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

Ivan Gudymenko
TU Dresden
01062 Dresden
Saxony, Germany
ivan.gudymenko@mail-box.tu-dresden.de

Felipe Sousa
UFCG
Rua Aprígio Veloso, 882 - Bodocongó
Campina Grande - PB, Brazil
felipe.silva@ccc.ufcg.edu.br

Stefan Köpsell
TU Dresden
01062 Dresden
Saxony, Germany
stefan.koepsell@tu-dresden.de

ABSTRACT

E-ticketing systems for public transportation (ESPT) are an integral part of intelligent transportation systems (ITS) which shape the urban environment of the future. The wide deployment of ESPT around the world has proven its success and demonstrated large potential. However, till now the majority of such systems being in operation adhere to the specific standards which are often closed. In this paper, we aim at suggesting an open and secure e-ticketing architecture which provides an alternative to the conventional paper-based ticketing and is customizable for different tariff schemes. Our solution is based on NFC which should facilitate its interoperability with value-added services (from the third parties as well) and pave the way to its convergence with other applications comprising an Internet of Things (IoT) ecosystem in the urban environment.

The suggested architecture covers the main processes specific to the ESPT scenario and provides for e-ticket unclonability, unforgeability as well as for protection from tapping and replay attacks. Finally, the results of practical validation using real hardware are presented to demonstrate the real-world pertinence of our solution.

Categories and Subject Descriptors

Domain-specific security and privacy architectures, Secure online transactions, Distributed systems security.

General Terms

Systems design

Keywords

E-ticketing, NFC, Security, Open Source

1. INTRODUCTION

The ever growing trend of substituting analog systems by their digital counterparts has had a profound impact on a complex urban ecosystem. Intelligent transportation systems becoming an integral part of any modern megalopolis are one of the consequences of such transition. In many cities around the world, the concept of the so-called electronic ticketing (e-ticketing) is being extensively used for issuing travel permissions which may eventually result in conventional paper-based tickets being completely phased out already in the nearest future. Opal Card in Sydney [8], Oyster Card in London [7], Touch & Travel in Germany [9] are all the examples of how well the e-ticketing has been accepted both by customers and public transportation companies.

However, the vast majority of such e-ticketing systems for public transportation (ESPT) deployed around the world adhere to closed specifications and are subject to different standards (which are for the most part closed as well). In this paper, we are aiming at suggesting a simple yet performant ESPT model which is based on open source components and is, therefore, by default extensible and open to public review. The proposed system architecture adheres to the so-called "honor-based" system for public transportation widely deployed in Germany. That is, the customer does not have to prove the possession of a travel permission every time using the transportation service (e.g., to pass through a turnstile) but rather be prepared to present a valid travel credential if being checked by a conductor.

The paper is organized as follows. The use case scenario of ESPT is described in Section 2 with the core requirements following in Section 3. The adopted attacker model is discussed in Section 4. The suggested solution is presented in Section 5 and evaluated in Section 6. The related work is discussed in Section 7. Section 8 concludes the paper.

Urb-IoT '14 October 27 - 28 2014, Rome, Italy

ACM 978-1-4503-2966-8/14/10...$15.00
http://dx.doi.org/10.1145/2666681.2666687

2. TARGET USE CASE SCENARIO

2.1 Core Processes

Our system provides a digital alternative to a conventional paper-based ticketing in public transportation. Therefore, the following processes have to be considered: *(1)* e-ticket acquisition, *(2)* e-ticket stamping, and *(3)* e-ticket checking.

(1) E-ticket acquisition *(online):* describes the process of a customer acquiring (i.e., buying) an e-ticket either via a Web interface (ticket issuing server, hence requiring online connection to the back-end) or from a local vending machine (via the NFC interface). It

is assumed that the obtained e-ticket must be further activated (i.e. "stamped") before entering the public transportation system.

(2) E-ticket stamping *(offline):* refers to activating the e-ticket making it valid for a current ride and ready for possible inspection. Stamping is performed via NFC by choosing the appropriate e-ticket from e-ticket manager app and then holding the smart phone in the vicinity of a dedicated area of stamping machine. Due to their wide distribution (e.g., several machines at every station), it would be too costly to assume that they can maintain constant connection to the back-end. Therefore, our system supports offline stamping.

(3) E-ticket validation *(offline):* captures the event of a conductor checking the validity of an e-ticket (that is, checking if the e-ticket was stamped correctly and is still valid). Validation is performed offline via NFC as during stamping. Similarly to the stamping case, it would be too costly to assume that every conductor device can constantly maintain online connection to the back-end at each point of the route (e.g., including the tunnels, underground and other areas with weak or no network coverage).

2.2 E-ticket States

According to the aforementioned processes, the e-ticket can have the following states: *(1) acquired* after having been obtained from a vending machine (after process 1), and *(2) stamped* after interacting with a stamping machine (after process 2). The latter state can be further divided into *(a) valid* (that is, the e-ticket is stamped and valid in the current context) and *(b) invalid* (the e-ticket is stamped but not valid in the current context, e.g., it is used in another zone or the time since the first ride has been exceeded, etc.)

2.3 Main Actors

Corresponding to the processes described above, the following entities (main actors) are considered: *(a)* a customer (acting via an e-ticketing app installed on his/her NFC-enabled smart phone), *(b)* a vending machine (remote server or a local e-ticket vending kiosk), *(c)* a stamping machine, and *(d)* a checking machine (a conductor's device performing e-ticket inspection). Main system players are summarized in Figure 1 for clarity. Stamping machines are situated at each station in order to provide for locality (that is, issuing such attributes as station ID during stamping, etc.). Conductors possess mobile checking devices allowing them to check the validity of e-tickets.

3. SYSTEM REQUIREMENTS

The aforementioned use case scenario essentially defines a set of functional requirements. Additionally, several requirements with respect to efficiency and security must be addressed. The resultant requirements set considered within this paper is summarized in Table 1.

4. ADOPTED ATTACKER MODEL

The set of possible attackers with respect to the target system can be roughly divided into *(1)* external attackers (referring to the entities exogenous to the system) and *(2)* internal adversaries (the entities which are involved into system processes). Within the first category, the following attacker types can be distinguished: *(a)* an observing attacker tapping the communication between system entities (actors) and *(b)* a modifying adversary performing a spoofing attack (i.e. essentially masquerading itself as a legitimate system actor). Among the internal adversaries, the possible attackers can be further categorized as follows: *(a)* a user trying to forge/clone an e-ticket, *(b)* a vending machine issuing the invalid ticket/a stamping machine providing the incorrect stamp, *(c)* a conductor framing the user as having an invalid e-ticket.

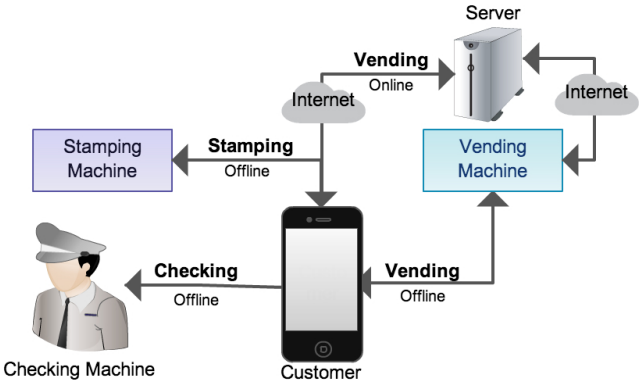

Figure 1: Main actors of the e-ticketing system.

Table 1: 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)

5. SYSTEM DESCRIPTION

5.1 An E-ticket

Within this paper, it is implied that an e-ticket is a digital alternative to a conventional paper ticket. E-ticket is essentially a digital token describing the acquired travel permission bound to a concrete user. The binding is performed using the user's public key PK_u (see Section 5.2 for details).

Our system provides support for many different types of e-tickets by using attributes (see Table 2). This enables for the implementation of various tariff schemes providing each transport authority with the opportunity of customize its service as required. Without loss of generality, however, it can be stated that the two most popular types of e-tickets are: time-bounded tickets (e.g. hourly tickets) and single ride tickets which are considered in our paper as an example.

In general, each e-ticket is comprised of *(1)* attributes (including PK_u) and *(2)* a vending machine's signature over the hash of the attributes (see Figure 2). The set of attributes defines the type of an e-ticket as well as the validity

conditions, see Table 2. The signature ensures the integrity of an e-ticket. The certificate of a vending machine, which is sent to the user device along with the actual e-ticket, must be signed by the root certificate of a transport authority (that is known to the e-ticketing app).

Table 2: E-ticket attributes.

Attribute	Description
PK_u	public key of the customer to which an e-ticket is bound
ID_{vm}	the ID of the vending machine that issued an e-ticket
Serial number	unique number of the e-ticket
Time limit	validity period of the e-ticket
Zone	a region for which the e-ticket is valid
Price	price of the e-ticket

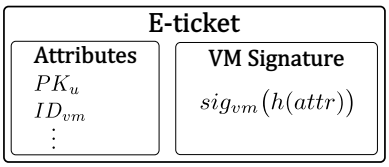

Figure 2: E-ticket structure.

5.2 Protocols

The protocols corresponding to the main processes described in Section 2 are presented in detail below. The notations used during the description of each protocol are summarized in Table 3 for clarity.

Table 3: A summary of the notations used.

Notation	Meaning
(PK_u, SK_u)	user's public/private (secret) key pair;
$cert_{vm}$	certificate of a vending machine;
$cert_{sm}$	certificate of a stamping machine;
$cert_{cm}$	certificate of a checking machine;
nonce	a number used once;
chal	challenge sent to the e-ticket

5.2.1 Vending

In order to acquire an e-ticket, a user triggers the respective action in the e-ticketing app. Then, in case the purchase is performed online, a secure connection to the online vending service is established using the certificate of a vending machine, $cert_{vm}$. The latter is checked using the root certificate of a transport authority which is stored inside the e-ticketing app. Afterwards, the following interactions between the e-ticketing app and the server are performed:

1. The user app sends the request for a certain type of an e-ticket (which can be resolved to the respective attributes, see Table 2) together with the user's public key PK_u.
2. Having agreed on payment details (out of scope of this paper), the vending service selects a particular e-ticket (with the corresponding attributes), binds it to PK_u, signs the result and sends it back to the user.
3. The user app checks the well-formedness of the obtained e-ticket including the verification of the vending machine's signature and sends an acknowledgment.

Main steps of the vending process are depicted in Figure 3.

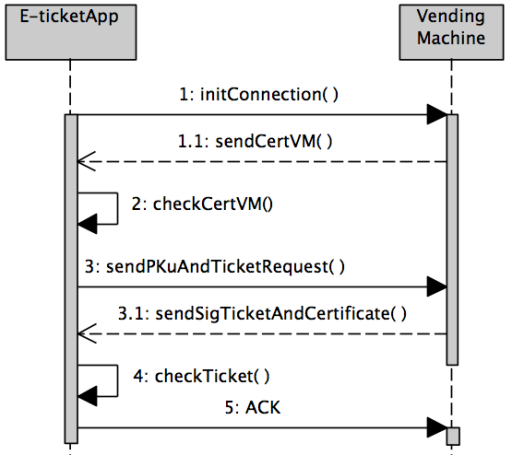

Figure 3: E-ticket vending.

5.2.2 Stamping

Stamping is performed in order to activate the e-ticket and to make it valid for the particular ride (that is, essentially transforming the e-ticket to the stamped state, see Section 2.2). The transformation to the stamped state is performed via local interaction with a stamping machine. As a result of the stamping protocol, a user obtains a stamp together with the respective signature of the stamping machine over the stamp's hash (see Figure 4).

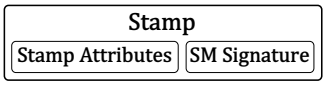

Figure 4: Stamp structure.

The main attributes included into a stamp are summarized in Table 4.

Table 4: Stamp attributes.

Attribute	Description
Serial number	serial number of the e-ticket
Timestamp	date and time of stamping
ID_{st}	station ID
ID_{sm}	stamping machine ID.

Stamping essentially encompasses the following steps:

1. The e-ticketing app (EA) generates a nonce r_e and sends it to the stamping machine (SM) along with the user's public key PK_u.
2. SM creates a challenge, *chal*, by generating another nonce r_{sm} and appending it to the one received from the e-ticketing app: $chal \leftarrow (r_e||r_{sm})$. The challenge is then encrypted with PK_u and subsequently signed by SM. The result is then sent to EA along with the certificate of SM, $cert_{sm}$.
3. EA checks the validity of $cert_{sm}$. If it is invalid, EA aborts. Otherwise, the signature on *chal* is checked.

If it is valid, *chal* is decrypted using the user's private key SK_u, the nonce r_{sm} generated by SM is extracted from *chal* and its hash is sent back to SM together with the e-ticket to be stamped.

4. SM checks if the EA's answer was correct. Then it checks if the received e-ticket is indeed bound to the PK_u exchanged within the first step. Afterwards, SM stamps the e-ticket by adding the respective stamping attributes (see Table 4). Finally, the signed stamp is sent to EA.
5. EA checks the signature and well-formedness of the stamp and sends an acknowledgment back to SM.

The stamping protocol is depicted in Figure 5.

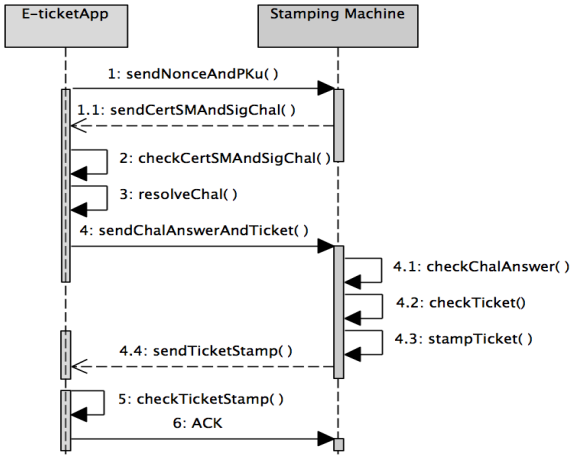

Figure 5: The stamping protocol.

5.2.3 *Checking*

Similarly to stamping, checking is performed locally by a conductor using a checking machine (CM). Checking protocol follows the same pattern as the stamping one right until the step when the EA answer is checked, see Figure 5. After this step, the actions specific to the checking protocol are performed, namely the stamp and the e-ticket attributes are inspected. More specifically, it is checked if the e-ticket is in a *stamped* and *valid state* (see Section 2.2). The checking protocol is depicted in Figure 6.

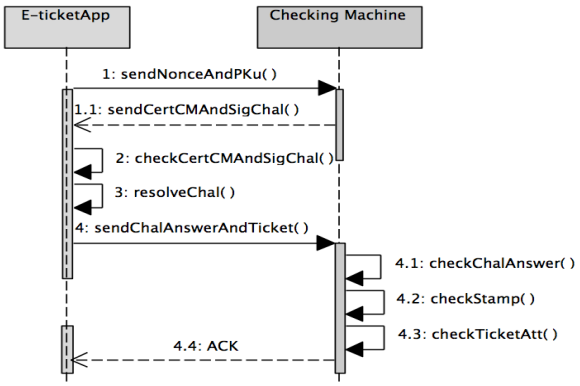

Figure 6: The checking protocol.

6. EVALUATION AND DISCUSSION

6.1 Satisfying the Requirements

The suggested e-ticketing system for public transportation satisfies the main requirements posed to it, see Table 1. It by design allows for open source development (see the open source components used for a prototype, Section 6.2). Stamping and checking are performed offline. The ticket can not be forged, since it would imply that the signature of the vending machine could be forged as well. Should an observing attacker (see Section 4) try to replay the tapped message to the stamping or vending machine, no gain would be achieved due to *(1)* nonce exchange during the protocol session (session binding) and *(2)* check of the knowledge of the private key SK_u (which an attacker does not have) corresponding to the public key to which the e-ticket is bound. To provide for e-ticket unclonability, the following measures can be taken. Firstly, a common assumption in this case is that the private key SK_u resides in a protected memory area (even possibly tamper-resistant in case of a secure element is used, for example) and therefore, can not be extracted and copied. Stamping and checking in our system require the proof of knowledge of the corresponding private key. Therefore, the e-ticket in essence can not be used on another device, since the corresponding private key can not be transferred together with the e-ticket. Moreover, in case the e-ticket purchase had been an identifying transaction, the act of cloning can be detected by analyzing the logs from stamping and checking machines (e.g., the same ticket appeared in two different places roughly at the same time) and the corresponding action towards the user can be taken. The double spending prevention can be ensured by having the e-ticketing app enforce the secure transition of the e-ticket's state from acquired to stamped (see Section 2.2), that is to say, deleting the unstamped e-ticket after it has been stamped. Moreover, if stamping and checking machines could be interconnected in a non-real time fashion and regularly synchronize their logs, then the act of double spending could be detected and the serial number of the e-ticket spent twice can be added to the respective blacklists. Note, that stamping and checking would still be performed offline and the user experience would not be negatively affected by network delays, etc. If the e-ticket acquisition had been an identifying transaction, then an appropriate action could be additionally imposed on the user. Lastly, in order to assess the practical performance of our system, the respective prototype was implemented, see the discussion in Section 6.2.

Our system covers all attacker types introduced in Section 4. An observing attacker (type *1-(a)*) is defended against by securing the underlying connection between the user device and other system actors. Since a certificate-based authentication is used, a spoofing adversary (type *1-(b)*) would not be able to successfully mount an attack either. In case of internal attackers, a malicious user (type *2-(a)*) can be mitigated by using the measures against forgery and cloning discussed above. In order to defend an honest user against a malicious vending or stamping machine (attacker *2-(b)*), the attribute inspection together with well-formedness and signature check is performed by the e-ticketing app. In case a conductor trying to frame a user (attacker *2-(c)*), an optional online check (e.g., a special checking service running in the back-end) can be used to prove (e.g. to security personnel) the validity of the e-ticket.

6.2 Practical Evaluation

In order to practically evaluate our solution, the prototype was created covering all main actors in the system (see Fig-

ure 1). Namely, *a customer's device* is represented by a Samsung Galaxy Nexus GT-I9250 smart phone running Android 4.4. An e-ticketing app (see Figure 7) is installed on the user device to manage e-tickets, acquire the new ones, and interact with the other actors in a system. *A vending machine* was implemented as a vending service running on a server. *A stamping machine* as well as *a conductor device* were implemented as an NFC terminal. The latter consists of *(1)* a Raspberry Pi Model B (512 MB RAM) with Raspbian OS (version 2014) and *(2)* the NFC front-end represented by Adafuit PN532 RFID/NFC Breakout Board connected to the Raspberry Pi via SPI (serial peripheral interface), see Figure 8. The terminal side was running an open source library Libnfc 1.7.0 [4] for NFC communication. To make our solution portable and platform independent, the NFC channel was implemented on top using the so-called inverse reader mode, firstly mentioned by [5] and further enhanced at our chair. The source code of our project is available as open source, see [6].

RSA (with key length of 1024 and 2048) was chosen as the underlying cryptographic primitive in our system, namely for a user key pair (for e-ticket binding) as well as for the certificates. Interestingly enough, Elliptic Curves (with key lengths of 160 and 224 bits) turned out to be one order *slower* than RSA (with the corresponding bit lengths of 1024 and 2048 bits respectively), both for signing and signature checking (that is, for effectively decrypting and encrypting). We assume that the reason for this may be *(1)* the poor implementation of ECC in Bouncy Castle cryptographic library we used and/or *(2)* the underlying hardware optimization being available for RSA and not available for ECC on a user device. For hashing, a widely used SHA-1 was used. The performance of each protocol is summarized in Table 5.

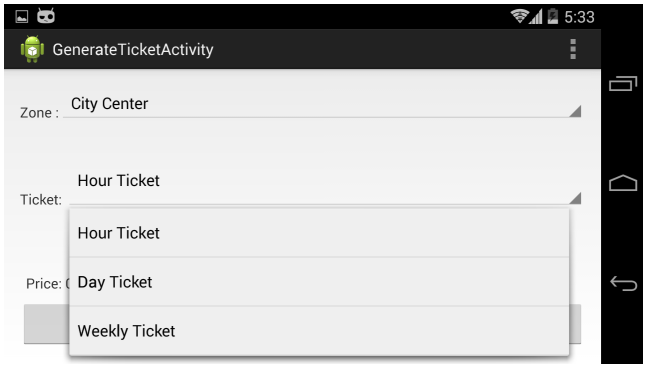

Figure 7: The implemented e-ticketing app.

Table 5: Performance of each protocol

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

As it can be seen from Table 5, the vending protocol is the fastest, since in our prototype it was carried out via a Web interface. Stamping and vending are considerably slower since they were run via NFC. The reason for this is that our own implementation of the underlying platform-independent interactive NFC communication is relatively slow. Basically,

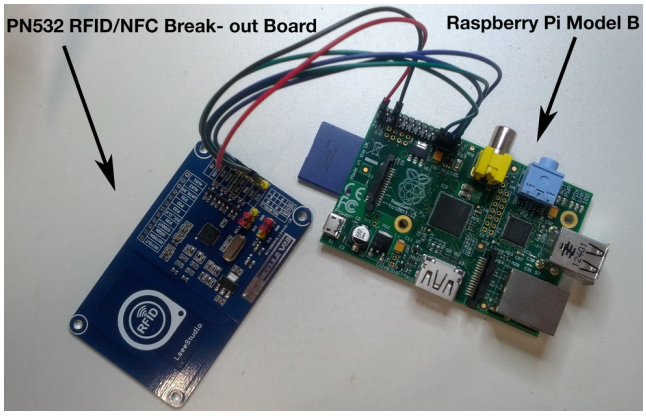

Figure 8: Terminal hardware.

the execution times of all three protocols are comparable and the difference is largely defined by the communication channel (that is, the socket based communication with the server vs. interactive NFC communication). With the improvement of the underlying interactive NFC communication, the protocol execution time is going to be substantially enhanced. Even in the current prototype, the execution time for stamping and checking is acceptable, let alone the one of vending.

The last remark should be made with respect to the checking protocol. Depending on the concrete tariff scheme in use, the analysis of the attributes contained in the stamp and in the e-ticket may take different amount of time. For example, the conductor may "manually" meet the decision on the validity of e-ticket being inspected by taking into account the notification from his/her checking device that the e-ticket and stamp signature are valid (that is the e-ticket is in the stamped state, see Section 2.2) and examining the extracted e-ticket attributes. An alternative way would be to fully automate the conductor application to have it output "valid" or "invalid" based on context information, such as the current date and time, location, etc.

6.3 Privacy Implications

Binding an e-ticket to the customer's public key is an important security measure which, however, may introduce an undesired tracking parameter by this raising additional concerns over privacy. A straightforward measure to address this would be using a new key pair (PK_u, SK_u) for each e-ticket. Modern smart phones possess enough resources to maintain the resultant key pool. Moreover, after the e-ticket has been used, the corresponding key pair together with the e-ticket can be deleted by the e-ticketing app for memory management. A more serious privacy concern, however, is raised by the actual buying procedure of an e-ticket. In case of an online purchase, the vast majority of today's transactions are the identifying ones (e.g., containing a reference to the bank account number and account holder name). Therefore, the acquired e-ticket can be linked to the user and the latter can be tracked and even identified. In case an anonymous pre-paid account was used for a purchase, the e-ticket can still be traced. The most privacy-respecting option for online purchase would then be *(1)* using untraceable anonymous payment systems similar to the e-cash concept [2],

(2) untraceable variants of electronic currency such as bitcoin (e.g., bitcoins with "laundry" systems [1]), or *(3)* paying through a trusted third party. Another obvious but possibly not the most convenient way to obtain an e-ticket in a privacy-preserving manner would be simply paying cash at the local vending machine (that is, offline with respect to the communication between a user device and a vending machine). An important trade-off between privacy (user anonymity and untraceability) and security (especially double spending prevention and unclonability) should be mentioned at this stage. The designer of a concrete e-ticketing system for public transportation must take this trade-off into account and carefully analyze the resulting implications on both system security and customer privacy.

7. RELATED WORK

Saminger *et. al* described an e-ticketing system based on the so-called NFC inverse reader mode in their paper [5]. Their solution was primarily targeted at interoperability between different NFC platforms (hardware) and operating systems (with respect to user devices, i.e. NFC-enabled smart phones). According to the authors, among the three standardized NFC modes (reader/writer, card emulation, and peer-to-peer), the reader/writer mode is the most supported one across different platforms as well as the most lightweight. By inverting the reader/writer mode, the advantages of the classic reader/writer mode can be leveraged together with a further one: bypassing the secure element and by this making the e-ticketing system independent from the manufacturer of the secure element. An important caveat of this approach is, however, that not every commercially available reader device supports the card emulation mode (the terminal side) which is required for the novel inverse reader mode to function (the details are provided in [5]). Moreover, since the approach presented in [5] was rather focused on the novel inverse reader/writer model, the e-ticketing architecture was only superficially described.

The authors of [3] presented a prototype of an e-ticketing system based on NFC. During the evaluation, an NFC-enabled Nokia 6131 phone with a secure element was used as a user device. Unfortunately, the protocols related to e-ticketing (such as vending, checking, etc.) were not presented in detail and thus their properties could not be assessed. However, in [3], the usability tests were performed which were used for providing the recommendations for better acceptance of the developed e-ticketing system in the real-world scenario.

Lastly in [11], a solution was presented allowing to integrate NFC ticketing into an existing infrastructure which conforms to the special standard for interoperable public transport and revenue sharing (known as the core application, VDV-KA, see [10]). Despite providing a solid concept having high pertinence to the real world scenario, the solution is tailor-made for the specific context of VDV-KA standard. This may substantially limit its adoption in other systems for public transportation which are not based on it. Moreover, similarly to [3], the relevant processes were not described in detail but rather implicitly referenced in the complex VDV-KA specification which is not open for a public review (at least, not all parts of it). Furthermore, in [3, 11] the prototypes used for concept evaluation were based on NFC-enabled Nokia phones running Symbian as a user device, whereas in our paper the evaluation was performed on more modern devices running Android.

8. CONCLUSION AND FUTURE WORK

In this paper, an architecture of an e-ticketing system for public transportation (ESPT) based on NFC which supports open source development was presented. The suggested architecture satisfies the major requirements posed to such kind of a system. The main protocols generic to ESPT (vending, stamping, checking) were discussed in detail. Our concept is backed up with a practical implementation based on open source components. For future work, it is planned to perform usability tests as it was done in [3]. Moreover, the countermeasures against relay attacks are going to be additionally researched on.

Acknowledgments

The authors would like to express their gratitude to Manuel Weißbach for his work on the interactive NFC interface. The first author is financially supported by the Free State of Saxony and the European Social Fund (ESF) under grant number 100097574. The second author carried out his work within the NoPa project "TrueGrid".

9. REFERENCES

[1] BitLaundry.com. BitLaundry - For all your Bitcoin washing needs. `http://app.bitlaundry.com/about`. Accessed on 18.06.2014.

[2] D. Chaum et al. Untraceable Electronic Cash. In *Proceedings of CRYPTO '88*, CRYPTO '88, pages 319–327, London, UK, UK, 1990. Springer-Verlag.

[3] S. L. Ghìron et al. NFC Ticketing: A Prototype and Usability Test of an NFC-Based Virtual Ticketing Application. In *Proceedings of the First International Workshop on NFC*, NFC '09, pages 45–50, Washington, DC, USA, 2009. IEEE.

[4] libnfc community. Libnfc: Official wiki. `http://nfc-tools.org/index.php`. Accessed on 12.06.2014.

[5] C. Saminger et al. An NFC Ticketing System with A New Approach of An Inverse Reader Mode. In *5th International Workshop on NFC*, pages 1–5, Feb 2013.

[6] F. Sousa, I. Gudymenko, and S. Köpsell. A Simple and Secure E-Ticketing System for Intelligent Public Transportation based on NFC: A Source Code. `https://bitbucket.org/dud-team/ssticket`, September 2014.

[7] Transport for London. Oyster Online. `https://oyster.tfl.gov.uk/oyster/entry.do`, 2012. Accessed on 30.10.2012.

[8] Transport for NSW. Opal Card. `http://www.transport.nsw.gov.au/content/opal-card`, 2014. Accessed on 10.06.2014.

[9] Transport for NSW. Touch & Travel. `http://www.touchandtravel.de/`, 2014. Accessed on 10.06.2014.

[10] VDV-KA KG. VDV-Kernapplikation ist der elektronische Ticket-Standard. `http://mitglieder.vdv.de/wir_ueber_uns/vdv_projekte/vdv_kernapplikation_efm.html`. Accessed on 18.06.14.

[11] R. Widmann et al. System Integration of NFC Ticketing into an Existing Public Transport Infrastructure. In *4th International Workshop on NFC*, pages 13–18, March 2012.