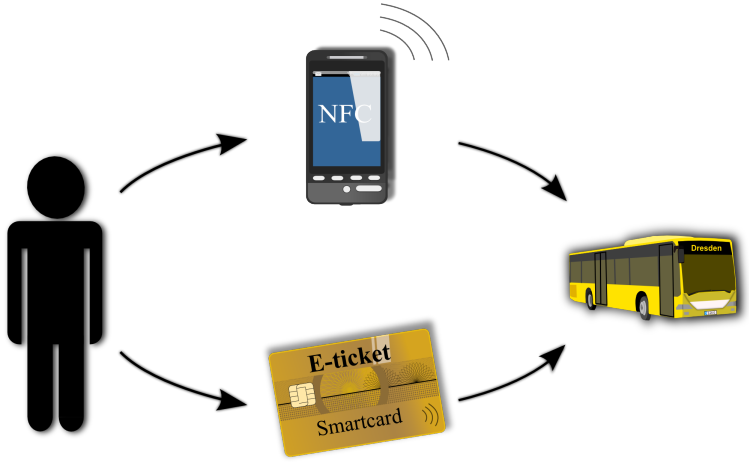
A Simple and Secure E-Ticketing System for Intelligent Public Transportation based on NFC



Ivan Gudymenko

TECHNISCHE UNIVERSITÄT DRESDEN

Felipe Sousa

UFCG
SUPRA OMNES LUX LUCIS
Universidade Federal de Campina Grande

Stefan Köpsell

TECHNISCHE UNIVERSITÄT DRESDEN

Outline

Outline

Intelligent Public Transportation and Urban Environment of the Future

- Intelligent Transportation Systems (ITS) are going to shape the urban environment of the future

- **Public transportation** is an integral part of ITS

E-ticketing in Public Transportation

[Courtesy of MünsterscheZeitung.de]

The Notion of an E-ticket

- A digitalized version of a travel permission (e.g., a token)
- Stored on a user device:
  - Smart Card
  - NFC-enabled smart phone

Non-interactive vs. Interaction-based E-ticketing Systems

- Non-interactive

- Interaction-based

Non-interactive vs. Interaction-based E-ticketing Systems

- Non-interactive

- Interaction-based

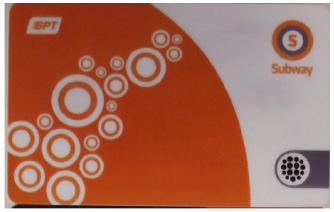
Outline

Our Goal

- Providing a digital alternative to a conventional paper-based ticketing
- Based on open source components
- NFC-enabled smart phone as a user device

Many Cards – One Single App

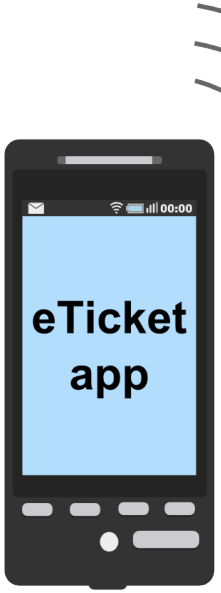
Glasgow



Karlstad

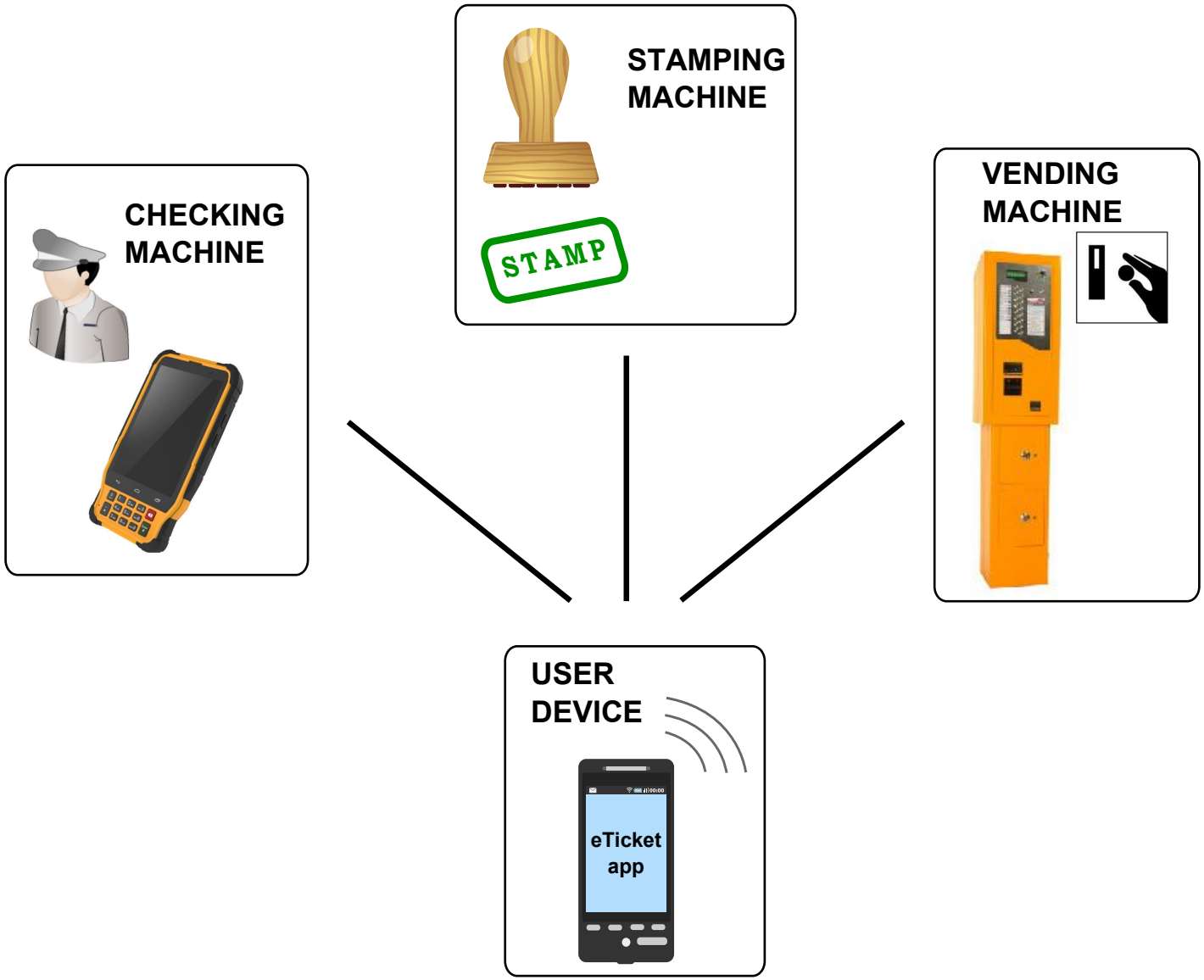
Amsterdam

Porto



Interoperable solution

System Main Actors



Core Processes Considered

(1) **E-ticket acquisition:**

a) *(online)* via a Web interface

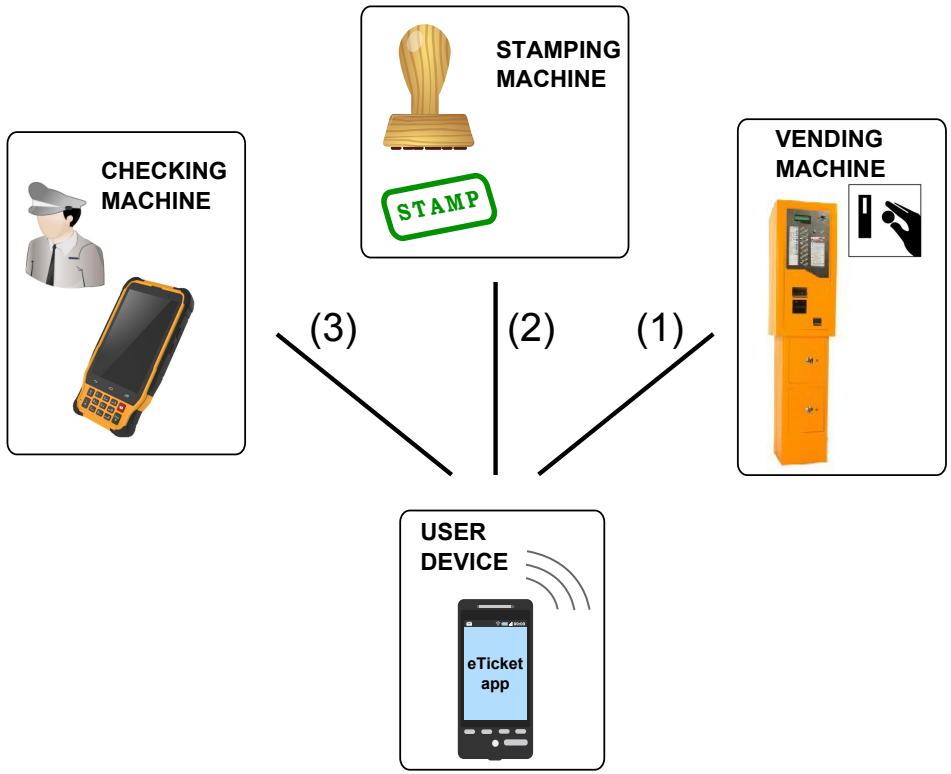
b) *(offline)* via NFC with a vending machine

(2) **E-ticket stamping:**

a) *(offline)* via NFC with a stamping machine

(3) **E-ticket validation**

a) *(offline)* via NFC with a checking machine



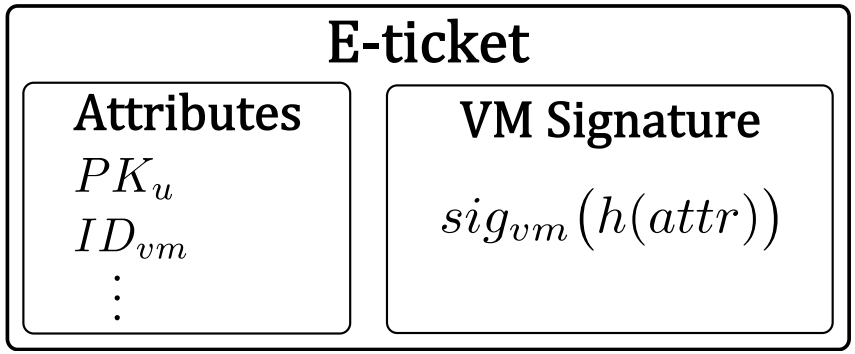
→ The processes are implemented through corresponding protocols (see further)

Main Requirements

1. Open source components
2. Offline stamping and checking (as opposed to vending)
3. Ticket unforgeability
4. Protection from replay attacks
5. Ticket unclonability
6. Double spending prevention
7. Timing (especially for stamping and checking)

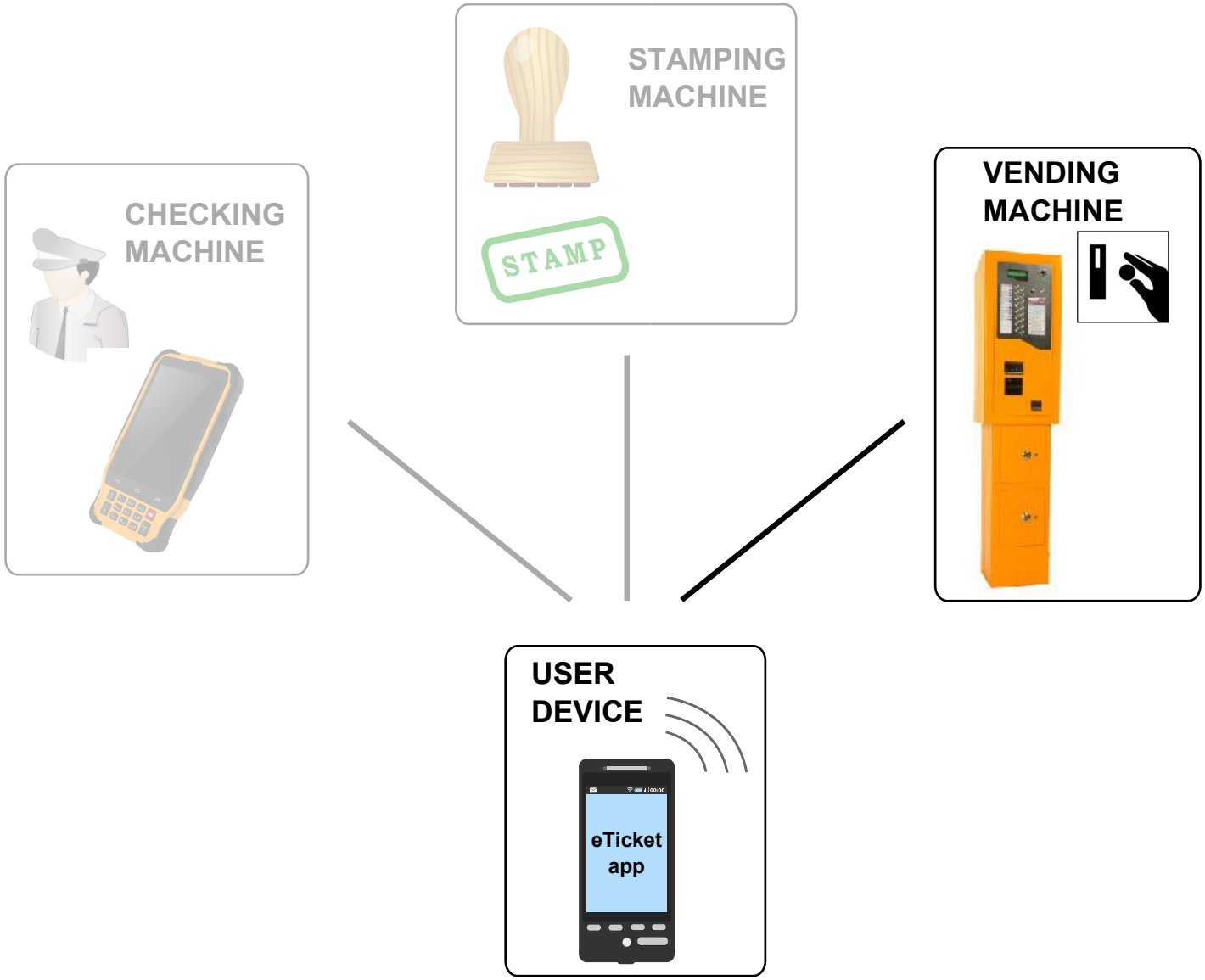
An E-ticket

- Essentially is a digital token
- Describes the acquired travel permission
- E-ticket is bound to the user's public key PK_u
- Different ticket types are supported through attributes
  - time-bounded (e.g., hourly tickets)
  - single ride
  - and many more...



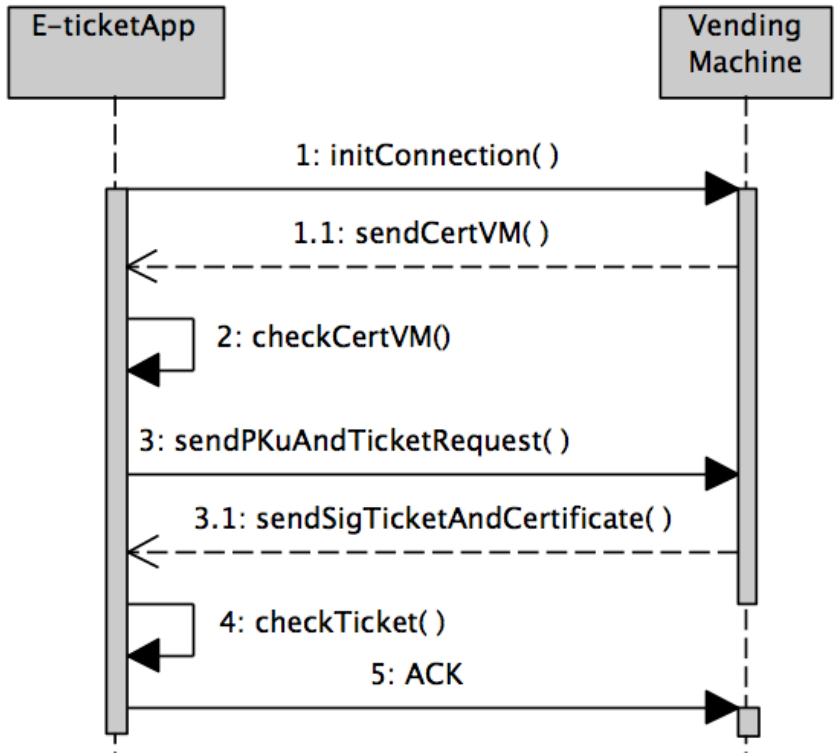
Attributes
PK_u
ID_{vm}
Serial number
Time limit
Zone
Price
...

Protocols: Vending

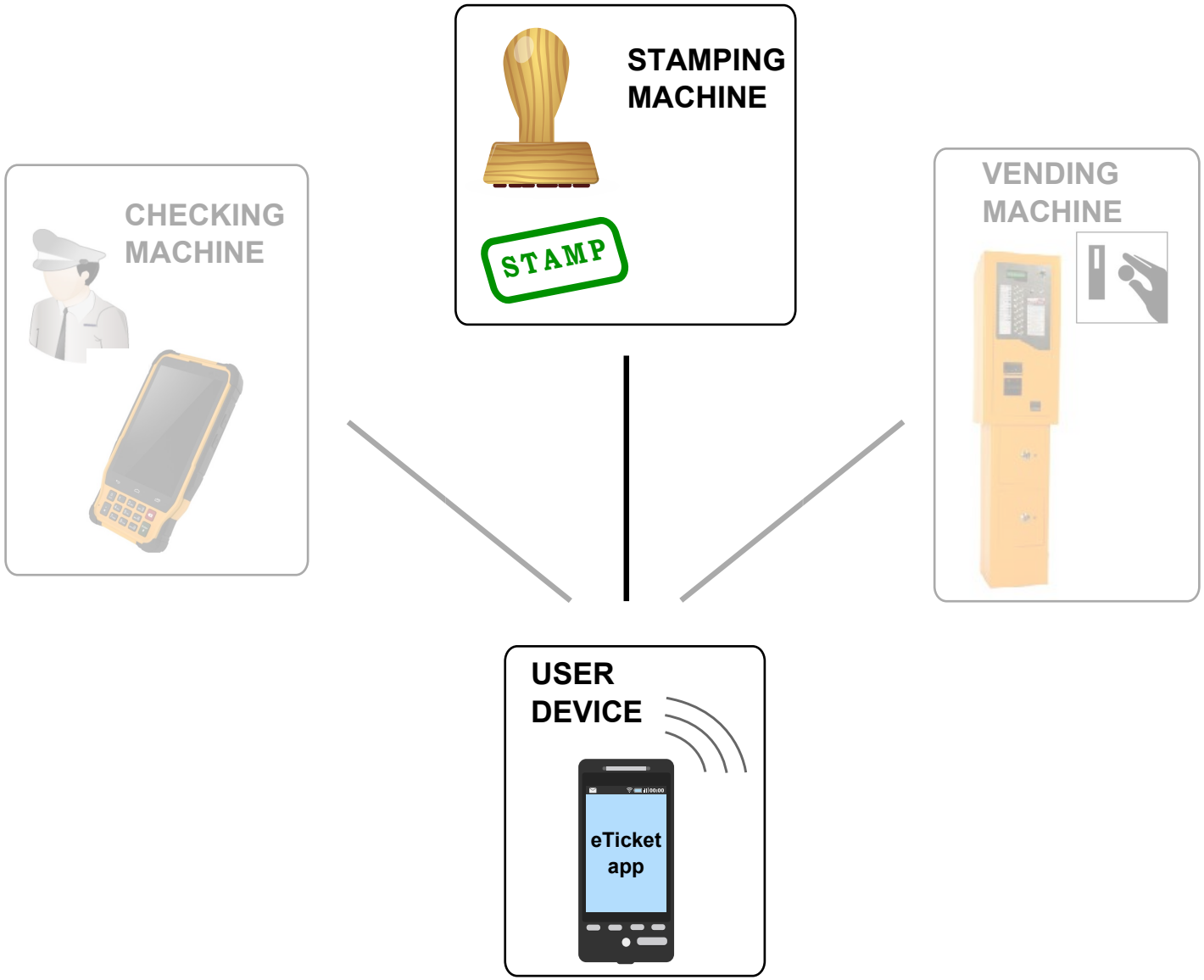


Protocols: Vending

- Actors: **E-ticket app (EA)** and a **Vending machine (VM)**
- Either online or offline
- EA requests a certain ticket type and sends user public key PK_u
- After payment, VM issues an e-ticket binding it to PK_u
- EA verifies the e-ticket

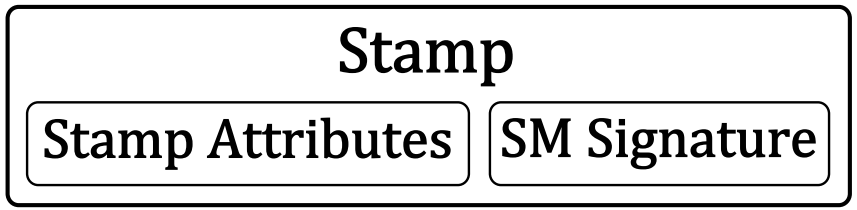


Protocols: Stamping



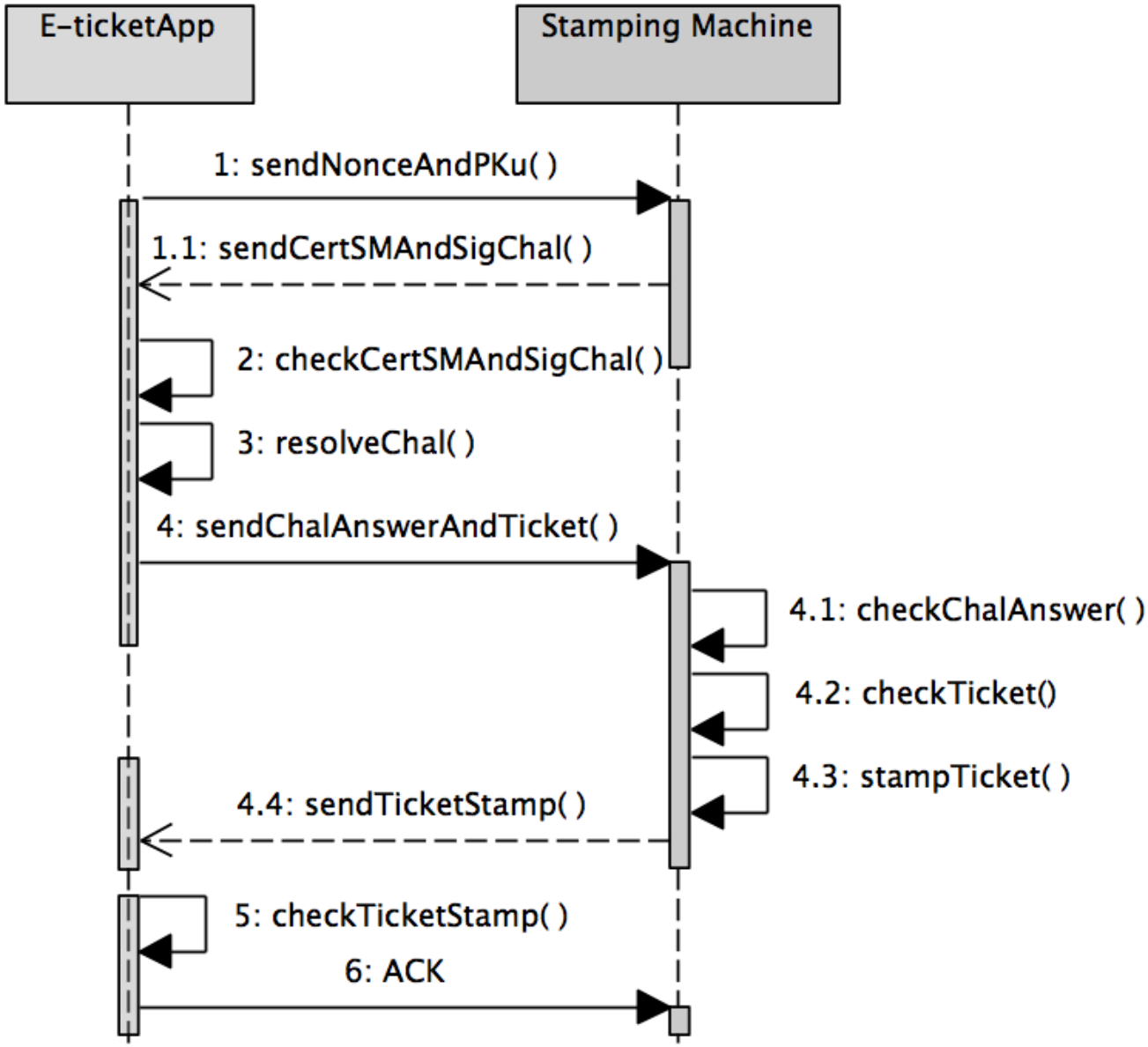
Protocols: Stamping

- Actors: **E-ticket app (EA)** a **Stamping machine (SM)**
- Offline
- Stamping essentially activates the e-ticket for a ride
- As a result, a stamp is obtained

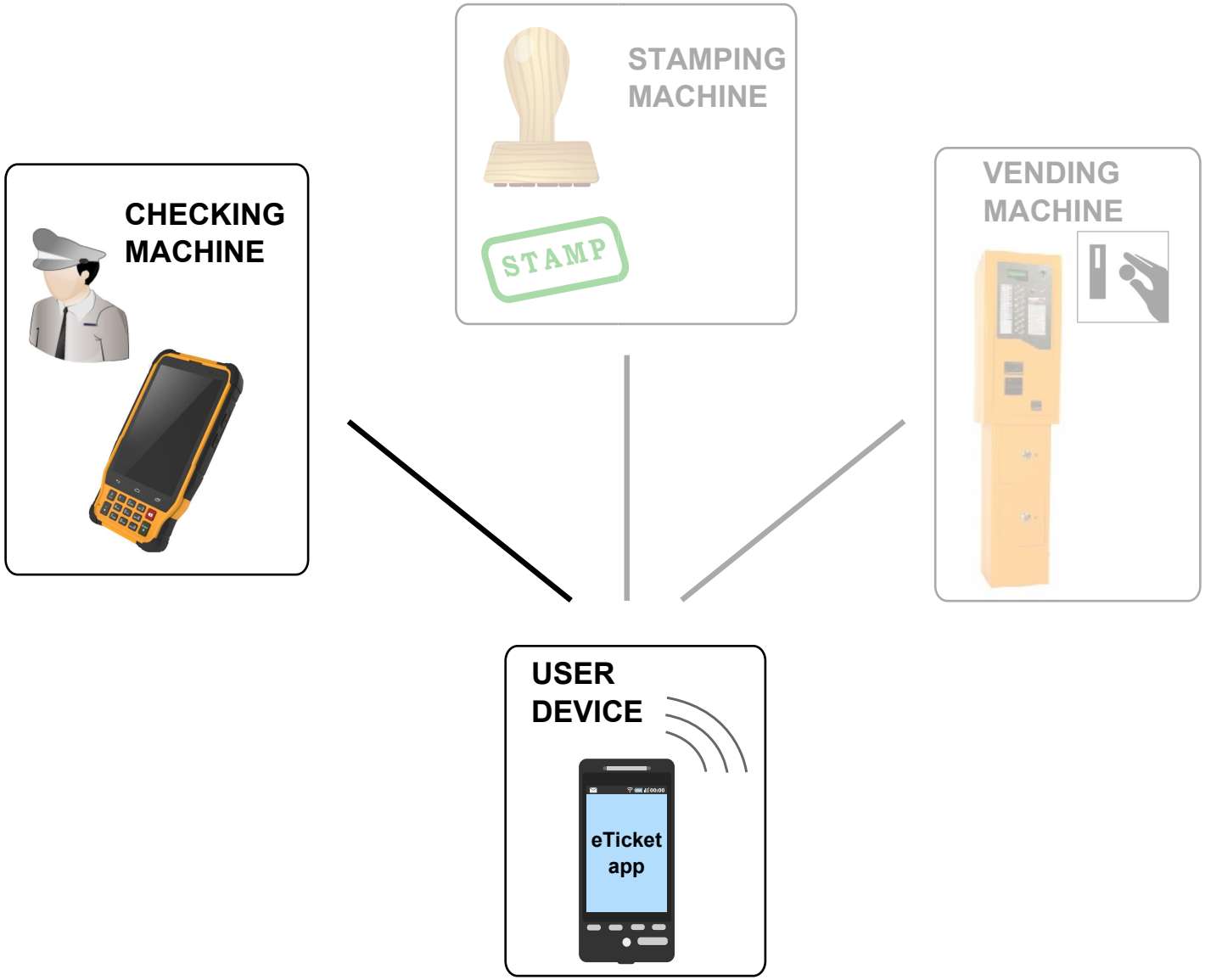


Attributes	**Description**
Serial #	e-ticket number
Timestamp	stamping time
ID_{st}	station ID
ID_{sm}	SM ID.

Protocols: Stamping



Protocols: Checking



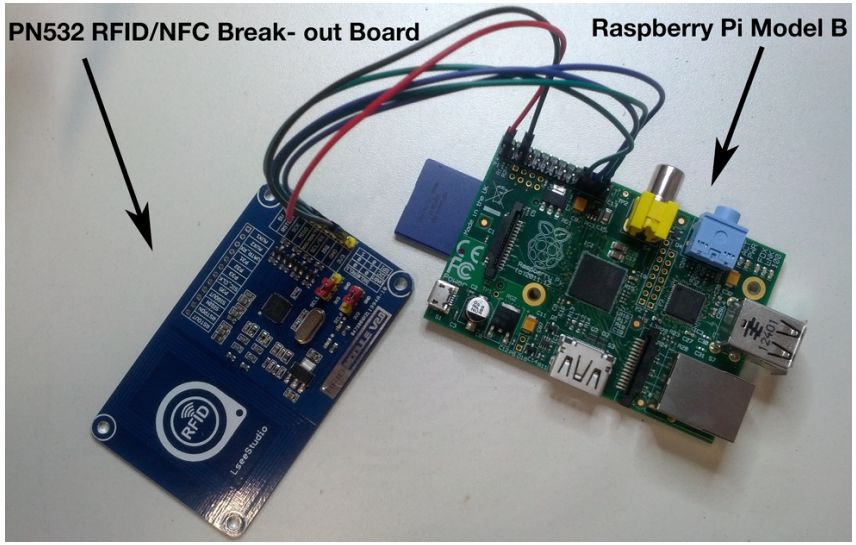
Protocols: Checking

- Performed between an **E-ticket App** and a **Checking machine**
- Follows similar pattern as the stamping protocol
- Up to the point where the previously obtained stamp is checked:
  - the e-ticket must be in *stamped* and *valid* state

Outline

Practical Evaluation

- Terminal part:
  - NFC front-end: PN532 RFID/NFC Breakout Board
  - Controller and logic: Raspberry Pi Model b
- Customer device:
  - Samsung Galaxy Nexus GT-I9250
  - Android 4.4 OS



Performance Figures and Demo

Protocol	**Execution time**	
	RSA-1024	RSA-2048
Vending	0.09 s	0.12 s
Stamping	3.85 s	4.65 s
Checking	3.33 s	4.23 s

- And now a short demo is going to be presented

Conclusion

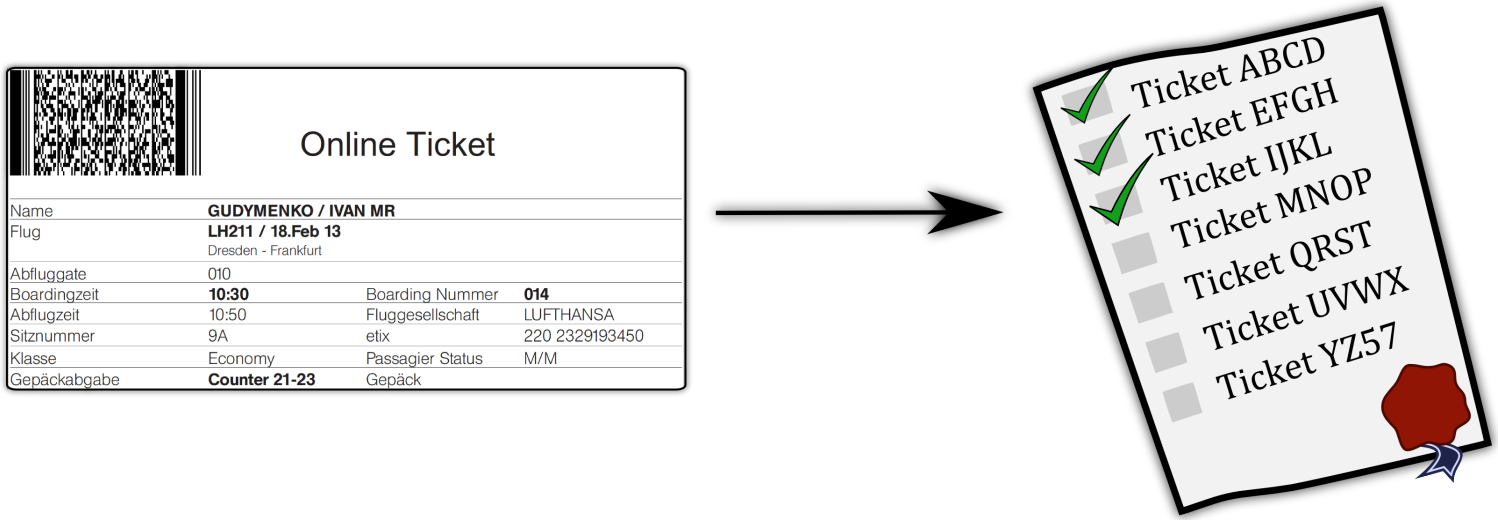
- The designed e-ticketing system has been presented
- It is based on open source components
- The first results of practical evaluation are feasible

Thank you very much for your attention! Do you have any questions/comments/suggestions?

Back up slides

What an E-Ticket is NOT

- A widely used “online ticket” (air transport, etc.)
- Pointing to the respective entry in the back-end DB



Online Ticket

Name	**GUDYMENKO / IVAN MR**		
Flug	**LH211 / 18.Feb 13** Dresden - Frankfurt		
Abfluggate	010		
Boardingzeit	**10:30**	Boarding Nummer	**014**
Abflugzeit	10:50	Fluggesellschaft	LUFTHANSA
Sitznummer	9A	etix	220 2329193450
Klasse	Economy	Passagier Status	M/M
Gepäckabgabe	**Counter 21-23**	Gepäck	

Adopted Attacker Model

(1) **External attackers**

a) An observing attacker

b) A modifying attacker (spoofing, masquerading)

(2) **Internal attackers**

a) A user trying to forge/clone an e-ticket

b) Vending machine issuing invalid e-tickets

c) Stamping machine providing an incorrect stamp

d) A conductor framing the user as having an invalid e-ticket

Protocols: Stamping (Detailed)

E-ticket app (EA)		**Stamping machine (SM)**
(PK_u, SK_u), $eticket$		$cert_{sm}$
Generate random r_e	$\xrightarrow{r_e, PK_u}$	
		Generate random r_{sm}
		Create challenge: $chal \leftarrow (r_e \| \| r_{sm})$
		Encrypt $chal$ under PK_u: $ce \leftarrow E_{PK_u}(chal)$
		Sign ce using $cert_{sm}$: $csign \leftarrow Sign_{cert_{sm}}(ce)$
	$\xleftarrow{ce, csign, cert_{sm}}$	
Check $cert_{sm}$, check $csign$		
Decrypt ce, extract r'_{sm}		
Compute the answer: $ans \leftarrow h(r'_{sm})$		
	$\xrightarrow{ans, eticket}$	
		Check ans, check if $eticket$ is bound to PK_u
		Create a signed stamp for the $eticket$:
		$(stamp, ss \leftarrow Sign_{cert_{sm}}(stamp))$
	$\xleftarrow{stamp, ss}$	
Verify $stamp$ and the signature ss		