be able to determine whether the same tag is involved in both interactions.

Library example

- RFID tags used to label books
- automatic check-in and check-out
- eavesdropper near library gates could spy on RFID communications
- malicious reader could identify books carried by people

Tracing RFID tags is easier than with other technologies (GSM, Bluetooth, etc.)

- tags cannot be switched off
- tags answer without the agreement of their bearer
- tags have a long life (no battery)
- tags are almost invisible
- lost-cost readers and increasing communication range

Solutions to the Privacy Problem

Palliative

- kill keys
- blocker tags
- Faraday cages
- policies

Definitive

- Design an RFID protocol that allows only authorized parties to identify a tag while an adversary is neither able to identify it nor trace it.

Cryptography for RFID Tags

Private authentication

- privacy of prover (tag) has to be preserved
- mutual authentication of reader and tag

Trade-off between capacity and cost

- asymmetric crypto is too heavy for tags
- cheapest tags have no crypto at all

We will consider tags with symmetric cryptography capabilities (hash function, block cipher).

Existing Challenge-Response Protocols

Basic PRF-based private authentication protocol by Molnar and Wagner

System

choose a

\xrightarrow{a}

find (s, ID) in the database

s.t. $ID = \sigma \oplus f_s(0, a, b)$

$\xleftarrow{b, \sigma = ID \oplus f_s(0, a, b)}$

$\xrightarrow{\tau = ID \oplus f_s(1, a, b)}$

Tag

choose b

check that

$ID = \tau \oplus f_s(1, a, b)$

Does not scale well with many tags!
denoted CR thereafter

Using Symmetric Cryptography

Number of keys

- 1 secret for all tags → bad! tags are not tamper-proof
- 1 secret per tag → expensive time complexity

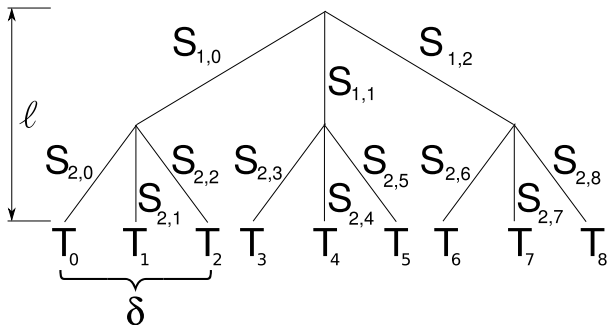
Tag identification complexity

- one tag: $O(n)$ operations (exhaustive search)
- whole system (n tags): $O(n^2)$ operations
- avoid system bottleneck (real-time library inventory)

- 2 Description and Attack of CR/MW
Molnar and Wagner's scheme
Privacy-Weakening Attack

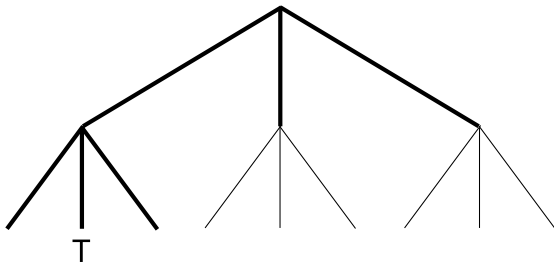
Molnar and Wagner's scheme

CR/MW



Setup

- n : number of tags in the system
- δ : branching factor
- l : depth of the tree = $\log_{\delta}(n)$



Interrogation

- tag is queried level by level from root to leaf
- if authentication fails at one level \rightarrow tag rejects reader
- instead of searching once among n secrets, search ℓ times among δ secrets

Time complexity

- identifying one tag: $\delta \log_{\delta}(n)$ operations
- identifying n tags: $n\delta \log_{\delta}(n)$ operations

Example

- library with 2^{20} tagged books, 2^{23} operations/second
- identifying one tag takes 0.002 milliseconds ($\delta = 2$)
- identifying the whole system takes 2 seconds ($\delta = 2$)

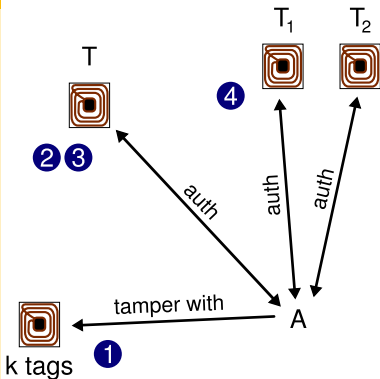
Privacy-Weakening Attack

Goal

Distinguish one tag among others

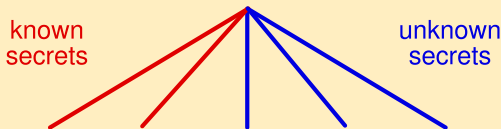
Course of the attack

- 1 attacker tampers with k tags and obtains their identifiers
- 2 chooses any target T
- 3 can query T at will but cannot tamper with it
- 4 attacker queries T_1 and T_2 to determine which of the two is T



Operation

- tags share secrets
- by opening tags, an attacker can learn parts of other tags' secrets
- at each tree level, the attacker knows one or more secrets and uses these to try to identify its target
- probability of success depends on the number of known branches at each level



Five cases

- 1 T_1 on known branch and T_2 on unknown branch
→ attack **succeeds**
- 2 T_2 on known branch and T_1 on unknown branch
→ attack **succeeds**
- 3 T_1 and T_2 both on known but different branches
→ attack **succeeds**
- 4 T_1 and T_2 both on unknown branches
→ attack definitively **fails**
- 5 T_1 and T_2 both on the same known branch
→ attack **fails** at level i but can move on to level $i + 1$

Probability that the attack succeeds is

$$\frac{k_1}{\delta^2} (2\delta - k_1 - 1) + \sum_{i=2}^{\ell} \left(\frac{k_i}{\delta^2} (2\delta - k_i - 1) \prod_{j=1}^{i-1} \frac{k_j}{\delta^2} \right),$$

where

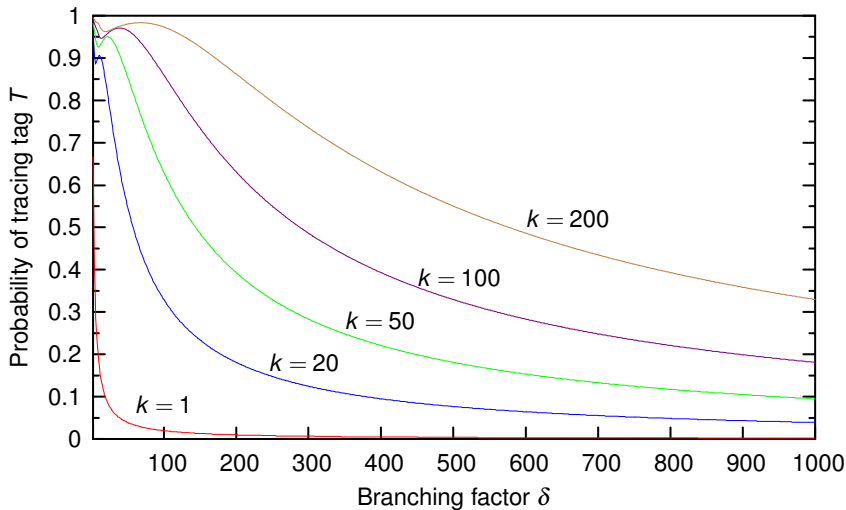
$$k_1 = \delta \left(1 - \left(1 - \frac{1}{\delta} \right)^k \right) \quad k_{i>1} = \delta \left(1 - \left(1 - \frac{1}{\delta} \right)^{g(k_i)} \right)$$

and

$$g(k_i) = k \prod_{j=1}^{i-1} \frac{1}{k_j}.$$

Tracing the Tags

Results



Tracing the Tags

Results

$k \backslash \delta$	2	20	100	500	1000
1	66.6%	9.5%	1.9%	0.3%	0.1%
20	95.5%	83.9%	32.9%	7.6%	3.9%
50	98.2%	94.9%	63.0%	18.1%	9.5%
100	99.1%	95.4%	85.0%	32.9%	18.1%
200	99.5%	96.2%	97.3%	55.0%	32.9%

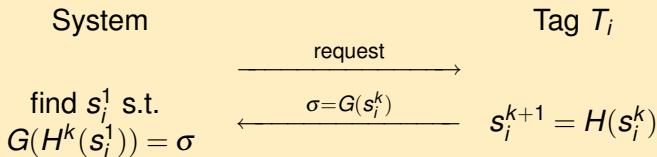
- ③ Improvement of OSK Using a Time-Memory Trade-Off
 - Ohkubo, Suzuki and Kinoshita's Protocol
 - Time-Memory Trade-Off
 - Avoine and Oechslin's Improvement

Ohkubo, Suzuki and Kinoshita's Protocol

Setup

- 2 hash functions G and H
- tag T_i stores a random identifier s_i^1
- system database contains $\{s_i^1 \mid 1 \leq i \leq n\}$

Interrogation



no mutual authentication

Ohkubo, Suzuki and Kinoshita's Protocol

Identification

From each n initial identifiers s_i^1 , the system computes the hash chains until it finds r_i^k or until it reaches a given maximum limit m on the chain length.

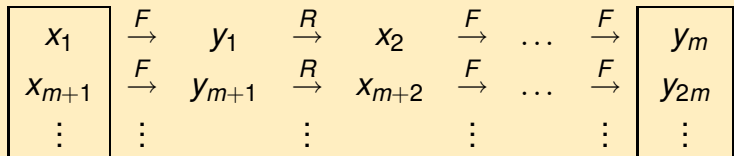
$$\begin{array}{ccccccc} s_1^1 & \rightarrow & r_1^1 & r_1^2 & \dots & \dots & r_1^{m-1} & r_1^m \\ \vdots & \rightarrow & \dots & \dots & \dots & \dots & \dots & \vdots \\ s_i^1 & \rightarrow & \dots & \dots & \boxed{r_i^k = G(H^{k-1}(s_i^1))} & \dots & \dots & r_i^m \\ \vdots & \rightarrow & \dots & \dots & \dots & \dots & \dots & \vdots \\ s_n^1 & \rightarrow & r_n^1 & r_n^2 & \dots & \dots & r_n^{m-1} & r_n^m \end{array}$$

Complexity

$m \cdot n$ hash operations (average)

Time-Memory Trade-Off

Invert a one-way function $F: X \rightarrow Y$



- *Reduction function* $R: Y \rightarrow X$ that generates an arbitrary input of F from one of its outputs.
- Given one output y_i of F , we generate a chain starting at y_i : $y_i \xrightarrow{R} x_i \xrightarrow{F} y_{i+1} \xrightarrow{R} \dots$ until we find an end of a chain. We can then regenerate the complete chain and find x_{i-1} .
- Computation time is $T \propto N^2/M^2$.

Avoine and Oechslin's Improvement

Applied to OSK

- input space of F must cover only existing identifiers, otherwise the system has no advantage over an attacker
- $F : (i, k) \mapsto r_i^k = G(H^{k-1}(s_i^1)) \quad 1 \leq i \leq n, 1 \leq k \leq m$
- $R : r_i^k \mapsto (i', k') \quad 1 \leq i' \leq n, 1 \leq k' \leq m$
e.g. $R(r) = (1 + (r \bmod n), 1 + (\lfloor \frac{r}{n} \rfloor \bmod m))$
- complexity with optimal parameters and chosen amount of memory

$$T \approx \frac{3^3}{2^3} \frac{m^3 \gamma}{c^3 \mu^2}$$

Example

- library with 2^{20} tagged books, hash chain length is 2^7
- 2^{23} hash operations/second, 1.25 GB RAM
- identifying one tag takes 0.002 milliseconds
- identifying the whole system takes 2 seconds
- precomputations require 17 minutes

Complexity comparison

- both protocols are parameterizable
- storage: on tags and in the system
- tag identification time

Scheme (parameter)	Time (milliseconds)
CR	62.500
CR/MW ($\delta = 2^{10}$)	0.122
CR/MW ($\delta = 2$)	0.002
OSK	16'000.000
OSK/AO (342 MB)	0.122
OSK/AO (1.25 GB)	0.002

Privacy and Performance

- CR secure but CR/MW degrades privacy (when tags are not considered tamper-proof)
- CR/MW trade-off between complexity and privacy
- OSK/AO private like OSK, does not modify reader \leftrightarrow tag messages
- OSK/AO can have the same performance as CR/MW