Secure communication between web browsers and NFC targets by the example of an e-ticketing concept

Peter Kleebauer
University of Applied Sciences Hagenberg
EC-Web `08 – September 3rd, 2008
NFC – What is it all about ...

- NFC can be seen as a further development of RFID
- Radio Frequency (RF) based proximity coupling technology
- Range: 0 – 10 cm (proximity Technology)
- Integrated in mobile devices for consumer market
  - Mobile phones
  - PDAs
- Transmissions on unsecured communication ways
  - Integrity and Authenticity must be guaranteed
  - Authorization mechanism must be implemented
NFC Device Operating Modes

- **Data exchange (P2P – NFC peer-to-peer)**
  - Bidirectional connection to exchange data between devices
  - P2P Payment, Contacts, vCards, …

- **Reader/Writer mode (PCD – Proximity Coupling Device)**
  - Mobile Device is able to read external tags/smartcards
  - SmartPoster, WiFi Config, …

- **Tag emulation (PICC – Proximity Card)**
  - Reader can’t distinguish between smartcard & tag emulation
  - Handset could contain multiple smartcards (smartcard chips)
NFC Secure Element

- Dynamic environment where applications can be stored and administrated
  - Delimited memory for each application (sandbox)
  - No communication possible between different applications
  - Cryptographic functions to encrypt, decrypt or sign data
Goal of the Thesis: NFC Secure Communication

- Simple communication between web browser and NFC devices (e.g. mobile phones)
  - Installation without any user activity (web browser plug-in)
  - Better usability (known tools)
- Secure communication protocol
  - Prevention of any data manipulation (AAA: authenticity, authorization, accounting)
  - Bilateral authentication between all communication parties
  - Timely transmission of tickets (or other data)
- Ubiquitous applications
  - Authentication on web sites
  - Payment
Security concept – Public Key Infrastructure

Participants
- Control instance – Trusted Third Party (TTP)
  - Self-signed certificate
  - Confirms the validity of the ticket by its signature
  - Implemented as Web server application
- Issuer
  - Responsible for ticket generation and accounting
  - Implemented as Web Server (View - HTML Content) and Web server application
- Access control
  - Controls the protocol – communication with Secure Element
  - Examines the authenticity of tickets
- Ticket Memory
  - Application for managing tickets in the Secure Element
  - Performing cryptographic functions
  - Implemented as JavaCard application
Ticket

- **ID_I + ID_E** for event
  - Identification of the Issuer/Event
  - Split between Issuer and Event
- **ID_T** for ticket
  - Identification of the ticket
- **Payload**
  - Counter, name, period of validity
- **KeyI**
  - Public key of the Issuer
  - Key is used to encrypt communication during verification process (issuer content)
- **KeyT**
  - Public key of the ticket (identification)
  - Key is used to encrypt communication during verification process (ticket content)
Setup - Key Exchange

- **Issuer**
  - Public/Private Key *Issuer*
  - Public key *Control Instance*
  - Public/Private Key *Tickets*

- **Ticket Memory**
  - Public/Private Key *Ticket Memory*
  - Public Key *Control Instance*

- **Access Control**
  - Public/Private Key *Issuer*
  - Private Keys *Tickets (ticket database)*

-ticket request

*Control Instance*

- Public/Private Key *Control Instance*
- Public Key *Issuer*
- Public Key *Ticket Memory*
Protocol – Ticket Preparation

Login and ticket credentials

- Establish secure connection (1)
  - Web browser and Web server (https)
- Website login using the MifareID
Protocol – Ticket Preparation

**Login and ticket credentials**
- Establish secure connection (1)
  - Web browser and Web server (https)
- Website login using the MifareID
  - Challenge – Response procedure
  - Random number prevents Replay attacks
- Ticket data transferred to proxy application (2)
  - XML based structure
  - Further communication controlled by proxy application
Protocol – Ticket Preparation

Ticket preparation

- Proxy application establish new secured connection (3)
  - Transfer xml based ticket request
- Preparing ticket
  - Web server „ticket production“
- Ticket signing via Control Instance (4,5)
  - Web server „ticket signing“
  - Signed with Private Key Control Instance
  - Encrypted with Public Key Ticket Memory
- Signed and encrypted ticket (6)
  - Proxy application

```
<table>
<thead>
<tr>
<th>IDI</th>
<th>IDE</th>
<th>IDT</th>
<th>Value</th>
<th>Information</th>
<th>Date</th>
<th>KeyI</th>
<th>KeyT</th>
<th>Signature</th>
</tr>
</thead>
</table>
```

Signed and encrypted ticket (6)
Protocol – Ticket Preparation

Ticket transfer

- Signed and encrypted ticket is processed by JavaCard application
  - Ticket decryption
  - Signature check (Control instance)
  - Ticket stored in Secure Element (7)
- Status information (8)
  - Issuer activates the ticket (9)
  - User confirmation (web browser) (10)
Protocol – Ticket Verification

- Bilateral authentification (authentication of the Access Control and the Ticket Memory)
- Encrypted communication
  - Issuer content encrypted with KeyI (part of the ticket)
  - Ticket content encrypted with KeyT (part of the ticket)
- Strictly scheduled protocol sequence
  - Random numbers prevent Replay attacks
- Ticket modification within the ticket itself and the Access Control ticket database
- Collection of protocol errors
  - Protocol aborts produces defined status
Protocol – Ticket Verification

**Issuer content**
- Valid event IDs are enumerated (1,2)
- **Access Control** choose event ID (3)
  - **Access Control** is authenticated (R2)

**Ticket content**
- **Ticket Memory** sends ticket data (4)
  - **Ticket Memory** is authenticated
- **Access Control** modifies the ticket (5)
  - Ticket is stored in the Secure Element
- **Ticket Memory** sends the modified ticket (6)

<table>
<thead>
<tr>
<th>Access Control</th>
<th>Communication: ISO 14443</th>
<th>Ticket Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ID_Issuer</td>
<td>R1</td>
</tr>
<tr>
<td>2</td>
<td>(R1, R2, ID_events)_{KeyT}</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>(R2, R3, ID_Event)_{KeyT}</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>(R3, R4, ticket_data)_{KeyT}</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>(R4, R5, ticket_data)_{KeyT}</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>(R5, ticket_data)_{KeyT}</td>
<td></td>
</tr>
</tbody>
</table>

Access

Ticket modification
Summarize Security issues

- The *Ticket Memory* is implemented as JavaCard Applet in the Secure Element
  - At no point a 3rd party can access information in the Secure Element without holding the correct key
  - Authorized instances are not able to read other ticket information than their own
- Without a bilateral authentication, neither the smartcard nor the server application will allow a transaction
  - Server credentials in the JRE Certification Store
  - Client credentials in the Secure Element
- No User interaction required at Gate or when ticket is received
  - Good usability to the end users beside ensuring high security
- Issuer immediately knows whether the ticket arrived safely or not
Happy to answer any questions

Peter Kleebauer
University of Applied Sciences Hagenberg
peter.kleebauer@fh-hagenberg.at