EE 418 Project 1 – RFID Security and Privacy

Autumn 2017

Instructor: Xuhang Ying
Network Security Lab (NSL)
Department of Electrical Engineering

Many thanks to Prof. Radha Poovendran and Dr. Tamara Bonaci.
Project Guidelines

• Due **Nov 9 (Thursday), 2017**
• Complete **both** parts of the project.
• Max. group size is **3**. Email the names of your group members to the instructor and TA by **Oct 23 (Monday)**.
• Submit both **project report** and **source code** via Dropbox
  • Matlab code is provided and supported, but you may switch to other languages (e.g., Python).
  • You provide in-line comments to help understand your code, and make sure your code is ready to run.
• On the front page of your project report, provide
  • Names and student IDs of group members.
  • Clear description of each member’s contribution.
Part I – Attacks on RFID Mutual Authentication
Radio Frequency Identification (RFID)

• RFID Tags
  • Store information
  • Transmit information when queried
  • Low power, computation limits

• RFID Readers
  • Query tags, obtaining stored information
  • Relay information to central database
RFID Mutual Authentication

- Tags may store sensitive information
  - Passport data, credit cards, medical information
  - Tags are resource limited, and too weak for crypto operations.

- How do a reader and a tag authenticate each other?
  - 1\textsuperscript{st} Goal: Authenticate a reader to a tag
    - Tag should only send information to authenticated reader
  - 2\textsuperscript{nd} Goal: Authenticate a tag to a reader
    - Prevent attackers, impersonating tags, from giving false information to a reader.
  - Solutions: rely on bitwise operations.
Bitwise Operations

• Concatenation (||)
  • Example: 0011 || 0100 = 00110100

• Bitwise XOR (⊕)
  • 0 if bits are the same, 1 otherwise.
  • Example: 0011 ⊕ 0100 = 0111

• Circular shift to right (>>)
  • Shift to the right by r bits, wrap the last r bits around.
  • Example: 0011 >> 2 = 1100

• Circular shift to left (<<)
  • Shift to the left by l bits, wrap the last l bits around.
  • Example: 0011 << 1 = 0110
The Song-Mitchecll Protocol

• All variables are strings of \( l \) bits, \( h \) is hash

Reader

\[(s, t)\]
- \( s \): string for the tag
- \( t = h(s) \): hash of \( s \)
- \( r1 \): a random string

Check if \( M2 = h(t || (r1 \oplus M1 \oplus t)) \)

Compute:
- \( r2 = M1 \oplus t \)
- \( M3 = s \oplus (r2 >> l/2) \)
- \( s = ((s << l/4) \oplus (t >> l/4) \oplus r1 \oplus r2) \)
- \( t = h(s) \)

Tag

\[(t)\]
- Choose \( r2 \)
- \( M1 = t \oplus r2 \)
- \( M2 = h(t || (r1 \oplus r2)) \)

\( s = M3 \oplus (r2 >> l/2) \)

Check if \( h(s) = t \)

If yes, update
- \( t = h((s << l/4) \oplus (t >> l/4) \oplus r1 \oplus r2) \)
Impersonating a Tag

Reader

Adversary

Tag

r1

M1 = r1 \oplus r1' \oplus M1'
M2 = M2'

Authenticated as a tag!

Accepted as a tag!

r1'

M1', M2'
Impersonating a Reader – Step 1

- At attacker eavesdrops on a legitimate session.

```
Reader          Adversary          Tag
Not updated     r1                 Not updated
M1, M2          M3                 Not updated
```
Impersonating a Reader – Step 2

• \([x]_L – \text{Left half of string } x; \ [x]_R – \text{Right half of string } x\)

\[
[M3']_L = [M1]_R \oplus [M3]_L \oplus [M1']_R \\
[M3']_R = [M1]_L \oplus [M3]_R \oplus [M1']_L
\]

Authenticated as reader

Adversary

\(r1'\)

Tag

\(M1', M2'\)

\(M3\)

Authenticated adversary as reader
Your Assignment

• Read the papers by Song and Mitchell, Cai et al.

• Simulate both the Tag Impersonation Attack and the Server Impersonation Attack.

• Answer questions in the project assignment.
Part II – Secure RFID System Design
Scalable RFID Systems

- So far, we have considered one tag and one reader.
- In practice, the system may consist of millions of tags.
- Question: How can a reader differentiate them?

- In order to privately identify a tag, a reader needs to know the secret key.
- But the reader does not know the secret key until it identifies the tag.
- Question: Can we privately and efficiently identify the tag?
Tree-Based Approach – Setup

- Start with a binary tree

![Binary Tree Diagram]

- Each edge corresponds to a unique secret key.
- Each leaf corresponds to a unique tag.
- Each tag is given keys on the path from the root to its leaf (Molnar et al `2004)
Tree-Based Approach – Identification

- One round of protocol for each branch
- Reduce database search time
Tag Compromise Attack

• Suppose that an attacker can \textit{physically} compromise a tag.
  • It can obtain \textit{all keys held by the tag}
  • It can now breach \textit{privacy of uncompromised tags}.

Tags 1 and 2 can be differentiated. Tags 1 and 2 cannot be differentiated.
Constant-Time Approach

• Can we authenticate efficiently while avoiding tag compromise?
• Yes – by using a larger database (Alomar et al`2010)

<table>
<thead>
<tr>
<th>H(Pseudonym $\psi$, counter $c$)</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h(000, 1)$</td>
<td>$k_0$</td>
</tr>
<tr>
<td>$h(001, 2)$</td>
<td>$k_1$</td>
</tr>
<tr>
<td>$h(001, 3)$</td>
<td>$k_2$</td>
</tr>
</tbody>
</table>

c: internal counter of tag that is incremented every time authentication fails.
Your Assignment

• Read the papers by
  • Molnar and Wagner
  • Alomair et al.

• Answer questions as part of written report.
  • Including simulation of tag compromise attack.