RFID privacy

Foundations of Secure e-Commerce
(bmevihim219)

Dr. Levente Buttyán
Associate Professor
BME Hálózati Rendszerek és Szolgáltatások Tanszék
Lab of Cryptography and System Security (CrySyS)
buttyan@hit.bme.hu, buttyan@crysys.hu
Outline

- the problem
- key-tree based approach
- group based approach
- privacy metrics and their comparison
- computing the level of privacy for key-trees and groups
Private authentication – the problem

- authentication protocols often reveal the identity of the authenticating party (prover) to an eavesdropper

- when devices move around and authenticate themselves frequently, the location of them can be tracked

- typical examples are RFID tags and contactless smart card based systems
An example – ISO 9798-2

- the protocol:

  (1) \( B \rightarrow A : r_B \)
  (2) \( A \rightarrow B : E(K, r_B \mid B^*) \)

where \( K \) is a shared key between \( A \) and \( B \), and \( E(.) \) denotes encryption

- “it is assumed that the parties are aware of the claimed identity of the other either by context or by additional cleartext data fields”

  (0) \( A \rightarrow B : A \)
Solutions based on public-key crypto

- encrypt identity information of the authenticating party with the public key of the verifier

- setup a confidential channel between the parties using the basic Diffie-Hellman protocol and send identity information through that channel
  - IKE in main mode works in this way

- common disadvantage: public key operations may not be affordable in devices with limited resources (e.g., public transport cards, RFID tags)
Naïve solutions for low-cost tags

- encrypt (hash) identity information with a single common key
  
  **drawback:**
  - compromise of a single member of the system has fatal consequences

- encrypt (hash) identity information with a unique key
  
  **drawback:**
  - number of keys need to be tested by the verifier grows linearly with the number of potential provers
  - doesn’t scale (potentially long authentication delay in large systems)
Better solutions for low cost tags

- **tree-based approach**
  - proposed by Molnar and Wagner in 2004
  - improved by Buttyan, Holczer, and Vajda in 2006
  **advantage:**
  - authentication delay is logarithmic in the number of members
  **drawback:**
  - increased overhead (at the prover’s side)
  - level of privacy quickly decreasing as the number of compromised members increases

- **group-based approach**
  - proposed by Avoine, Buttyan, Holczer, Vajda in 2007
  **advantage:**
  - higher level of privacy and smaller overhead than in the tree-based approach
  **drawback:** ???
The tree-based approach

try all these keys until one of them works

$k_1, k_{11}, k_{111}$

tag

$k_1$

$k_{11}$

$k_{111}$

$E(k_1, R' | R), E(k_{11}, R' | R), E(k_{111}, R' | R)$

$R$

$k_1, k_{11}, k_{111} \rightarrow$ tag ID
A problem

- if a member is compromised, its keys are learned by the adversary
- however, most of those keys are used by other members too
- the adversary can recognize the usage of those compromised keys
- consequently, the level of privacy provided by the system to non-compromised members is decreased

this decrease can be minimized by careful design of the tree!
Anonymity sets

- compromised tags partition the set of all tags
  - tags in a given partition are indistinguishable
  - tags in different partitions can be distinguished

→ each partition is the \textit{anonymity set} of its members
the level of privacy provided by the system to a randomly selected member is characterized by the average anonymity set size:

\[
\bar{S} = \sum_{i=0}^{\ell} \frac{|P_i|}{N} |P_i| = \sum_{i=0}^{\ell} \frac{|P_i|^2}{N}
\]

where N is the total number of members

we normalize this to obtain a value between 0 and 1:

\[
R = \frac{\bar{S}}{N} = \sum_{i=0}^{\ell} \frac{|P_i|^2}{N^2}
\]
Computing NAASS when a single tag is compromised

\[ |P_0| = 1 \]
\[ |P_1| = b_\ell - 1 \]
\[ |P_2| = (b_{\ell-1} - 1)b_\ell \]
\[ |P_3| = (b_{\ell-2} - 1)b_{\ell-1}b_\ell \]
\[ \ldots \]
\[ |P_\ell| = (b_1 - 1)b_2b_3 \ldots b_\ell \]

\[ R = \frac{\bar{S}}{N} = \sum_{i=0}^{\ell} \frac{|P_i|^2}{N^2} \]
\[ = \frac{1}{N^2} \left( 1 + (b_\ell - 1)^2 + ((b_{\ell-1} - 1)b_\ell)^2 + \ldots + ((b_1 - 1)b_2b_3 \ldots b_\ell)^2 \right) \]
\[ = \frac{1}{N^2} \left( 1 + (b_\ell - 1)^2 + \sum_{i=1}^{\ell-1} (b_i - 1)^2 \prod_{j=i+1}^{\ell} b_j^2 \right) \]
A trade-off between privacy and efficiency

- efficiency of the system is characterized by the maximum authentication delay:

\[ D = \sum_{i=1}^{\ell} b_i \]

- examples:
  - naïve linear key search \( \ell = 1 \)
    - \( R = 1 - 2(N-1)/N^2 \approx 1 - 2/N \approx 1 \) (if \( N \) is large)
    - \( D = N \)
  - binary key-tree \( \ell = \log N \)
    - \( R = 1/3 + 2/(3N^2) \approx 1/3 \) (if \( N \) is large)
    - \( D = 2 \log N \)

- how to maximize \( R \) while keeping \( D \) below a threshold?
The optimization problem

Given the total number $N$ of tags and the upper bound $D_{\text{max}}$ on the maximum authentication delay, find a branching factor vector $B = (b_1, b_2, \ldots, b_\ell)$ such that

$$R = \frac{1}{N^2} \left( 1 + (b_\ell - 1)^2 + \sum_{i=1}^{\ell-1} (b_i - 1)^2 \prod_{j=i+1}^\ell b_j^2 \right)$$

is maximal, subject to the following constraints:

$$\prod_{i=1}^\ell b_i = N$$

$$\sum_{i=1}^\ell b_i \leq D_{\text{max}}$$
Analysis of the optimization problem

**Lemma 1:** we can always improve a branching factor vector by ordering its elements in decreasing order.

**Lemma 2:** lower and upper bounds on $R(B)$ (where $B$ is ordered):

$$\left(1 - \frac{1}{b_1}\right)^2 \leq R(B) \leq \left(1 - \frac{1}{b_1}\right)^2 + \frac{4}{3b_1^2}$$

**Lemma 3:** given two branching factor vectors (that satisfy the constraints), the one with the larger first element is always at least as good as the other.

**Lemma 4:** given two branching factor vectors the first $j$ elements of which are equal, the vector with the larger $(j+1)$-st element is always at least as good as the other.
A solution

- let P be the ordered vector of prime factors of N
- if P doesn’t satisfy the conditions, then no solution exists
- otherwise, let P’ be a subset of P such that
  - if we multiply the prime factors in P’ (let the product be Q), then the vector (Q, P\P’) still satisfies the constraints, and
  - Q is maximal
- the first element of the optimal branching factor vector is Q
- if all prime factors are used (P\P’ = ∅), then stop
- else repeat the procedure recursively with the remaining primes
Operation of the algorithm – illustrated

- let $N = 27000$ and $D_{\text{max}} = 90$

<table>
<thead>
<tr>
<th>recursion level</th>
<th>$P$</th>
<th>$d$</th>
<th>$P'$</th>
<th>$Q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(5, 5, 5, 3, 3, 3, 2, 2, 2)</td>
<td>90</td>
<td>(3, 3, 2, 2, 2)</td>
<td>72</td>
</tr>
<tr>
<td>2</td>
<td>(5, 5, 5, 3)</td>
<td>18</td>
<td>(5)</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>(5, 5, 3)</td>
<td>13</td>
<td>(5)</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>(5, 3)</td>
<td>8</td>
<td>(5)</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>(3)</td>
<td>3</td>
<td>(3)</td>
<td>3</td>
</tr>
</tbody>
</table>

- the optimal tree for these parameters is $(72, 5, 5, 5, 3)$
  - $R \approx 0.9725$
  - $D = 90$
Proof sketch of the algorithm

- let $B^* = (b^*_1, \ldots, b^*_L)$ be the output of the algorithm
- assume that there’s a $B' = (b'_1, \ldots, b'_K) \neq B^*$ such that $R(B') > R(B^*)$

- $B^*$ is obtained by maximizing $b^*_1 \rightarrow b^*_1 \geq b'_1$
- if $b^*_1 > b'_1$ then $R(B^*) \geq R(B')$ by Lemma 3 $\rightarrow b^*_1 = b'_1$ must hold

- $B^*$ is obtained by maximizing $b^*_2$ (once $b^*_1$ is determined) $\rightarrow b^*_2 \geq b'_2$
- if $b^*_2 > b'_2$ then $R(B^*) \geq R(B')$ by Lemma 4 $\rightarrow b^*_2 = b'_2$ must hold

... 

- $B^* = B'$ must hold, which is a contradiction
The general case (any tags can be compromised)

- number and size of partitions depend on which tags are compromised
Approximation of NAASS for key-trees

- let the branching factors of the tree be $b_1, b_2, \ldots, b_L$

- select a tag $T$ randomly (without loss of generality, we assume that the left most tag of the tree is selected)

- we want to compute the expected size of $T$’s anonymity set when some tags are compromised

- we assume that each tag is compromised with probability $p = C/N$

- the probability that a given edge (key) is compromised at level $i$ is $q_i = 1 - (1 - p)^{N_i}$

where $N_i = N/(b_1 b_2 \ldots b_i)$ is the number of tags below that edge
Approximation of NAASS for key-trees

The probability that $T$’s anonymity set size is exactly $k$ ($k = 1, 2, \ldots, b_L-1$) is:

$$
(1 - q_L) \left( \frac{b_L - 1}{k - 1} \right) (1 - q_L)^{k-1} q_L^{b_L-k} = \left( \frac{b_L - 1}{k - 1} \right) (1 - q_L)^k q_L^{b_L-k}
$$
Approximation of NAASS for key-trees

- The probability that $T$’s anonymity set size is $kb_L$ ($k = 1, 2, ...$, $b_{L-1}-1$) is:

$$\left( \frac{b_{L-1} - 1}{k - 1} \right) (1 - q_{L-1})^k q_{L-1}^{b_{L-1} - k}$$
Approximation of NAASS for key-trees

- in general, the probability that $T$’s anonymity set size is $kb_L b_{L-1} \ldots b_{i+1} = kN_i \ (i = 1, 2, \ldots, L \text{ and } k = 1, 2, \ldots, b_i-1)$ is:

$$\left( \frac{b_i - 1}{k - 1} \right) (1 - q_i)^k q_i^{b_i - k}$$

- from this, the expected size of $T$’s anonymity set is:

$$\bar{S} = \sum_{i=1}^{L} \sum_{k=1}^{b_i-1} kN_i \left( \frac{b_i - 1}{k - 1} \right) (1 - q_i)^k q_i^{b_i - k} + 1 \cdot p + N \cdot (1 - p)^N$$
Verification of the approximation

B = [30 30 30]
Conclusion:
First element of the branching factor vector determines the level of privacy in the general case too.
Probability of traceability as a privacy metric

- traceability game:
  1. The adversary can tamper with a certain number $C$ of tags (Compromised tags are put back in circulation)
  2. The adversary chooses a tag $T$ and queries it as much as she wants (but she cannot compromise $T$)
  3. The adversary is presented two tags $T_1$ and $T_2$ such that $T$ is in $\{T_1, T_2\}$. The adversary can query both $T_1$ and $T_2$ as much as she wants, and she has to decide which one is $T$.

- the success probability of the adversary in the traceability game is a measure of privacy
Relation to NAASS

- if $T_1$ and $T_2$ are in the same partition then the adversary cannot distinguish them $\Rightarrow$ she cannot tell which one is $T$

- otherwise, she can distinguish $T_1$ and $T_2$ $\Rightarrow$ she can decide which one is $T$

- prob. of success $= 1 - \Pr\{T_1$ and $T_2$ are in the same partition}$

\[
\sum_{\forall |P|} \Pr\{\text{size of } T\text{’s partition is } |P|\} \cdot \frac{|P|}{N} =
\]

\[
\sum_{i=1}^{L} \sum_{k=1}^{b_i-1} \left( \frac{b_i - 1}{k - 1} \right) (1 - q_i)^k q_i^{b_i-k} \cdot \frac{kN_i}{N} + p \cdot \frac{1}{N} + (1 - p)^N \cdot \frac{N}{N} = \frac{S}{N}
\]
main idea:

• assume that a tag is compromised and this results in two equal size \( (N/2) \) partitions
• the adversary can tell each tag in either one of the partitions \( \rightarrow 1 \) bit of information has been disclosed
• in general, the amount of information that is disclosed due to tag compromise is

\[
I = \sum_{\forall P} \frac{|P| \log_2 \frac{N}{|P|}}{N}
\]

• (normalized) entropy based anonymity set size:

\[
\bar{S}_{entropy} = \frac{N}{2^I} \quad \Rightarrow \quad \frac{\bar{S}_{entropy}}{N} = \frac{1}{2^I}
\]
Comparison of NAASS and NEASS
(simulation)

$B = [30 \ 30 \ 30]$
The group-based approach

1.) try all group keys until one of them works
2.) authenticate the tag by using its individual key

Immediate advantage:
Each tag stores and uses only two keys
Computing NAASS for the group-based appr.

- partitioning depends on the number $C$ of compromised groups

- NAASS can be computed as:

\[
\frac{\bar{S}}{N} = \sum_{i} \frac{|P_i|^2}{N^2} = \frac{nC + (n(\gamma - C))^2}{N^2}
\]

- if tags are compromised randomly, then $C$ is a random variable
  - we are interested in the expected value of $S/N$
  - for this we need to compute $E[C]$ and $E[C^2]$
Computing NAASS for the group-based appr.

- let $c$ be the number of compromised tags
- let $A_i$ be the event that at least one tag is compromised from the $i$-th group
- the probability of $A_i$ can be computed as:

$$P(A_i) = 1 - \left( \frac{N-n}{c} \right) = 1 - \frac{c-1}{N} \prod_{j=0}^{c-1} \left( 1 - \frac{n}{N-j} \right)$$
Computing NAASS for the group-based appr.

- let $I_{A_i}$ be the indicator function of $A_i$
- $E[C] = ?$
- $E[C^2] = ?$

$$E[C] = E \left[ \sum_{i=1}^{\gamma} I_{A_i} \right] = \sum_{i=1}^{\gamma} P(A_i) = \gamma \left( 1 - \prod_{j=0}^{e-1} \left( 1 - \frac{n}{N-j} \right) \right)$$

$$E[C^2] = E \left[ \left( \sum_{i=1}^{\gamma} I_{A_i} \right)^2 \right] = E \left[ \sum_{i=1}^{\gamma} I_{A_i} \right] + E \left[ \sum_{i \neq j} I_{A_i \cap A_j} \right] = E[C] + (\gamma^2 - \gamma) P(A_i \cap A_j)$$
Computing NAASS for the group-based appr.

\[
P(A_i \cap A_j) = 1 - P(\overline{A_i} \cap \overline{A_j}) - 2P(A_i \cap \overline{A_j})
\]

\[
P(\overline{A_i} \cap \overline{A_j}) = \frac{{N - 2n \choose c}}{{N \choose c}} = \prod_{j=0}^{c-1} \left(1 - \frac{2n}{N-j}\right)
\]

\[
P(A_i \cap \overline{A_j}) = P(A_i|\overline{A_j})P(\overline{A_j}) = \left[1 - \prod_{j=0}^{c-1} \left(1 - \frac{n}{N-n-j}\right)\right] \cdot \prod_{j=0}^{c-1} \left(1 - \frac{n}{N-j}\right)
\]
Verification of the approximation

$N = 27000$

$\gamma = 90$
Comparison of approaches

- select a privacy metric (NAASS or NEASS)
- for a given set of parameters (number N of tags, max authentication delay D), determine the optimal key-tree
- compute the privacy metric for the optimal tree (function of c)
- determine the corresponding parameters for the group based approach ($\gamma = D-1$)
- compute the privacy metric for the groups (function of c)
Comparison in NAASS

$N = 2^{14}$
$D = 65$
Comparison in NEASS

\[ N = 2^{14} \]
\[ D = 65 \]
Summary

- we studied the problem of (efficient) symmetric-key private authentication

- we presented two approaches: key-trees and groups

- we gave an overview of proposed privacy metrics
  - NAASS, NEASS, prob. of traceability

- we showed some relationships between the metrics
  - prob. of traceability ~ NAASS, NEASS < NAASS

- we gave precise approximations of the NAASS for trees and for groups

- we compared the tree and the group based approaches using NAASS and NEASS
Conclusions

- we obtained controversial results
  - group-based approach achieves better privacy if we use NAASS
  - tree-based approach achieves better privacy if we use NEASS

→ be cautious which metric you use!

- yet, the difference between trees and groups does not seem to be large in terms of privacy

- groups may be a better trade-off, due to the smaller overhead

→ the group-based approach is a serious alternative to the tree-based approach
Open problems

1. Closed form approximation of the NEASS (both for trees and groups) ?

2. How to find the optimal tree when the metric is the NEASS?

3. How to preserve the efficiency of the tree and the group-based approaches and eliminate the exponential decrease of the level of privacy at the same time ??