Management of Multiple Cards in NFC-Devices

Gerald Madlmayr[1], Josef Langer[1], and Josef Scharinger[2]

[1] FH OOE F&E GmbH, NFC-Research Lab
Softwarepark 11, 4232 Hagenberg, Austria
{gmadlmay, odilling, jlanger}@fh-hagenberg.at
http://www.nfc-research.at
[2] Johannes Kepler Universität
Altenbergerstrasse 48, 4020 Linz, Austria
josef.scharinger@jku.at

Abstract. Near Field Communication (NFC) currently is one of the most promising technologies in handsets for contactless applications like ticketing or payment. These applications require a secure store for keeping sensitive data. Combining NFC with integrated smartcard chips in a mobile device allows the emulation of different cards. Representing each secure element with different UIDs poses several problems. Thus we propose an approach with a fixed UID dedicated to a Secure Element Controller (SEC). This approach allows an optimized backwards compatibility to already established reader infrastructures but also the communication in peer-to-peer mode with other NFC devices. Additionally the communication over peer-to-peer as well as the internal mode of secure elements at the same time is possible. This is approach poses a flexible alternative to the implementations proposed so far. In addition when there are to multiple, removable secure elements in a device it is ensured that the secure elements are only used by authorized user/devices. The SEC in this case handles the communication between the secure elements as well as their authentication.

1 Introduction

Near Field Communication (NFC) is a wireless communication technology to exchange data up to a distance of 10 cm. NFC uses inductive coupled devices operating at the frequency of 13.56 MHz. NFC, standardized in ISO 18092, ECMA 340 and ETSI TS 102 190, is compatible to the standard of contactless smartcards (ISO 14443). An NFC device can operate either in active mode (generating a field and initiating the communication) or in passive mode (unpowered, waiting for a request) [1]. An NFC device is typically made up of two parts: the NFC part responsible for peer-to-peer communication, and a secure element storing sensitive data. Traditionally they are treated as separate devices, represented by different UIDs. But this approach poses several problems.

Despite of possessing multiple IDs, an NFC device has only one antenna. This resource must be shared which by mutually routing antenna I/O to the NFC-IC and the secure elements. As a consequence, a polling device can easily

get confused: Which secure element and which NFC-ID belong together? Polling also takes longer as multiple cycles are needed to fetch all identifiers.

From a technical point of view it is possible to mix both data streams and thus allowing simultaneous access. However, certain existing RFID infrastructures which NFC should be compatible with do not allow more than one device in range (e.g. MasterCard's PayPass [2]) because of security concerns. Many applications communicate via NFC but need some of the secure element's capabilities at the same time (e.g. storage for certificates, encryption of peer-to-peer communication). Using security features of the secure element is not possible with current architectures [3] (Fig.: 3(b)). If one device is selected no other data transfers are allowed. Only after the active NFC connection has been released the secure element can get selected.

For an application it would not be acceptable to interrupt the NFC session for every request that needs to be sent to the secure element. After every break it would be necessary to do a new polling cycle as the target must change its random UID for each new selection. An NFC device should support multiple secure elements. For example, it may have a Universal Integrated Circuit Card (UICC) but may also allow SDIO expansion cards to be inserted. Now the initiator has several problems: Which of the listed ID's are secure elements? Which do belong together?

In our proposal these problems are solved by assigning every NFC device an own UID. The unique ID is complementary to the already existing random 0x08-range ID. When acting as a target, the device will only expose its ID and appears as a legacy contactless smartcard. But there is a difference when getting selected: The device will set bit 6 in its SAK reply, telling its peer it is NFCIP-1 enabled. A legacy infrastructure will ignore this bit and further believe talking to a smartcard. All requests sent will be processed by a Secure Element Controller (SEC) which will distribute the data appropriately. This way it is possible to use many different interfaces for secure elements: conventional T=0/1, upcoming USB, SDIO, SWP, etc.

An NFC device wanting to set up a peer-to-peer connection will just continue sending an ATR request after enumeration. The SEC will intercept this package and divert it to the NFC core. From that time on the device runs in NFC peer-to-peer mode. Another possibility would be to require the initiator to send a special APDU which would cause the target to switch into NFC target mode and reply with its 0x08 random address. The initiator could then immediately select the target without a prior request.

That way, an application can also exchange data via peer-to-peer and send internal requests to a secure element the same time, as the wireless connection is not needed. This data path will be routed through the controller, made available to applications by Host Controller Interface (HCI) commands [4], for example. The HCI defines the architecture of the secure elements around the NFC controller and also specifies communication between secure elements. Thus an internal secure element could communicate with an external tag (e. g. in a smartposter) or establish a communication with another secure element. This

possibility could be used to implement an authentication mechanism between different secure elements. Authentication is a necessity to prevent fraud as a secure element should only work in authorized handsets.

The following section provides an insight into NFC technology and necessary background information for our idea proposed. Section three deals with the problem statement this paper is dedicated to. The concept of the single ID Secure Element Controller will be discussed in section four. The paper ends with a conclusion.

2 Near Field Communication

NFC is an amendment to already existing contactless smartcard technologies, while still being compatible to them. The integration of the technology in devices is driven by the NFC Forum, an industrial consortium founded by Nokia, Sony and NXP. The major idea was the creation of a general proximity communication standard to bridge the gap between Sony's Felica and NXP's Mifare Technology but also other smartcard standards based on ISO 14443 (A and B). Both, Mifare and Felica are proprietary implementations of contactless memory cards based on ISO 14443-3. NFC itself is defined in ISO 18092 and uses ISO 14443-A's data encoding scheme for transfer rates of 106 kBit/s and Felica for higher speeds (currently rates of 212 kBit/s and 424 kBit/s are defined, but may go up to 3 MBit/s in the future). NFC devices are able to interact with existing RFID readers by emulating cards (proximity inductive coupling cards, PICC). To read contactless smartcards or RFID tags, NFC devices can act as a reader or writer (proximity coupling device, PCD), respectively [5]. These two operation modes, card emulation as well as the reader/writer mode of NFC, are necessary for the compatibility with existing proximity card infrastructures.

Additionally, two NFC devices can exchange data through the peer-to-peer mode. Similar to PICC/PCD mode one device is the initiator (master) whereas the other device acts as a target (slave). The communication flow is initiated and controlled by the master device while the target only sends replies. There are two different modes. Either only the initiator emits a field in order to send requests while the targets answers by load modulation using the field or both devices create a field alternately [5]. Communication links between more than two participants at a time are currently not supported.

Although NFC technology is *work in progress* the major parts for device integration are covered by standards. There is an interface standardized between the host controller as well as the NFC controller, named Host Controller Interface (HCI). The HCI allows to control the NFC controller as well as to communicate with internal secure elements [6] and external targets or initiators. The HCI also defines an interface between the NFC controller and multiple secure elements. There can be various types of physical connections between the NFC controller and the SE, such as sign-in/sign-out (S2C) or Single Wire Protocol (SWP) for example [7]. The SWP is a special protocol that allows the NFC controller to exchange data with a UICC attached (Fig.: 1).

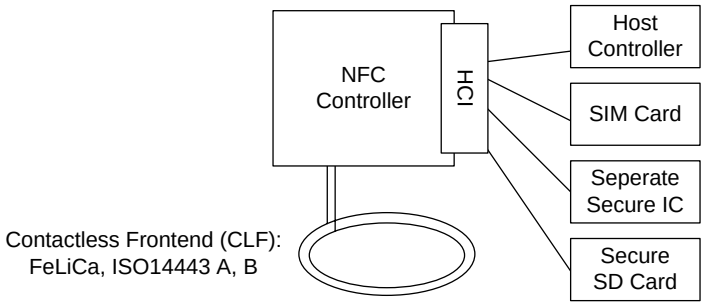


Fig. 1. NFC integration architecture using the HCI.

Per default, any NFC device is in target mode and thus does not generate an RF field and waits silently for a command from the initiator. In case an application on the device requires the initiator mode, the devices can switch into this mode. The field for the initiator mode is only activated, in case no external field is detected. In order to prevent two or more initiators to generate a field at the same time, a RF collision avoidance (RFCA) is implemented in each device. If there is no external field detected the device establishes a field, polls for an NFC device in range and initializes the communication with the target [1].

Whereas in classic smartcard environments the roles of the PCD and the PICC are clear, this is not the case for NFC devices as the conditions are different. As every device can initiate a communication, act as a peer-to-peer target or a transponder, a method is required to avoid chaos if several devices are close together. The approach of implementing the so called *Polling Loop* or *Mode Switch* is discussed in several papers as well as patents [8], [9], [10] (Fig.: 2).

While the device is in target mode, an external reader will detect two (or more) devices represented by UIDs: an NFC target as well as emulated cards. According to ISO 14443 and ISO 18092, targets can be represented by a 4, 7 or 10 byte long unique ID. Whereas the smartcard chip usually uses a fixed ID, the NFCID1 starts with 0x08 while the bytes left are random. This approach was chosen in order to protect the privacy of the device holder.

3 Problem statement

The issue with the existing implementation, as lined out in the previous chapter, is that a reader will detect multiple UIDs of one NFC device. The implementation causes the following problem:

Such an NFC device will not work with legacy readers which only allow one transponder in the field of the reader (mono mode card readers). These kinds of readers do not implement or do not make use of the ISO 14443 anti collision and therefore are not able to establish a communication in case there are multiple

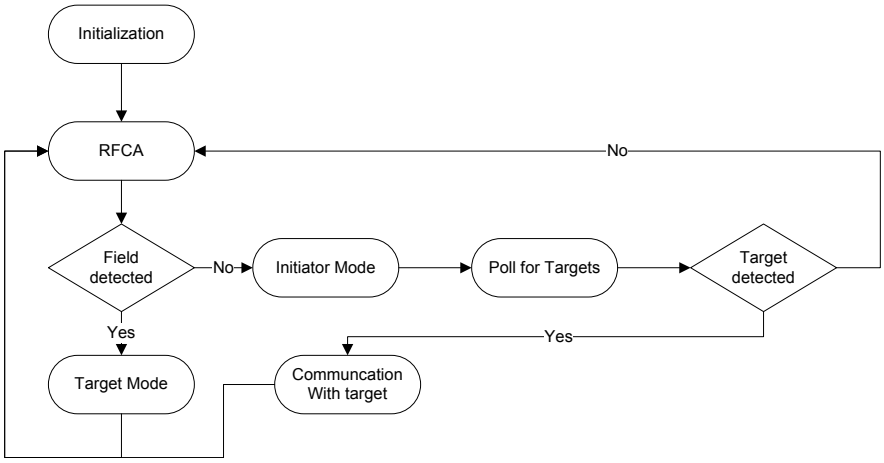


Fig. 2. NFC Mode Switch (simplified).

targets in the field. Regarding NFC devices, the reader infrastructure needs to be replaced in order to be compatible with the new devices. For this case, the backward compatibility is not given. For example, the Nokia 3220 shows multiple UIDs (secure element and NFC) during the polling loop.

Even if the reader is able to process multiple cards in the field, already existing readers are not aware of the fact that UIDs starting with 0x08 are NFCID1s. Thus the reader will select each identifier in the field and setup a communication channel with the target. As the HCI allows the attachment of several secure elements to the NFC controller, this process is inefficient and time consuming. NFC readers do not have this problem, as long as they want to establish a peer-to-peer connection. In this case the NFCID1 is selected. With regard to selecting the correct secure element/smartcard, also NFC readers face the same problem as ordinary contactless smartcard readers, as they can not know which UID to select. We will deal with this issue in the first part of the paper.

Additionally, having one or more removable secure elements other than the UICC requires additional mechanisms against theft, abuse and management. For example, secure SD Cards containing contactless applications could be taken from the original device and be inserted into a new NFC device. The application provider who has data stored in this SD Card will not be informed about this issue. Thus locking the secure element or managing contactless applications over-the-air (OTA) can not be granted as the application provider does not know the new Mobile Station International Subscriber Directory Number (MSISDN) of the new handset. This problem will be discussed in the second part of the paper.

3.1 Possible Solutions

Explicit Card Select: In this case the user has to explicitly select one of the secure elements to be presented to the reader. From a technical perspective the implementation is simple and will solve the problem in an efficient way. From a usability perspective it is not satisfying. One reason is that the user has to know which secure element to choose, as there can be multiple secure elements in the device. Additionally it is not possible to simply browse the secure element in order to figure out, which applications there are on which smartcard chip. This is due to privacy issues, because any other party would be able to see all the applications installed in the secure element as well. Secondly the explicit select is a complex process for the end-user from an interaction point of view. And this is actually not the idea behind NFC. The *Touch and Go* philosophy is a clear statement for a simple interaction between devices without browsing menus. Even if the explicit selection of the secure element is not an issue (in case there is only one SE in the device), the device still might operate in NFC target mode and thus send an NFCID1. Thus the user would directly have to select the application to use (payment, Bluetooth pairing etc.).

Time Multiplex: By using a time multiplex the secure elements are presented consecutively to the reader. In this case the reader will only see one UID at a time. The process is the following. The reader selects the first card and tries to access the application required. If the application is not found, the secure element is put into halt state. The halt state is the signal for the NFC device that the secure element selected was not the correct one. Then the reader sends another ISO 14443 request command. As a reaction the NFC device will present the second secure element to the reader. This process is repeated as long as the reader has found the secure element/NFC target required. If the required smartcard chip is not found and all cards of the NFC device are in halt state, no more card would be presented to the reader. From a technical point of view this is a feasible solution. The issue with this implementation is, that the reader has to put the card into halt state in case it was not the transponder it was looking for. This is actually not yet the case for reader infrastructures deployed and would require software/firmware updates of these readers. From a usability point of view, the implementation is user friendly, as there is no interaction required and the user does not have to care about switching between different modes or applications. A minor issue is the fact, that in worst case the reader has to go through all different cards/NFC targets in a device, which is not efficient.

Aggregation with one single UID: The third option would be the use of only one UID per device. In this case the NFC controller has to route the commands accordingly to the secure elements. This requires additional software running in the NFC controller inspecting if possible the data sent between NFC target/secure elements and external reader. From the technical side this implementation is the most complicated one. However, it neither requires any user interaction nor any modification to an already existing in-

frastructure. The major disadvantage of this approach is, that applications using the UID for identification or cryptographic functions (e. g. Mifare [11]) will not work. But as the UID of a smartcard is public using it for critical tasks as already mentioned is unsafe anyway. From the current perspective using only one UID per device is a suitable solution that allows interoperability with any established infrastructure. Therefore we consider this option for our implementation.

4 Implementation

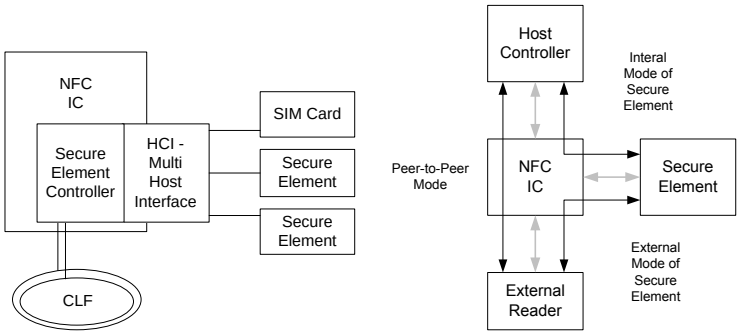
We propose the use of only one UID per NFC device and integrate a Secure Element Controller (SEC) to internally route the communication flows (Fig.: 3(a)). The SEC is a software implementation running in the NFC controller. For communication with the different instances such as secure elements, external readers or the host controller Application Protocol Data Units (APDU) are used. The communication handling makes use of the underlying HCI. The NFC device will be given an identifier out of the 0x08 range in order to correctly represent an NFCID1. NFC peer-to-peer capabilities are indicated by bit six in the SAK/SEL response of the NFC device after the selection command [12], [1]. Bit five indicates whether there is an ATS available and further ISO 14443-4 commands are accepted. Even if this bit is set, proprietary protocols could be used on top. For example, this is also the case for JavaCards integrating a Mifare section. With regard to the Mode Switch, complexity is reduced. There will be only a switch between initiator mode and *one* target mode. The SAK in this case is accomplished by the SEC. The explicit tasks for the SEC are:

1. Acting as a proxy between the smartcard chips and external readers to route the data correctly.
2. Managing the communication between the secure elements to implement an authentication mechanism.
3. Direct the OTA management connections from the UICC to other secure elements.

4.1 External Communication

With regard to the external communication, the NFC device has to reply to the following requests:

NFC initiator: The initiator establishes an RF field, runs the select process and after the SAK received, the initiator sends the attribute request and will continue with NFCIP-1 frames. Already out of the data encoding scheme the SEC can assume the following: If it is ISO 14443 A (106 kBit/s), the device polling is an NFC initiator or a ISO 14443 A reader. In case the SEC receives a request using the Felica encoding scheme (212 or 424 kBit/s) the initiator could also be a Felica reader. As after the selection the initiator



(a) Integration of a Secure Element Controller (SEC).

(b) NFC Operating Modes [3]

would send an attribute request as mandatory in NFCIP-1. In this case the SEC forwards the data stream to the NFC core indicating that the data stream is a peer-to-peer connection.

Smartcard Reader: To handle multiple, different smartcards, the SEC is required to keep track of the secure elements in the NFC device. When a secure element is inserted, the SEC analyses the type (Felica, ISO 14443 A or ISO 14443 B). The internal routing of the data stream is quite simple as long as there is only one instance of a smartcard type in the device. If there is one smartcard chip of a type, the routing gets more complex (Felcia is not considered at the moment, as there was too little documentation available).

With regard to Mifare, the SEC has to provide an aggregation of all the Mifare Application Directories (MAD) ID. Although the proprietary authentication and encryption of Mifare also uses the UID of the smartcard chip, and thus is not feasible with our implementation, this might be different for the upcoming Mifare+ using AES. The external Mifare reader in this case is able read the MAD from the SEC like from an ordinary Mifare card. When trying the access a special block containing the application data, the SEC redirects the authenticate-, read-, and write-command to this chip.

In JavaCard OS using GlobalPlatform the functionality is a little different. An external reader is not able to query all the applications available, without proper access rights. Usually the reader sends a select request with the Application Identifier (AID) to the transponder whose answer it either positive or negative. In our case the select request is sent to the SEC, which distributes the request to all smartcard chips. If there is a positive responses the SEC then will redirect the communication flow accordingly. If there is no positive response, the SEC will return a negative answer to the reader. In case there are more than one positive response sent to the SEC, the SEC would forward a negative response to the reader and additionally contact the trusted service manager (TSM). In this case there is a management issue with the secure elements [7]. The SEC keeps a temporary list of AIDs

and the appropriate secure elements in a registry, which is cleared in case the phone is rebooted. This helps the SEC to save time as the multi cast has to be performed only once.
When an external smartcard reader communicates with the NFC device, the SEC identifies the type of the reader by analyzing the commands/APDUs after the selection process has been completed and forwards the data stream to the appropriate smartcard chips. In case there is more than one instance of a smartcard type available, the SEC will consult the AID registry to route the data correctly. Thus the external reader is not able to distinguish between a smartcard and an NFC device.
An explicit selection of the appropriate secure element by the reader by indicating a communication through a logical channel is another option. This implementation would require, that there is fixed mapping between secure elements and logical channel IDs. Additionally, the reader has to know on which secure element there is the application the reader is looking for. This is the major issue with the mentioned implementation. Therefore logical channels are not further considered for explicit selection.

4.2 Internal Authentication of Secure Elements

Besides only having a single UID for contactless communication, the SEC also poses another advantage. The SEC can be used as a central instance in the device to perform the management of the secure elements. As the SEC connected thought the SWP with the UICC, the SEC is able to make use of the wireless communication capabilities of the handset. This is an important feature with regard to the authorization of secure elements. The issue with a removable secure element, like the ones used in the Benq T80, is that the secure element can be inserted into any other mobile phone. As the authenticity of the phone is not validated by the secure element and vice versa, the vital feature of remote management of the secure elements is lost. The HCI is designed for a direct communication between different secure elements. This option allows a bilateral authentication between secure elements and/or the SEC. The UICC of the mobile phone serves as a gateway in order to retrieve certificates for the validation if necessary. By a simple example we demonstrate how the implemented system works on a JavaCard platform.

Each secure element, no matter if it is a UICC or a removable secure element such as a SD Card, contains an activation applet in the issuer security domain. This applet holds a public/private key pair and a certificate from the issuer and the appropriate root certificate for validation. Also the SEC contains a public/private key pair as well as a certificate from the issuer.

During the boot sequence of the mobile phone, the SEC first selects the activation applet on the UICC. Then a secure channel is establishes and a bilateral authentication is performed. In case this is the first time for the UICC communicating with the SEC, the UICC established a data connection to the CA/TSM in order to validate the certificate of the SEC. The public key to validate the certificate is kept by the UICC, hence the validation of the SEC does not require

an OTA connection the next time. The validation of the UICC's certificate for the SEC is more difficult, as the SEC can not yet trust the UICC. However, the SEC instructs the UICC to establish a connection to the appropriate CA/TSM to validate the UICC's certificate. Then the SEC encrypts the UICC's certificate with its public key, signs it and adds an identifier. The external party processes the request, checks the certificate, encrypts the appropriate public key with the SEC's public key and also signs it. The SEC can verify the authenticity of the data by validating the signature of the package and additionally only the SEC is able to decrypt the data. The retrieved public key allows the SEC to validate the certificate of the UICC and sets the activation flag in the activation applet. This flag will be set to false again, as soon as the UICC is reseted/powered off. The activation applet on the UICC provides a shared interface that allows other applets to read the activation flag and then decides whether to take an action or not (FIg. 3).

After activating the UICC a second secure element can be authenticated. This is a slightly more complex process, as the secure element can neither trust the SEC nor the UICC. Basically the SE establishes the communication with the UICC first, in order the have an online connection to validate the certificate of the UICC. The secure element also receives the certificate to validate the authenticity of the SEC.

In both cases the online connection to obtain the certificates from the CA/TSM for validation needs to be established once. The certificates are kept in the activation applets of the UICC, SEC and SEs for further use. As soon as the UICC or the SEs are reseted, the activation flag is set to false and needs to authenticate the other components again. With this implementation, the use of stolen SEs is prevented. The SE can store more than one certificate, allowing the user to change the SE between two handsets. Besides protecting the use of secure elements without permission, the platform manager (TSM) is always able to remotely access the secure elements OTA (Fig. 4).

5 Conclusion

The proposed a Secure Element Controller (SEC) enhances the compatibility of upcoming NFC devices with existing infrastructures. The implementation results in a single ID for NFC devices with multiple operating modes and multiple secure elements. The SEC takes over the routing of the data streams to the appropriate secure element chip. The next step of our conception approach is the implementation of a SEC using the NFCBox [5].

There are several major benefits of our proposal. The most important one is the interoperability with already existing infrastructures. Although NFC is claimed to be compatible anyway, this is not the case when it comes to readers supporting only one card in the field. One of the most popular services using such a kind of reader is *MasterCard's PayPass*.

Secondly the integration of a SEC allows the parallel use of the peer-to-peer mode and the internal communication with the secure elements (internal mode;

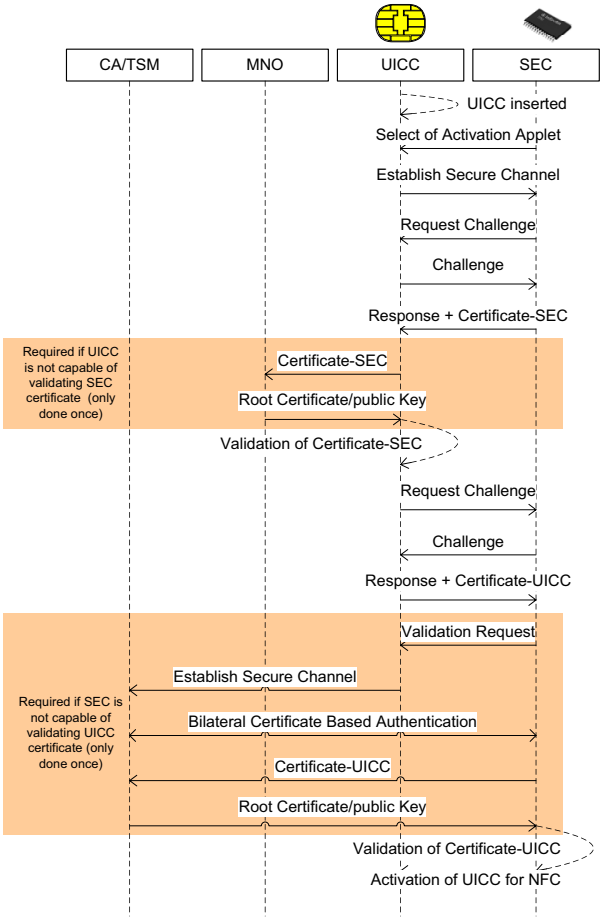


Fig. 3. Activation of the UICC by the SEC for contactless communication.

Fig.: 3(b)). In order to secure the plain NFC peer-to-peer data stream the use of a secure element to perform authentication or encryption is a reasonable feature, but not supported by the current architecture. Services that only rely on the fixed identifier of the smartcard are not feasible with this implementation, as the ID is partly random. However, systems only using the UID of contactless smartcards are unsecured anyway. This is due to the reason that the UID is primarily necessary for the anti-collision and selection of the transponder. Hence, no authentication or encryption is needed to read this ID.

Additionally the SEC can facilitate the authorization of secure elements as well as the OTA management. Hence, Trusted Service Managers are able to handle applications in a secure element other than the UICC and the he proposed authentication mechanisms avoids abuse of contactless applications. To sum up,

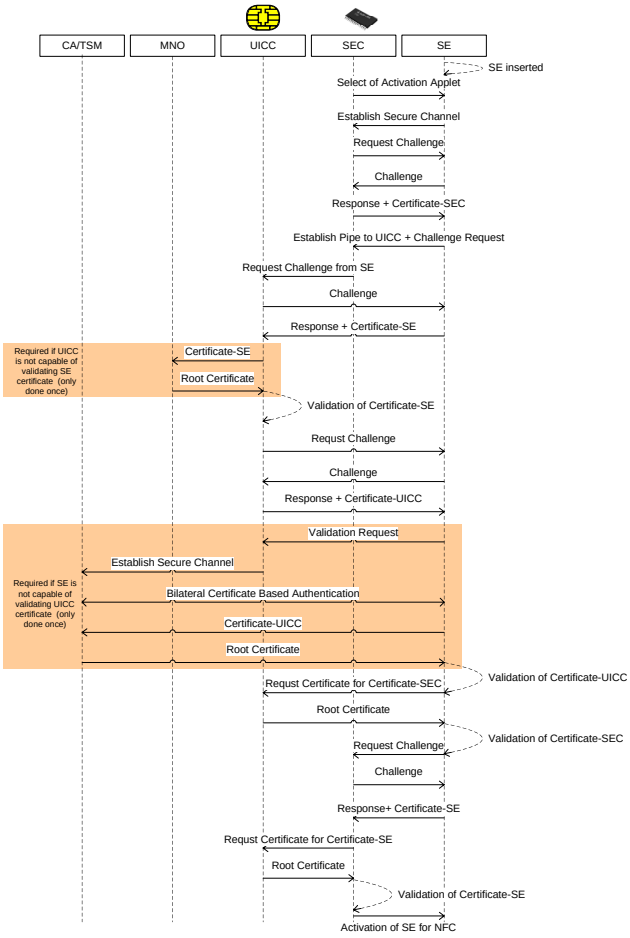


Fig. 4. Activation of an additional SE by the SEC for contactless communication.

the integration of a SEC would bring several benefits to an NFC device with regard to interoperability, security and manageability.

References

1. International Organization for Standardization: Near Field Communication - Interface and Protocol (NFCIP-1). ISO/IEC 18092 (2004)
2. EMVCo LLC: EMV Contactless Specifications for Payment Systems. http://www.emvco.com/ (2006) PayPass ISO/IEC 14443 Implementation Specification.
3. Kunkat, H.: NFC und seine Pluspunkte. Electronic Wireless **01** (2005) 4 – 8

4. ETSI: Smart Cards: UICC-CLF interface; host Controller Interface. www.etsi.org (2007) Draft (Release 7).
5. Dillinger, O., Langer, J., Madlmayr, G., Muehlberger, A.: Near field communication in embedded systems. In: Proceedings of the Embedded World Conference 2006. Volume 01. (2006) 7
6. Bishwajit, C., Juha, R.: Mobile Device Security Element. Mobey Forum, Satamaradankatu 3 B, 3rd floor 00020 Nordea, Helsinki/Finland. (2005)
7. GSMA London Office 1st Floor, Mid City Place, 71 High Holborn, London WC1V 6EA, United Kingdom: mobile NFC technical guidelines. 2.0 edn. (2007) 1st Revision.
8. Dillinger, O., Madlmayr, G., Schaffer, C., Langer, J.: An approach to nfc's mode switch. In Dreiseitl, S., Hendorfer, G., Jodlbauer, H., Kastner, J., Mueller-Wipperfuerth, T., eds.: Proceedings of the Science Day of the University of Applied Sciences Upper Austria. Volume 2., FH OOe F & E GmbH, Shaker Verlag, Aachen (2006) 6
9. Dawidowsky, F.: Method for operating a near field communication system. Europena Patent Office (2006) EP 1 653 632 A1.
10. Rowse, G., Pendleburyrel, J.: Electronic near field communication enabled multifunctional device and method of its operation. Patent (2006) US Patent Application Publication.
11. Nohl, K.: Cryptanalysis of crypto-1. http://www.cs.virginia.edu/~kn5f/Mifare.Cryptanalysis.htm (2008)
12. International Organization for Standardization: Proximity cards. ISO/IEC 14443 (2003)