Downloaded from
www.mroland.at

Comparison of the usability and security of NFC's different operating modes in mobile devices

Michael Roland, Josef Langer

NFC Research Lab Hagenberg, University of Applied Sciences Upper Austria
Softwarepark 11, 4232 Hagenberg, Austria
E-mail: michael.roland@fh-hagenberg.at

Abstract: This paper highlights the benefits and drawbacks of NFC's different operating modes with regard to their usability and security. Based on an analysis of both traditional and new communication concepts for mobile NFC devices, their current availability and, specifically, the features to provide security are evaluated. The result of this evaluation is a comparison between the availability, the usability and the security of NFC's different operating modes.

Keywords: Near Field Communication, Mobile phone, Operating modes, Security, Availability

Vergleich der Benutzbarkeit und Sicherheit der unterschiedlichen Betriebsmodi von NFC in Mobiltelefonen

Kurzfassung: Dieser Artikel zeigt die Vor- und Nachteil der unterschiedlichen Betriebsmodi von NFC in Bezug auf deren Benutzbarkeit und Sicherheit auf. Ausgehend von einer Analyse herkömmlicher und neuer Kommunikationskonzepte für mobile NFC-Geräte werden deren derzeitige Verfügbarkeit sowie deren aktuelle Sicherheitsfunktionen evaluiert. Das Ergebnis dieser Evaluierung ist ein Vergleich zwischen der Verfügbarkeit, der Benutzbarkeit und der Sicherheit der unterschiedlichen Betriebsmodi von NFC.

Schlüsselwörter: Near Field Communication, Mobiltelefon, Betriebsmodi, Sicherheit, Verfügbarkeit

Received July 25, 2013, accepted September 9, 2013, published online November 7, 2013

1. Introduction

Since the integration of Near Field Communication (NFC) technology into the Android operating system in 2010, more and more smart phones and other mobile devices come equipped with NFC functionality [11]. This increasing number of available NFC devices results in a growing interest in the development of NFC applications. However, developers regularly stumble upon finding the right mode for NFC devices to communicate with each other in their application use-cases.

Mobile NFC devices can operate in three different communication modes: peer-to-peer mode, reader/writer mode, and card emulation mode [7]. Peer-to-peer mode is defined in ISO/IEC 18092, the core standard of NFC technology. This mode is intended for direct communication between two NFC devices. Reader/writer mode and card emulation mode were designed to provide compatibility to various existing contactless smartcard standards like ISO/IEC 14443 and JIS X 6319-4 ("FeliCa"). In reader/writer mode, an NFC device can interact with NFC tags and contactless smartcards. In card emulation mode, an NFC device may be accessed by contactless reader infrastructures as if it were a smartcard.

However, this clear separation between different operating modes for different use-cases becomes more and more blurred. There are two main reasons behind this trend: interoperability and complexity. For instance, there is a widely varying level of support and interoperability of each of NFC's operating modes across different mobile phone platforms. With regard to complexity, there are specific difficulties associated with certain operating modes. For instance, card emulation mode using a secure element requires complex infrastructures to roll-out applica-

This is a post-peer-review, pre-copyedit version of an article published in e & i Elektrotechnik und Informationstechnik (2013), 130/7. The final publication is available at Springer via https://doi.org/10.1007/s00502-013-0157-x

Downloaded from
www.mroland.at

tions. Some modes, like peer-to-peer mode, may require bulky protocol stacks. Besides their usability from a developer's point of view, the different operating modes provide varying levels of security and privacy for stored data and communication. Often, an increased level of security also increases the complexity of developing applications. As a consequence, each operating mode has its advantages and disadvantages with regard to specific applications and use-cases.

2. NFC's Operating Modes

NFC's three operating modes differ in used communication protocols and intended use-cases. For integration into NFC devices, the communication protocols have been harmonized by the NFC Forum in the *NFC Analog* and *NFC Digital Protocol* technical specifications.

2.1 Peer-to-peer Mode

Peer-to-peer mode was originally designed to elevate NFC devices from being pure contactless smartcard readers or pure contactless smartcards to being capable of directly communicating with each other. Therefore, protocol elements from both, reader-side and card-side, were combined into this mode and extended by higher layer protocols specific to peer-to-peer mode.

Fig. 1 outlines the protocol stack of NFC's peer-to-peer mode. The lowest layer is the low-level communication technology (NFC-A for 106 kbps and NFC-F for other data rates) which defines the modulation, coding, framing and device detection procedure.

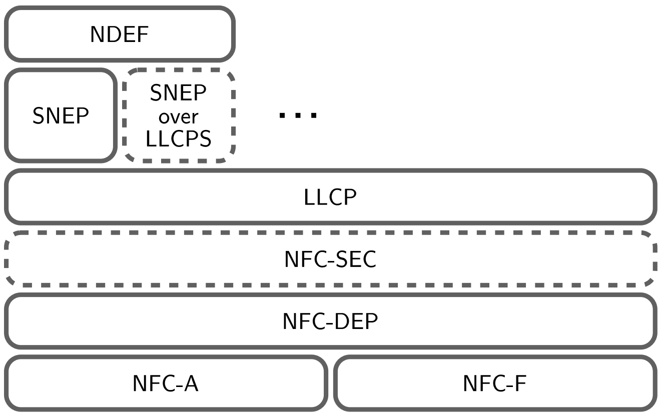


Fig. 1. Protocol stack of NFC's peer-to-peer mode

NFC-A and NFC-F are based on the smartcard protocols ISO/IEC 14443 Type A and JIS X 6319-4 respectively.

The *NFC Data Exchange Protocol* (NFC-DEP) adds device activation and basic data exchange capabilities on top of NFC-A and NFC-F. The layers up to NFC-DEP are standardized in ISO/IEC 18092. In NFC-DEP, one device takes the role of the master that controls the communication (“initiator”) and one device takes the role of the slave (“target”). Moreover, the two devices can communicate in either active mode or passive mode. In active mode, both, initiator and target, alternately generate their own HF carrier signal. In passive mode, the initiator sends data using its own HF signal while the target responds by modulating its load on the initiator's HF carrier signal.

On top of NFC-DEP, the *Logical Link Control Protocol* (LLCP) provides connection-oriented and connection-less end-to-end communication. These communication endpoints can be used by applications and services on NFC devices and permit multiple simultaneous communication channels. A well-known and widely supported service on top of LLCP is the *Simple NDEF Exchange Protocol* (SNEP). It permits easy exchange of NDEF messages between applications. NDEF, the *NFC Data Exchange Format*, allows applications to exchange data regardless of the used communication mode and the used communication technology.

Two protocols exist to provide security for peer-to-peer communication: *NFCIP-1 Security and Services Protocol* (NFC-SEC) and *TLS over NFC LLCP* (LLCPS, cf. Urien [17]). NFC-SEC introduces channel encryption and authentication between NFC-DEP and the LLCP layer while LLCPS provides a TLS (Transport Layer Security) protocol layer similar to HTTPS for LLCP services. However, none of these security protocols are known to be available in current NFC-enabled smart phones.

2.2 Reader/writer Mode

Fig. 2 outlines the protocol stack of NFC's reader/writer mode. In reader/writer mode, NFC devices typically support three low-level communication technologies: NFC-A (based on ISO/IEC 14443 Type A), NFC-B (based on ISO/IEC 14443 Type B) and NFC-F (based on JIS X 6319-4). Some NFC devices also support NFC-V which is based on ISO/IEC 15693. The *ISO Data Exchange Protocol* (ISO-DEP)

Downloaded from
www.mroland.at

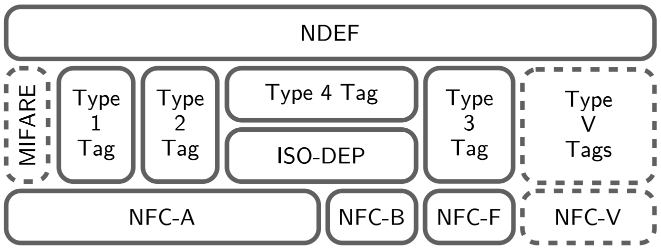


Fig. 2. Protocol stack of NFC's reader/writer mode

layer is equivalent to the common transport protocol defined in ISO/IEC 14443-4.

On top of these communication technologies, four standardized tag platforms (Type 1–4) and several proprietary tag platforms (e.g. MIFARE Classic, Type V) define memory structures for storing NDEF messages on these tags. The tag platforms provide an abstraction layer that allows NFC applications to read and write NDEF messages independently of the actual tag hardware and memory layout.

Besides the abstraction through NDEF, some device platforms also support direct communication with tags by exchanging commands using ISO-DEP or the low-level communication protocols.

2.3 Card Emulation Mode

Fig. 3 outlines the protocol stack of NFC's card emulation mode. Current NFC devices with card emulation capabilities support one or more of the low-level communication protocols NFC-A, NFC-B and NFC-F [10].

Consequently, in card emulation mode, an NFC device can interact with existing contactless smartcard readers. Card emulation is typically associated with the secure element (SE), usually a secure smartcard microchip connected to the device's NFC controller that performs the actual emulation. Besides access through the contactless interface, an SE can also be accessed by applications that run on the NFC device's application processor.

A typical SE uses only one low-level communication protocol. SEs that support NFC-A or NFC-B communicate using APDUs (ISO/IEC 7816-4) on top of ISO-DEP. Some SE that use NFC-A are also capable of emulating NXP's MIFARE Classic for legacy purposes. SEs that support NFC-F usually communicate using Sony's FeliCa protocol.

The SE, just as other modern smartcard microchips, is protected against various kinds of physical and

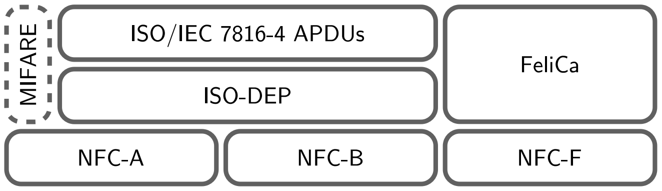


Fig. 3. Protocol stack of NFC's card emulation mode

logical attacks by sensors and counter measures in hardware, and by a robust design process. Chip hardware and software, as well as their design processes, are usually evaluated and certified according to high security standards. Consequently, an SE provides a high security execution environment and data storage. In addition it provides hardware-based support for various cryptographic functions.

Nevertheless, even SEs may be vulnerable to attacks. For instance, the software-based relay attack (cf. Roland et al. [13] and Roland et al. [14]) abuses potentially insecure application processors to gain unauthorized access to a secure element.

3. Availability on Device Platforms

While these modes would provide sufficient means for communication in most NFC application scenarios, not all NFC devices support the full spectrum of these features. Jakl and Roland [6] give a good overview on the supported NFC features in various mobile phone platforms.

3.1 Peer-to-peer Mode

Most current mobile phone platforms (e.g. Android, BlackBerry, Symbian, Windows Phone 8) use LLCP for communication in peer-to-peer mode. Android, BlackBerry and Windows Phone 8 use SNEP on top of LLCP to automatically exchange NDEF messages between devices. The SNEP layer provides interoperability across these device platforms. Though, Android versions before 4.0 did not support SNEP but used a proprietary protocol named *NDEF Push Protocol* (NPP) instead.

On Android and Windows Phone 8, the exchange of NDEF messages using SNEP (or NPP) is the only communication feature supported in peer-to-peer mode. Direct use of LLCP sockets is not possible on these platforms. Even worse, only one NDEF message can be exchanged per NFC interaction. However, there exist concepts to overcome this limita-

Downloaded from
www.mroland.at

tion: OPEN-SNEP [8] permits bi-directional communication over SNEP by shutting down the peer-to-peer link (i.e. by turning off the RF field) between each exchange of NDEF messages.

Besides availability in NFC-enabled mobile phones, peer-to-peer mode is only available in dedicated NFC reader devices (e.g. SCL3711, ACR 122U). Many contactless smartcard readers (e.g. OMNIKEY 5321) do not support this mode at all.

3.2 Reader/writer Mode

All current NFC-enabled mobile phone platforms support some form of reader/writer mode. For example, only a minimum level of support has been implemented in Windows Phone 8: This platform can only interact with NDEF-formatted tags. Moreover, only one NDEF message can be exchanged (either read or written) per interaction with an NFC tag.

However, most platforms do not have such tight limitations and can even read and write multiple NDEF messages within one interaction. Some platforms (e.g. Android and BlackBerry) even have reader/writer mode support beyond access to simple NFC tags. These platforms permit direct communication with tags and contactless smartcards using ISO-DEP or low-level communication protocols (e.g. NFC-A, NFC-B, NFC-F).

Moreover, all current smart phone platforms have some form of automatic processing of NFC tags. I.e. even if no specific application is active, the phone tries to detect NFC tags and triggers actions based on the contents of the NDEF messages (e.g. open a URL in a web browser).

3.3 Card Emulation Mode

Most current NFC-enabled smart phones contain an embedded SE chip or have support for a UICC-based (universal integrated circuit card, “SIM card”) secure element that is connected to the NFC controller through a single wire protocol (SWP) connection.

While these SEs are accessible through either proprietary development interfaces or standardized interfaces like the Open Mobile API, getting applications onto an SE is usually a difficult task. As an SE is a highly trusted and resource-limited environment, not every developer can access it to install applications. Instead, secure elements are managed by a trusted party that assures that applications meet certain security requirements. Thus, only this trusted party knows the keys necessary to manage applications on the SE. In the case of UICC-based SEs, the mobile network operator that provided the UICC is typically in possession of these keys. In the case of an embedded SE, the keys are usually owned by the device manufacturer. As a consequence, different SEs are often managed by different trusted parties.

Another impeding factor besides stringent security requirements and the complex ecosystem is the low memory capacity of SEs. A typical SE has only some hundred kilobytes of memory for applications and data. This memory must be shared among several applications. Thus, the trusted party that manages the SE will prefer some applications (most likely more profitable ones) over others. As a result, the roll-out of applications to SEs is a complicated and costly procedure.

4. New Communication Concepts

NFC's three operating modes result in three communication concepts:

1. In peer-to-peer mode, two NFC devices communicate directly with each other.
2. In reader/writer mode, an NFC device accesses NFC tags and contactless smartcards.
3. In card emulation mode, an NFC device uses its secure element to emulate a contactless smartcard.

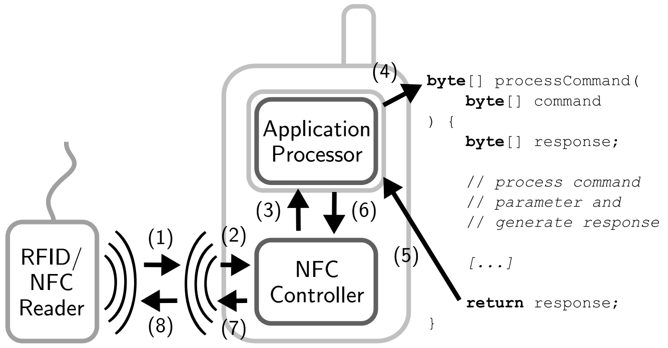
While these traditional scenarios would allow for a broad range of use-cases, the above limitations of current NFC-enabled smart phones and secure element infrastructures caused developers to seek after new communication concepts.

4.1 Software Card Emulation

Software card emulation [10] (also called “soft-SE” [3] or “host card emulation” [18]) is a new approach to card emulation in NFC-enabled smart phones. In this mode, the NFC controller routes the ISO-DEP communication between the contactless interface and an application on the NFC device's application processor instead of passing all the communication to an SE (see Fig. 4). Software card emulation first appeared in the BlackBerry 7 platform. Besides card emulation using an SE, these devices allow applications to emulate NFC tags and contactless smartcards through software that runs on the mobile phone's application processor [2, 9].

On BlackBerry, an application can emulate a contactless smartcard using the ISO-DEP protocol on top of

Downloaded from
www.mroland.at



```
byte[] processCommand(
    byte[] command
) {
    byte[] response;

    // process command
    // parameter and
    // generate response

    [...]

    return response;
}
```

Fig. 4. Software card emulation [10]

either NFC-A or NFC-B. Moreover, the BlackBerry platform permits easy emulation of NFC Forum Type 4 tags on top of ISO-DEP. An application that wants to emulate such a tag simply needs to specify an NDEF message. The phone, then, automatically emulates a Type 4 tag containing that message. Consequently, another NFC device that operates in reader/writer mode could read that NDEF message from the virtual tag and could even write new NDEF messages to the virtual tag that can then be processed by the application. Therefore, the combination of software card emulation and reader/writer mode could be used as a replacement for peer-to-peer mode which is currently not fully supported on some devices.

The main advantage behind software card emulation is that it permits card emulation applications while getting rid of the difficult secure element ecosystem [10]. As a consequence, any developer can develop and roll-out card emulation applications.

However, bypassing the secure element comes at the price of security. Applications executed on a mobile phone's application processor do not benefit from the secure data storage and the trusted execution environment of a secure element. Unless, of course, the application processor itself provides some form of trusted computing technology – which is not the case with most current mobile phones.

Without secure storage, it becomes difficult for card emulation applications to store sensitive data. Moreover, the lack of a trusted execution environment could allow for (intentional) interference by other applications. For instance, recent vulnerabilities of Google Wallet allowed an attacker to recover credit card numbers, account balance, card holder information and even the wallet's PIN code, because, even though Google Wallet has access to a secure element, this data was cached within the app's private data storage in the mobile phone memory [1, 5, 15]. Moreover, current smart phone operating systems are often vulnerable to privilege escalation exploits that permit malicious applications to access code and data of other applications (e.g. Android bug 8219321, mempodroid, Levitator, zergRush, GingerBreak, cf. Roland [11]). There even exist frameworks that allow easy integration of arbitrary exploits into (malicious) applications [4].

Unfortunately, many typical applications of card emulation are in the areas of payment, ticketing and access control. These applications usually need a safe place to store private keys, tickets and other credentials. While that information can be stored by an application in memory areas that are private to that specific application, in comparison to the secure element scenario, there is a significantly higher risk that malicious applications and malicious users gain access to that information if it is handled by processes on the application processor. Nevertheless, this risk may be tolerable for some application scenarios.

4.2 Cloud-based Secure Element

The cloud-based secure element is an approach to using software card emulation without storing critical data on the mobile phone. In that scenario, the mobile phone acts as a proxy between the contactless interface and a server on the Internet (see Fig. 5). The server acts as if it was a secure element. Consequently, all ISO-DEP commands are transparently forwarded to and handled by that server and all responses from the server are sent through the contactless interface.

Under the assumption that the server provides sufficient protection against unauthorized access to the

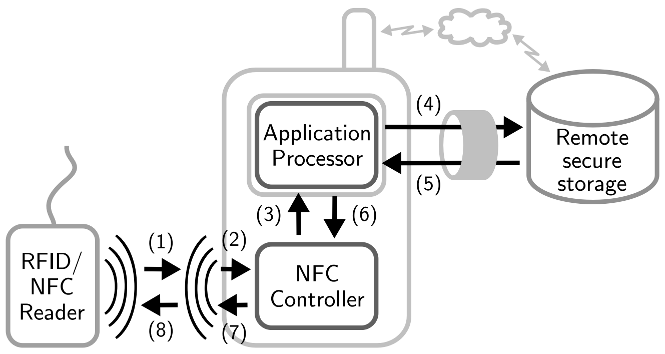


Fig. 5. Cloud-based secure element [10]

Downloaded from
www.mroland.at

critical data and that no malicious user can use another user's cloud-based secure element, this scenario may provide a sufficient level of security for many card emulation applications. However, malicious applications may still be able to gain access to the tunnel between the mobile phone and the remote server. Consequently, such applications could abuse the cloud-based secure element for their own purposes.

Besides this remaining security issue, the cloud-based secure element requires the NFC device to have a stable Internet connection during a transaction in order to establish a tunnel to the remote server.

Nevertheless, there exist service providers that offer cloud-based secure element solutions (e.g. YES-wallet[1] and SimplyTapp[2]).

4.3 Inverse Reader Mode

Up to now, however, the only smart phone systems that support software card emulation are the BlackBerry 7 and 10 platforms and the CyanogenMod after-market firmware for Android devices. Thus, the majority of devices does not permit software card emulation. An alternative approach has been introduced by Saminger et al. [16]: Information stored on the mobile phone is served through reader/writer mode instead of card emulation or peer-to-peer mode. Therefore, the receiver of the information (e.g. an access control gate, a ticket validator or a payment terminal) operates in (software) card emulation mode to interact with the mobile device. Thus, the operating modes of the reader device and the storage device are inverse compared to a regular contactless smartcard system.

As with software card emulation mode, credentials could be stored within a smart phone app or in the cloud. This results in similar security implications.

Systems that already adopted this approach are, for example, EVVA's AirKey[3] and IMA's NFC Porter[4]. Both store their credentials on the mobile phone for offline use.

Nevertheless, on some platforms, inverse reader mode suffers from similar problems as peer-to-peer mode. For example, on Windows Phone 8, only one NDEF message can be transmitted in a single direction per NFC interaction. This limitation makes bi-directional communication in inverse reader mode impossible on that platform.

5. Comparison

Use-cases for NFC can be clustered in four generic groups: peer data exchange (e.g. sharing business cards or URLs, handshake for other communication technologies), NDEF messages trigger actions on devices, access to contactless smartcards (e.g. on-device authentication), and credential storage (e.g. payment, ticketing, access control). While access to contactless smartcards is only possible in reader/writer mode, the other scenarios can be accomplished with multiple operating modes.

5.1 Peer Data Exchange

Peer data exchange is possible using peer-to-peer mode, (inverse) reader/writer mode or software card emulation. Security protocols for channel encryption and authentication exist for peer-to-peer mode. However, these protocols are not implemented on current device platforms. Therefore, regardless of the used communication mode, implementation of security protocols is left to the applications.

There is some support for reader/writer mode and peer-to-peer mode on most mobile device platforms. Peer-to-peer mode has more protocol overhead and has varying levels of support on different platforms. Software card emulation is only supported on a small number of platforms. Nevertheless, many contactless smartcard readers can only communicate with devices in (software) card emulation mode.

Overall, there seems to be no ideal communication mode, as the usability of either mode greatly depends on the interacting device platforms.

5.2 NDEF to Trigger Actions

Triggering actions based on NDEF messages can be accomplished in peer-to-peer mode or reader/writer mode. Peer-to-peer mode is not equally well supported on all platforms while reading NDEF tags is supported on all platforms. Moreover, the protocol stack for peer-to-peer mode is significantly more complex than that for reader/writer mode. Therefore, if the sender of the NDEF messages supports emulation of an NFC tag, the preferred mode for the receiver would be reader/writer mode.

[1] http://www.yes-wallet.com/
[2] http://www.simplytapp.com/
[3] http://www.evva.at/
[4] http://www.nfcporter.com/

Downloaded from
www.mroland.at

With regard to security, the NFC Forum released a specification for digital signature of NDEF messages. Though, that specification is known to be insufficient for integrity protection and authentication of NDEF messages in many situations (cf. Roland et al. [11, 12]). However, a revised version of the digital signature specification and of a certificate infrastructure for NDEF applications is currently developed by the NFC Forum.

5.3 Credential Storage

Stored credentials can be shared using card emulation mode, software card emulation, inverse reader mode or peer-to-peer mode. Card emulation using a secure element and software card emulation provide the best level of interoperability with contactless smartcard readers used in existing applications. However, the secure element ecosystem is too complex and too expensive for many developers. While software card emulation seems to be a promising alternative, that mode is still not widely supported on mobile device platforms.

Peer-to-peer mode and, in particular, inverse reader/writer mode have the best support with regard to smart phone platforms. However, these modes require the counter parts to support peer-to-peer mode and (software) card emulation mode respectively. In addition, peer-to-peer mode requires a rather complex protocol stack compared to inverse reader/mode.

With regard to security, the secure element is certainly the best option while credential storage in a smart phone's regular memory is the worst-case scenario. Cloud-based storage of credentials, which could be combined with software card emulation, inverse reader mode and even peer-to-peer mode, provides an intermediate security level between a secure element and pure credential storage in the mobile phone's memory. Nevertheless, this scenario requires online connectivity during transactions and is potentially vulnerable to hijacking of the cloud-based credential storage.

6. Conclusion

We gave an overview over NFC's different operating modes and new communication concepts based on them. We then compared these modes with regard to their availability, usability and security. Overall, we found that current NFC devices do not yet provide generic security protocols but leave security protocols up to application specific implementations. Moreover, the support of different communication modes varies widely between different device platforms. As a result, the usability of NFC's different communication modes also strongly depends on the device platforms used in actual applications.

Acknowledgments

This work is part of the project "High Speed RFID" within the EU program "Regionale Wettbewerbsfähigkeit OÖ 2007–2013 (Regio 13)" funded by the European regional development fund (ERDF) and the Province of Upper Austria (Land Oberösterreich).

Moreover, this work has been carried out in cooperation with "u'smile", the Josef Ressel Center for User-Friendly Secure Mobile Environments, funded by the Christian Doppler Gesellschaft, A1 Telekom Austria AG, Drei-Banken-EDV GmbH, LG Nexera Business Solutions AG, and NXP Semiconductors Austria GmbH.

References

1. Benninger, C. (2012): Google Wallet – Last Four Digits Revealed to Malware Vulnerability. Intrepidus Group Insight, 20 April 2012.
2. Clark, S. (2011): RIM releases BlackBerry NFC APIs. NFC World, 31 May 2011.
3. Francis, L., Hancke, G.P., Mayes, K.E., Markantonakis, K. (2011): Practical Relay Attack on Contactless Transactions by Using NFC Mobile Phones. Cryptology ePrint Archive, Report 2011/618.
4. Höbarth, S., Mayrhofer, R. (2011): A framework for on-device privilege escalation exploit execution on Android. 3rd Intl. Workshop on Security and Privacy in Spontaneous Interaction and Mobile Phone Use, San Francisco, CA, USA.
5. Hoog, A. (2011): Forensic security analysis of Google Wallet. viaForensics Mobile Security Blog, 12 December 2011.
6. Jakl, A., Roland, M. (2013): Near Field Communication (NFC) Developer Comparison. http://www.nfcinteractor.com/.
7. Langer, J., Roland, M. (2010): Anwendungen und Technik von Near Field Communication (NFC). Berlin, Heidelberg: Springer.
8. Lotito, A., Mazzocchi, D. (2013): OPEN-SNEP project: Enabling P2P over NFC using NPP and SNEP. 5th Intl. Workshop on Near Field Communication (NFC 2013), Zurich, Switzerland: IEEE.
9. RIM (2011): Blackberry API 7.0.0: Package net.rim.device.api.io.nfc.emulation.

Downloaded from
www.mroland.at

10. Roland, M. (2012): Software Card Emulation in NFC-enabled Mobile Phones: Great Advantage or Security Nightmare? 4th Intl. Workshop on Security and Privacy in Spontaneous Interaction and Mobile Phone Use, Newcastle, UK.
11. Roland, M. (2013): Security Issues in Mobile NFC Devices. Ph.D. thesis, Johannes Kepler University Linz, Department of Computational Perception, Linz, Austria.
12. Roland, M., Langer, J., Scharinger, J. (2011): Security Vulnerabilities of the NDEF Signature Record Type. 3rd Intl. Workshop on Near Field Communication (NFC 2011), Hagenberg, Austria: IEEE: 65–70.
13. Roland, M., Langer, J., Scharinger, J. (2012): Relay Attacks on Secure Element-enabled Mobile Devices: Virtual Pickpocketing Revisited. Information Security and Privacy Research, IFIP AICT, vol. 376/2012, Heraklion, Creete, Greece: Springer: 1–12.
14. Roland, M., Langer, J., Scharinger, J. (2013): Applying Relay Attacks to Google Wallet. 5th Intl. Workshop on Near Field Communication (NFC 2013), Zurich, Switzerland: IEEE.
15. Rubin, J. (2012): Google Wallet Security: PIN Exposure Vulnerability. zveloBLOG, 8 Feb 2012.
16. Saminger, C., Grünberger, S., Langer, J. (2013): An NFC ticketing system with a new approach of an inverse reader mode. 5th Intl. Workshop on Near Field Communication (NFC 2013), Zurich, Switzerland: IEEE.
17. Urien, P. (2013): LLCPS: A New Security Framework Based On TLS For NFC P2P Applications In The Internet Of Things. Proceedings of the IEEE Consumer Communications and Networking Conference (CCNC 2013), Las Vegas, NV, USA: IEEE: 845–846.
18. Yeager, D. (2012): Added NFC Reader support for two new tag types: ISO PCD type A and ISO PCD type B. Commit to the CyanogenMod GIT source code repository, 20 January 2012.

Authors

Michael Roland was born in Linz, Austria, in 1984. He is a researcher at the NFC Research Lab Hagenberg (University of Applied Sciences Upper Austria). His main research interests are NFC, security and Android. He is the creator of NFC TagInfo, one of the most successful NFC developer tools for Android devices, and co-author of the book “Anwendungen und Technik von Near Field Communication (NFC)”. He holds a B.Sc. and an M.Sc. degree in Embedded Systems Design (University of Applied Sciences Upper Austria, 2007 and 2009) and a Ph.D. (Dr.techn.) degree in Computer Science (Johannes Kepler University Linz, 2013).

Josef Langer has been working for more than 15 years in the SmartCard, RFID and NFC industry area. He studied electrical engineering at the TU Vienna and RWTH Aachen and received his Ph.D. (Dr.techn.) in Computer Sciences from the Johannes Kepler University in Linz. Langer is professor for microprocessor engineering at the University of Applied Sciences Upper Austria since January 2003. He is head of the NFC Research Lab Hagenberg and head of the Research Group Embedded Systems at his university. He is author of more than 40 publications and co-author of the first NFC book.