Chapter 2
Basics

This chapter summarizes basic concepts of smartcards, Near Field Communication (NFC) and payment cards.

2.1 Smartcards

Smartcards are identification cards equipped with a microchip (integrated circuit, IC). Depending on their functionality, they can be grouped into memory cards and processor cards [40]. Memory cards contain simple memory logic that can be accessed with primitive read and write commands. Processor cards contain a microprocessor that can execute complex programs. Classic smartcards have a contact interface standardized in ISO/IEC 7816-2 [22]. These contact pads can be used with various synchronous and asynchronous communication protocols. Synchronous protocols are typically used for memory cards while asynchronous protocols are typically used for processor cards. ISO/IEC 7816-3 [21] defines a set of asynchronous transmission protocols. Some smartcards even have a Universal Serial Bus (USB) interface as standardized in ISO/IEC 7816-12 [19]. Besides classical contact interfaces, cards can also have contactless interfaces that follow Radio Frequency Identification (RFID) standards.

Smartcards are present in our everyday lives. We use them as credit and debit cards, as ID cards and as access control tokens. There also exist smartcards in other form factors. For instance, the Universal Integrated Circuit Card (UICC) in mobile phones (also known as the Subscriber Identity Module (SIM) card) is a smartcard too. Even electronic passports contain contactless smartcard technology. In particular smartcards with a contactless interface are no longer bound to specific form factors. Instead they could be integrated into virtually any object.

M. Roland, *Security Issues in Mobile NFC Devices*, T-Labs Series
in Telecommunication Services, DOI 10.1007/978-3-319-15488-6_2

2.1.1 Protocol Stack

The smartcard protocol stack is standardized in the ISO/IEC 7816 series. Part 1 [27] describes the physical characteristics of smartcards. Part 2 [22] describes the contact interface. Part 3 [21] describes the electrical interface and the low-level transport protocols. Part 4 [20] describes the application layer protocol.

2.1.1.1 ISO/IEC 7816-3

ISO/IEC 7816-3 [21] defines an asynchronous serial protocol for character based exchange of information between a smartcard reader ("terminal") and a smartcard. The standard further defines the reset procedure that initializes the communication with the smartcard. In response to this reset procedure, the smartcard sends its Answer-to-Reset (ATR). The ATR contains information about communication speed, a list of supported protocols and parameters and product-specific data. On top of the asynchronous serial protocol, ISO/IEC 7816-3 defines two half-duplex transport protocols: the byte-oriented protocol T = 0 and the block-oriented protocol T = 1.

2.1.1.2 ISO/IEC 7816-4

ISO/IEC 7816-4 [20] defines an application layer protocol for smartcards. The protocol consists of a file system and commands for access to the file system, management of logical communication channels, and securing the communication. The file system consists of a master file (MF), dedicated files (DFs) and elementary files (EFs). DFs can be seen as directories. They may host complete applications, group files or store data objects [20]. EFs are the leaf nodes of the file system and contain the actual data.

The application level communication protocol is mapped on top of the lower layer transport protocol (e.g. T = 0 or T = 1). Command-response pairs are called Application Protocol Data Units (APDUs). Commands are always sent from the smartcard reader to the card while responses are always sent from the card to the reader.

Table 2.1 shows a command APDU. It consists of a header and a body. The header field contains the command class (CLA), an instruction code (INS) and instruction parameters (P1, P2). The body contains data associated with the command and length fields for the command data (Lc) and the expected response (Le).

Table 2.2 shows a response APDU. It consists of a body and a trailer. The body contains the response data. The trailer contains the status word (SW1, SW2). Typical status words are:

Table 2.1 Command APDU (based on [20])

Field type	Field name	Size	Description
Header	CLA	1 byte	Command class
	INS	1 byte	Instruction byte
	P1	1 byte	Parameter byte 1
	P2	1 byte	Parameter byte 2
Body	Lc	0–3 bytes	Command data length N_c
	DATA	N_c bytes	Command data
	Le	0–3 bytes	Response data length N_e

Table 2.2 Response APDU (based on [20])

Field type	Field name	Size	Description
Body	DATA	$\leq N_e$ bytes	Response data
Trailer	SW1	1 byte	Status word (first byte)
	SW2	1 byte	Status word (second byte)

- `0x9000`: The command has been successfully executed.
- `0x61NN`: The command has been successfully executed, but `0xNN` bytes of further data are waiting for retrieval.
- `0x6XYY` ($2 \leq X \leq 3$): The execution of the command ended with a warning.
- `0x6XYY` ($4 \leq X$): The execution of the command ended with an error.

2.1.2 Contact versus Contactless Smartcards

Instead or in addition to a contact interface, some smartcards have a contactless interface. The most common interface for contactless smartcards is standardized in the ISO/IEC 14443 series. This standard defines a proximity RFID system based on inductive coupling with an operating frequency of 13.56 MHz. On top of the ISO/IEC 14443 protocol stack, a contactless smartcard can either use a proprietary protocol or use the APDU-based protocol defined in ISO/IEC 7816-4. Figure 2.1 shows a comparison of the protocol stack of contact-based and contactless smartcards. The ISO/IEC 14443 standard is split into four parts. Part 1 [23] specifies the physical characteristics of the card and the antenna. Part 2 [25] defines modulation and coding schemes of the bit transfer layer and the power supply of passive cards over the Radio Frequency (RF) interface. Part 3 [26] defines the activation and anti-collision sequence and a frame-based communication protocol. Part 4 [24] specifies a half-duplex block-oriented transmission protocol comparable to T = 1. ISO/IEC 14443 is split into two types: Type A and Type B. These types differ in their modulation and coding schemes, in their activation and anti-collision protocols and in their frame-

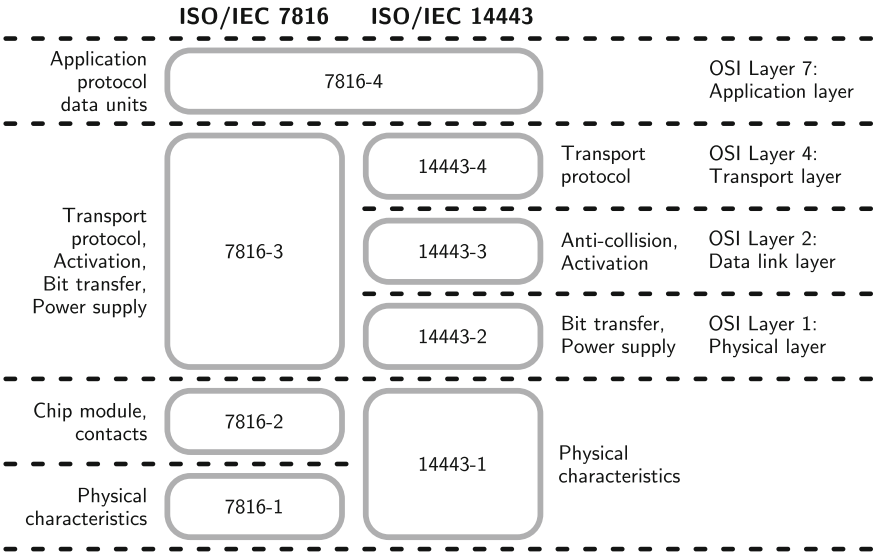

Fig. 2.1 Comparison of the ISO/IEC 7816 contact protocol stack and the ISO/IEC 14443 contactless protocol stack (*Source* [29])

based communication protocol. The block-oriented transmission protocol is the same for both types.

2.1.2.1 ISO/IEC 14443-3

ISO/IEC 14443-3 [26] is a reader-talks-first protocol. Thus, the communication is always started with a request from the reader to the card. The card then returns a response to the reader. In ISO/IEC 14443 terminology, the smartcard reader is a Proximity Coupling Device (PCD) and the smartcard is a Proximity Integrated Circuit Card (PICC).

PICCs have unique or pseudo-unique addresses that are used to identify and operate multiple cards simultaneously with one PCD. While the PCD is in idle mode, it polls for PICCs with repeated REQUEST commands (REQA for Type A and REQB for Type B).

For Type A, the REQUEST command causes all cards that have not been activated before to synchronously answer with their ATQA (Answer-to-Request). The reader then knows that at least one new card is available and continues with the anti-collision procedure. The anti-collision procedure enumerates all cards based on their Unique Identifier (UID) using a binary search tree algorithm [10]. After successful anti-collision, the PICC reveals whether it supports the transport protocol according to ISO/IEC 14443-4 or uses a proprietary transmission protocol [10].

For Type B, the REQUEST command immediately starts the anti-collision protocol which is based on a slotted-ALOHA algorithm [10]. The slotted-ALOHA algorithm works by splitting the responses of the different PICCs to the REQUEST command into multiple time slots. Each card sends its ATQB (Answer-to-Request) in its time slot. The ATQB contains the Pseudo Unique PICC Identifier (PUPI), protocol parameters and application parameters of a card. If there is a collision-free transmission in one of the time slots, the PCD has the PUPI of that PICC and can then address the card.

Besides the anti-collision and activation procedure, ISO/IEC 14443-3 also defines the frame formats and timing requirements for exchanging data between the PCD and the PICC.

2.1.2.2 ISO/IEC 14443-4

ISO/IEC 14443-4 [24] defines the protocol activation and the half-duplex block-oriented transmission protocol for contactless smartcards. For Type A, the PCD first requests the Answer-to-Select (ATS) from the PICC. The ATS is similar to the ATR of a contact-based card and contains protocol parameters and the historical bytes. The historical bytes contain free-form product identification data. For Type B, the protocol and application parameters have already been exchanged with the ATQB. After protocol activation, the transmission protocol is the same for both, Type A and B.

2.1.2.3 Other Contactless Protocols

Besides ISO/IEC 14443, also other protocols for contactless smartcards exist. ISO/IEC 15693 is a vicinity RFID standard that uses the same operating frequency (13.56 MHz) and the same communication principle (inductive coupling) as ISO/IEC 14443. However, it is specified for longer communication distances at the price of slower data rates. ISO/IEC 15693 is mainly used for simple memory cards.

Sony's proprietary FeliCa is a smartcard technology that is similar to ISO/IEC 14443. FeliCa has a file system similar to that defined in ISO/IEC 7816-4. The file system and commands for access to the file system are standardized in JIS X 6319-4 [28]. In addition, the FeliCa system has proprietary cryptography and security features.

2.1.3 Smartcard Software

Application specific smartcards (e.g. bank cards) may use customized operating systems and their application software is usually programmed into a read-only memory during the manufacturing process. Today, however, there also exist generic smart-

card platforms which can be loaded with various applications. A single card can even contain multiple applications at the same time.

2.1.3.1 Java Card

A standardized framework for multi-application smartcards is the Java Card platform. Java Card operating systems provide a common set of application programming interfaces (APIs) and a standardized run-time environment. This allows development of applications that are independent of the actual smartcard hardware and of the actual operating system. As a consequence, a Java Card application that has been compiled for a certain version of the Java Card API can be run on any Java Card compliant smartcard that implements that API version.

A Java Card application consists of one or more *applets*. When an application is installed onto a Java Card each applet instance is assigned a name (application identifier, AID). Using this AID, the applet can be selected for further communication with the SELECT (file by DF name) command from the ISO/IEC 7816-4 command-set. After selection, further commands—except for the selection of other applets—are sent to the Java Card applet for processing. Hence, it is up to an applet to interpret commands and to (possibly) provide a file system like view on application data.

2.1.3.2 GlobalPlatform

Besides a common API, an application provider also needs a standardized interface to manage the lifecycle and the application software of a smartcard. "The GlobalPlatform architecture is designed to provide card issuers with the system management architecture for managing these smart cards" [13]. GlobalPlatform specifies interfaces, mechanisms and commands to allow secure smartcard application management. The management facilities are independent of the actual smartcard hardware and of the actual operating system, and are, thus, interoperable.

A GlobalPlatform compliant smartcard contains a *Card Manager*, which is the central component for card administration. It is responsible for managing card content (applications and data), security domains and the whole card lifecycle. GlobalPlatform provides standardized methods to load, install and configure applications on a smartcard. During the load operation the application executable load-file is stored on the card. Then the application and its applets can be installed, enabled for selection and personalized.

2.1.4 Data Structures Used on Smartcards

ISO/IEC 7816-4 defines five different files structures for smartcards:

1. transparent structure,
2. records of fixed size in a linear structure,

3. records of variable size in a linear structure,
4. records of fixed size in a cyclic structure, and
5. tag-length-value (TLV) format structure.

The transparent structure is a simple binary file format with byte-wise random access. Record files can be accessed on a per-record basis.

The TLV format is a special type of file structure where each data object consists of an identifier ("tag"), length information for the data part ("length") and a data part ("value"). TLV data objects can even be nested. Thus, the value of one data object might consist of one or more TLV data objects. TLV structures are not limited to EFs. Instead, many smartcard applications use these structures for various purposes. In this book, TLV structures are represented in the following format:

```
<TAG> <LENGTH> (name)
    <VALUE> (interpretation)
```

For instance:

```
6F 12 (FCI template)
   84 0E (DF name)
      325041592E5359532E4444463031 ("2PAY.SYS.DDF01")
   A5 00 (Proprietary information encoded in BER-TLV)
```

In this example `0x6F` is the tag of an *FCI template* data object that contains `0x12` (18) bytes of data. The FCI template contains two nested data objects: `0x84` is the tag of a *DF name* data object with a length of `0x0E` (14) bytes. The data object contains the application identifier "2PAY.SYS.DDF01" (the bytes 325041592E5359532E4444463031 represented in US-ASCII character encoding). `0xA5` is the tag of a *Proprietary information encoded in BER-TLV* data object that contains no data.

2.1.5 PC/SC

PC/SC (Personal Computer/Smart Card) is a standard to connect smartcards to PC platforms. It is supported on different operating systems (e.g. Microsoft Windows, Apple OS X and Linux). PC/SC APIs and wrapper APIs that rely on the PC/SC functionality of the underlying operating systems are available for various programming languages (e.g. C++, C#, Java and Python).

2.2 Near Field Communication

Near Field Communication (NFC) is a contactless communication technology for communication over short distances. NFC has been developed by NXP Semiconductors (formerly Philips Semiconductors) and Sony as an evolution of their inductively coupled proximity Radio Frequency Identification (RFID) technologies and

smartcard technologies. NFC has originally been standardized by Ecma International in ECMA-340 [4] and ECMA-352 [3]. These standards have later been adopted by ISO/IEC in ISO/IEC 18092 [17] and ISO/IEC 21481 [18]. Further ISO/IEC and Ecma standards exist that describe test methods and enhanced interface protocols. Besides standardization through these normative bodies, further specification of protocols, data formats and NFC applications is driven by the NFC Forum.

2.2.1 NFC Forum

The NFC Forum[1] is an association of industry organizations (in particular manufacturers, application developers, and financial service institutions) and non-profit organizations with an interest in NFC. The NFC Forum was originally founded by NXP Semiconductors, Sony and Nokia to promote the use of NFC technology [29].

Today, the NFC Forum creates specifications for data formats, protocols and reference applications. A certification program based on these specifications, assures interoperability between different products and implementations.

2.2.2 Operating Modes

NFC has three operating modes:

1. peer-to-peer mode,
2. reader/writer mode, and
3. card emulation mode.

2.2.2.1 Peer-to-Peer Mode

Peer-to-peer mode is an operating mode specific to NFC and allows two NFC devices to communicate directly with each other. This mode is based on the communication protocol standardized in ISO/IEC 18092 [17]. On top of this protocol, the NFC Forum specified the Logical Link Control Protocol (LLCP) as a protocol that allows bi-directional communication between logical end-points of the two NFC devices [29]. Further high-level protocols (e.g. Simple NDEF Exchange Protocol, SNEP) allow the exchange of standardized data structures across an LLCP link.

[1] http://www.nfc-forum.org/.

2.2.2.2 Reader/Writer Mode

In reader/writer mode, an NFC device can access passive NFC tags. NFC tags are a subset of RFID transponders that contain a simple memory structure for storing data in a standardized format. NFC tags provide a basis for interoperability between NFC devices. Besides NFC tags, many NFC devices in reader/writer mode can also interact with other proximity RFID transponders and contactless smartcards that are based on the standards ISO/IEC 14443 [23–26] and JIS X 6319-4 [28]. Some NFC devices can even communicate with NXP's MIFARE Classic tags and with vicinity RFID transponders based on the standard ISO/IEC 15693 [14–16]. As a consequence, reader/writer mode makes NFC devices interoperable with legacy RFID tag and smartcard infrastructures.

2.2.2.3 Card Emulation Mode

In card emulation mode, an NFC device emulates a contactless smartcard. Thus, while an NFC device is in this mode, it can be accessed by existing RFID readers as if it was a regular contactless smartcard. As a consequence, card emulation mode makes NFC devices interoperable with legacy RFID reader infrastructures.

There exist several possible options for NFC card emulation mode. Emulation can differ in communication standards, in supported protocol layers, in supported command sets and in the part of the NFC device that performs the actual emulation.

With regard to the communication standard, an NFC device could emulate ISO/IEC 14443 Type A, ISO/IEC 14443 Type B or JIS X 6319-4 (Sony's FeliCa). Support for either of these modes depends on the NFC controller, the secure element and typically the geographic region. For example, ISO/IEC 14443 is the prevalent technology in Europe and North America as it is used with many payment and access control applications. For instance, contactless credit card standards are based on ISO/IEC 14443. FeliCa (JIS X 6319-4) is popular in Japan where it is used for many payment systems.

Another difference is the part of the device that performs the actual emulation. On the one hand, a contactless smartcard can be emulated by a dedicated smartcard chip, the so-called *secure element*. On the other hand, card emulation can be performed in software on the main application processor of a device (*host-based card emulation* (HCE), *software card emulation* [41], or *soft-SE* [11]).

2.2.3 NFC Tags

NFC devices in reader/writer mode and NFC tags are used to enable NFC's *tagging* application scenario. The basic principle behind tagging is "*it's all in a touch*" [2]. I.e. touching an object with an NFC device triggers an action on that device. For example, a printed advertisement could contain a tag that links to interactive content.

Table 2.3 Overview of the NFC Forum tag types (based on [29])

Tag type	RFID technology[a]	Standards	Maximum memory size
Type 1	Innovision topaz	ISO/IEC 14443-3 Type A	2 KB
Type 2	NXP MIFARE ultralight	ISO/IEC 14443-3 Type A	2 KB
Type 3	Sony FeliCa lite	JIS X 6319-4	1 MB
Type 4	NXP MIFARE DESFire[b]	ISO/IEC 14443 & ISO/IEC 7816-4	64 KB (4 GB[c])

[a]The column *RFID technology* lists only the product that the tag types were originally based on
[b]An NFC Forum Type 4 tag could be implemented on any programmable contactless smartcard that supports ISO/IEC 14443 and ISO/IEC 7816-4
[c]In version 3.0 of the Type 4 Tag Operation specification

Tapping the tag with an NFC-enabled mobile phone could cause a web site to be opened, a phone call to be initiated or a ready-made SMS message to be sent.

In order to achieve interoperability between NFC tags and NFC devices, the NFC Forum defined four different tag types that should be supported by all NFC devices. These tag types are based on existing RFID tag technologies. Table 2.3 gives an overview of the four tag types. For each tag type, the NFC Forum released a *Tag Operation Specification* [36–39]. These specifications define the memory layout of the tags and commands to access the tags.

2.2.4 NFC Data Exchange Format (NDEF)

The *NFC Data Exchange Format* (NDEF, [30]) is defined by the NFC Forum as a common format for storing data on NFC tags and for data exchange between NFC devices in peer-to-peer mode [29]. NDEF abstracts the data from the storage medium and the communication channel. Thus, on the application level an NFC device can operate on NDEF messages and need not cope with different tag platforms and operating modes.

NDEF is a simple binary data format that encapsulates application data and meta-information [29]. Data is packed into NDEF records, where each record contains type information and optional identification information for the data packet. Multiple records are grouped into one NDEF message.

2.2.4.1 NDEF Record

Figure 2.2 shows the layout of an NDEF record. A record consists of multiple header fields and a payload field. The header starts with five flag bits:

1. *Message Begin* (*MB*): MB marks the first record of an NDEF message.
2. *Message End* (*ME*): ME marks the last record of an NDEF message.
3. *Chunk Flag* (*CF*): The CF, if set, specifies that the record payload is continued in the next record.

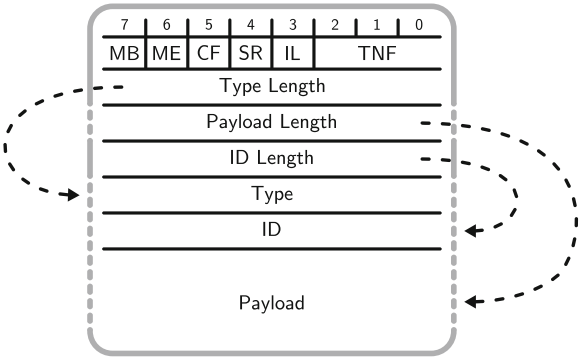

Fig. 2.2 Layout of an NDEF record (*Source* [29])

4. *Short Record* (*SR*): SR defines the size of the Payload Length field. If SR is set, the payload length is a 1-byte unsigned integer, otherwise it is a 4-byte unsigned integer. This flag is useful to reduce the memory consumption of short records.
5. *ID-Length Present* (*IL*): The IL flag, if set, specifies that the optional ID field and its corresponding length field are present.

The flags are followed by a 3-bit type classification field (Type Name Format, TNF). The value of the TNF field determines the interpretation of the Type field:

- *Empty* (`0x0`): The record is empty. The fields Type, ID and Payload are not present and their length fields are set to zero.
- *Well-known Type* (`0x1`): The Type field contains the relative Uniform Resource Identifier (URI) of an NFC Forum well-known type according to the *NFC Record Type Definition* (RTD, [31]).
- *Media Type* (`0x2`): The Type field contains a Multipurpose Internet Mail Extensions (MIME) media type identifier according to RFC 2046 [12].
- *Absolute URI* (`0x3`): The Type field contains an absolute URI according to RFC 3986 [1].
- *External Type* (`0x4`): The Type field contains the relative URI of an NFC Forum external type according to the *NFC Record Type Definition* (RTD, [31]).
- *Unknown* (`0x5`): The record contains data in an unknown format. No type information is present and the length of the Type field is zero.
- *Unchanged* (`0x6`): The record continues the payload of the preceding chunked record. No type information is present and the length of the Type field is zero.
- *Reserved* (`0x7`): This TNF value is reserved for future use.

The remaining header fields are the length information for the fields of variable length (Type Length, Payload Length, and ID Length), the type identification field (Type) and the optional record identifier (ID). The ID field may be used to specify a URI as a unique identifier for each record. This identifier can be used to cross-reference between multiple records.

Fig. 2.3 Multiple NDEF records form an NDEF message (based on [29])

The Payload field carries the actual data. The data is formatted and interpreted according to the type information in the Type field. If, for instance, the Type field specifies the MIME media type "text/x-vcard", then the payload is an electronic business card in the vCard format.

A data packet can be divided into multiple record chunks. In this case, the first record contains the type information and the optional record ID. The remaining chunks do not carry this information. Instead, their TNF field is set to `0x6` ("unchanged"). Except for the last chunk, every record chunk has its CF set, indicating that the payload is continued in the next record.

2.2.4.2 NDEF Message

Figure 2.3 shows the layout of an NDEF message. An NDEF message consists of one or more NDEF records. The first record of an NDEF message has its MB flag set. The last record has its ME flag set. The special case of an empty NDEF message is encoded by a single NDEF record with both, MB and ME set and with the TNF `0x0` ("empty").

2.2.5 NFC Record Type Definition (RTD)

The *NFC Record Type Definition* (RTD, [31]) defines two namespaces for NDEF record types: the NFC Forum well-known types and the NFC Forum external types.

NFC Forum well-known types are reserved for specifications of the NFC Forum. The type name is a Uniform Resource Name (URN) of the form "urn:nfc:wkt:<NAME>", where <NAME> identifies the type. To save storage space, the prefix "urn:nfc:wkt:" is not included into the Type field. Well-known type names can be either global or local. Global types start with an upper-case letter and have the same meaning regardless of their context. Local type names start with either a lower-case letter or a digit. They are defined for a specific context and are only valid within that context.

NFC Forum external types are reserved for self-allocation of global type names by organizations [31]. The type name is a URN of the form "urn:nfc:ext:<DOMAIN>:<NAME>", where <DOMAIN> is the issuing organizations Internet domain name and <NAME> identifies the type within that organization's namespace. In order to

Table 2.4 Format of a text record payload (based on [33])

Field	Offset[a]	Size	Coding of field content
Status byte	0	1 byte	Bit 7: Encoding of the text field (0: UTF-8, 1: UTF-16)
			Bit 6: Reserved for future use
			Bit 5..0: Length n of the language code
Language code	1	n bytes	ISO/IANA language code (US-ASCII encoded)
Text	$1 + n$	m bytes	Actual text (UTF-8/UTF-16 encoded)

[a]In bytes

save storage space, the prefix "urn:nfc:ext:" is not included into the Type field. As opposed to well-known type names, external types are case-insensitive.

The NFC Forum has defined a set of well-known types. These specifications cover primitive data types as well as complex data structures for specific use-cases.

2.2.5.1 Text Record Type

The Text Record Type Definition [33] specifies a record format for free-form text with language and encoding information. The text record has the well-known type name "urn:nfc:wkt:T". Table 2.4 lists the payload format of a text record. The payload consists of a status byte, a language code and the actual text. The language code can be used to choose one out of multiple text records that best-fits a user's language preferences.

2.2.5.2 URI Record Type

The URI Record Type Definition [34] specifies a record format for URIs. Thus, the URI record can be used to store website addresses, e-mail addresses, telephone numbers, SMS messages and other information that can be represented by URIs. The URI record has the well-known type name "urn:nfc:wkt:U". Table 2.5 lists the payload format of a URI record. The payload consists of an identifier code and a URI field. The identifier code is used to reduce the size of the URI by truncating common prefixes from the URI that is stored in the URI field. Table 2.6 lists the most common identifier codes. For example, the URI "http://www.mroland.at/" could be truncated to the URI field "mroland.at/" and the identifier code `0x01`.

Table 2.5 Format of a URI record payload (based on [34])

Field	Offset[a]	Size	Coding of field content
Identifier code	0	1 byte	Prefix code for compressing the actual URI
URI field	1	n bytes	Remaining URI (UTF-8 encoded)

[a]In bytes

Table 2.6 Common identifier codes and their URI prefixes (based on [34])

Identifier code	URI prefix
0x00	No prefix, the URI field contains the full URI
0x01	"http://www."
0x02	"https://www."
0x03	"http://"
0x04	"https://"
0x05	"tel:"
0x06	"mailto:"

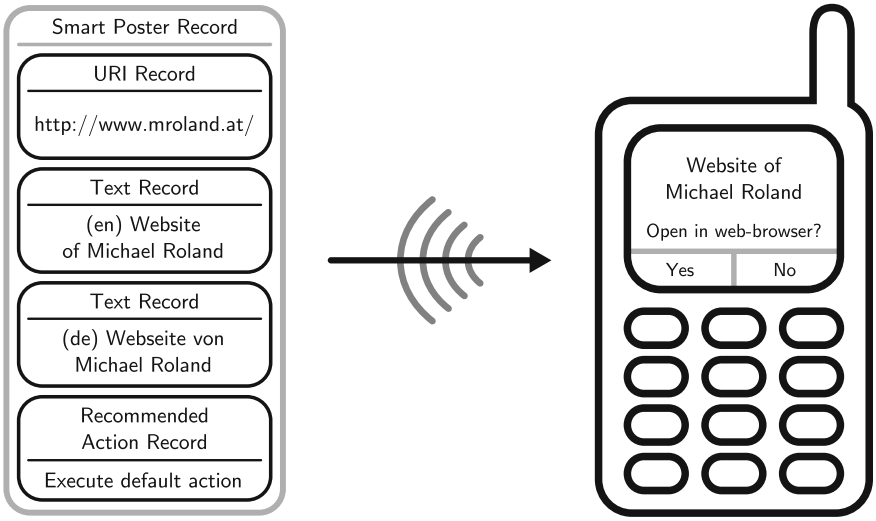

Fig. 2.4 Example of a smart poster record (based on [29])

2.2.5.3 Smart Poster Record Type

While many NFC devices associate a default action with stand-alone URI records (e.g. website addresses are opened in a web-browser), the URI Record Type Definition only specifies a container format. Instead, the NFC Forum created the *Smart Poster Record Type Definition* [32] to extend URI records with additional functionality.

A smart poster record (see Fig. 2.4) has the well-known type name "urn:nfc:wkt:Sp". Its payload is an NDEF message that consists of one URI record and optionally one or more other records. These other records can be text records that describe the URI in one or more languages, icon records that contain graphics that should be displayed together with the text, size and type information of the data referenced by the URI and a recommended action that should be performed with the URI. The Smart Poster Record Type Definition specifies three types of recommended actions:

1. Execute the default action associated with the given URI.
2. Store the URI for later use.
3. Open the URI in an appropriate editor.

Typical applications of smart poster records are advertisements with active content. Such an NFC-enabled advertisement could, for instance, link to a website or contain a ready-made SMS message for purchasing tickets [29].

2.2.5.4 Connection Handover Reference Application

The *Connection Handover* reference application [35] provides a means of using NFC as an enabler for other communication technologies (e.g. Bluetooth or Wi-Fi). The connection handover specification defines record types, message structures and handshake protocols for establishing a link through virtually any alternative carrier.

2.2.6 Card Emulation

There are several different options for performing card emulation with an NFC device. Card emulation could be performed by a secure element or by the software on the application processor ("host processor") of the device itself.

2.2.6.1 Secure Element

A secure element (SE) is a smartcard microchip that is integrated into an NFC device and connected to the NFC controller. In card emulation mode, the NFC controller routes all communication to the secure element.

A secure element can be a dedicated microchip that is embedded into the NFC device (embedded SE). Such a chip could also be combined into a single package with the NFC controller. An example for such a combined chip module is NXP's PN65N which contains a PN544 NFC controller and a secure element from NXP's SmartMX series. Another possibility is the combination of the secure element functionality with another smartcard/security device that is used within the NFC device. For instance, a UICC (also known as the SIM card) is a smartcard that is already present in many NFC devices (particularly in NFC-enabled mobile phones). Other security devices that are available equipped with smartcard technology are (micro) SD (secure digital) cards.

Many secure elements (e.g. NXP's SmartMX) are standard smartcard ICs as used for regular contact and contactless smartcards. They share the same hardware and software platforms. The only difference is the interface they provide: Instead of (or in addition to) a classic smartcard interface according to ISO/IEC 7816 (for contact cards) or an antenna (for contactless cards), the secure element has a direct interface

for the connection to the NFC controller. Such interfaces are the *NFC Wired Interface* (NFC-WI, [5]) and the *Single Wire Protocol* (SWP, [9]).

2.2.6.2 Software Card Emulation

Typical use-cases for card emulation are security critical applications such as access control and payment. For these applications the secure element provides secure storage, a secure execution environment and hardware-based support for cryptographic operations. Therefore, emulation by software on a non-secure application processor was not widely used in the past. Nevertheless, several NFC controllers and NFC devices have support for software card emulation.

Software card emulation or host-based card emulation (sometimes also referred to as "soft-SE" [11]) was first made available on NFC-enabled mobile phones by BlackBerry on their BlackBerry 7 platform. A BlackBerry application can emulate NFC Forum type 4 tags and ISO/IEC 14443-4 smartcards. For an NFC Forum type 4 tag, the application simply needs to define an NDEF message that is used by the operating system to emulate the virtual tag. For an ISO/IEC 14443-4 smartcard, the application receives the command APDUs sent by the smartcard terminal and generates corresponding response APDUs.

BlackBerry mobile phones were the first mobile phones known to support software card emulation. However, patches [42, 43] to the CyanogenMod aftermarket firmware for Android devices brought software card emulation to Android devices with NXP's PN544 NFC controller. Starting with Android 4.4, software card emulation became available on most Android NFC devices under the term *host-based card emulation* (HCE). Besides that, some NFC readers (e.g. ACS ACR 122U) can be used to perform software card emulation on PC platforms.

2.3 EMV

EMV is a series of standards for chip-based credit and debit cards. The EMV standards were initially created by Europay, MasterCard and Visa (hence the acronym EMV) in an effort to design a worldwide standard for chip-based payment cards and payment terminals [8]. Today, the EMV standards are maintained by EMVCo, an organization owned by the credit card companies American Express, JCB, MasterCard and Visa [8].

The EMVCo website [8] names several advantages of EMV chip-based payment cards in comparison to chip-less cards (e.g. magnetic stripe cards):

- EMV cards prevent fraud because the smartcard chip can contain data that is close to impossible to be cloned while magnetic stripes can easily be copied.
- A smartcard is capable of computing unique digital signatures/authentication codes for each transaction in both, online and offline environments.

- Smartcards support enhanced cardholder verification methods (e.g. offline PIN verification).
- A smartcard chip has significantly more data storage than a magnetic stripe card.

The *EMV Integrated Circuit Card Specifications for Payment Systems* [7] are based on ISO/IEC 7816 smartcard standards. An EMV payment card (also referred to as *Chip & PIN* card) can support various online and offline transaction modes. Possible security measures include authenticating the card to the payment terminal, authenticating the payment terminal to the card and authenticating the cardholder to the card. Besides interaction between the terminal and the smartcard, they also define test methods, secure key management, the interface between the payment terminal and its users and the interface between the payment terminal and the bank that manages the payment.

EMVCo does not only support contact-based smartcards. Instead, they extended their scope to contactless and mobile payment systems. The *EMV Contactless Specifications for Payment Systems* [6] are a series of standards for contactless payment systems based on ISO/IEC 14443 and ISO/IEC 7816. The EMV specifications for contactless payment systems are a mere aggregation of four different payment systems (one by JCB, one by MasterCard, one by Visa and one by American Express). For each of these payment systems, the terminal has a separate software module that processes transactions.

MasterCard's EMV-compliant contactless payment system is called MasterCard PayPass. It supports two different operating modes: emulation of the magnetic stripe system (*EMV Mag-Stripe* mode) and *EMV mode*. In Mag-Stripe mode, the card stores information comparable to that on the magnetic stripe and generates dynamic authentication codes to authorize payments. As the authorization codes do not contain any information about the payment transaction, this mode is for online transactions only. In EMV mode, the card authenticates itself to the terminal and signs the payment transaction, making it possible to verify and store transactions offline for later processing.

Other EMV-compliant contactless payment systems are Visa's payWave and American Express's ExpressPay.

References

1. Berners-Lee, T., Fielding, R.T., Masinter, L.: RFC 3986: Uniform Resource Identifier (URI): Generic Syntax. Internet Engineering Task Force (2005)
2. Chen, E.: NFC: Short range, long potential. Assa Abloy FutureLab News (2007). http://www.assaabloyfuturelab.com/FutureLab/Templates/Page2Cols____1905.aspx
3. Ecma International: ECMA-352: Near Field Communication Interface and Protocol-2 (NFCIP-2) (2003)
4. Ecma International: ECMA-340: Near Field Communication Interface and Protocol (NFCIP-1) (2004)
5. Ecma International: ECMA-373: Near Field Communication Wired Interface (NFC-WI) (2006)

6. EMVCo: EMV Contactless Specifications for Payment Systems (Books A-D). Version 2.1 (2011)
7. EMVCo: EMV Integrated Circuit Card Specifications for Payment Systems (Books 1–4). Version 4.3 (2011)
8. EMVCo: About EMV (2012). http://www.emvco.com/about_emv.aspx
9. European Telecommunications Standards Institute: ETSI TS 102 613: Smart Cards; UICC—Contactless Front-end (CLF) Interface; Part 1: Physical and data link layer characteristics (Release 11) (2012)
10. Finkenzeller, K.: RFID-Handbuch, 4th edn. Carl Hanser Verlag, München (2006)
11. Francis, L., Hancke, G.P., Mayes, K.E., Markantonakis, K.: Practical Relay Attack on Contactless Transactions by Using NFC Mobile Phones. Cryptology ePrint Archive, Report 2011/618 (2011). http://eprint.iacr.org/2011/618
12. Freed, N., Borenstein, N.S.: RFC 2046: Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types. Internet Engineering Task Force (1996)
13. GlobalPlatform: Card Specification. Version 2.2.1 (2011)
14. International Organization for Standardization: ISO/IEC 15693–1: Identification cards—Contactless integrated circuit(s) cards—Vicinity cards—Part 1: Physical characteristics (2000)
15. International Organization for Standardization: ISO/IEC 15693–2: Identification cards—Contactless integrated circuit(s) cards—Vicinity cards—Part 2: Air interface and initialization (2000)
16. International Organization for Standardization: ISO/IEC 15693–3: Identification cards—Contactless integrated circuit(s) cards—Vicinity cards—Part 3: Anticollision and transmission protocol (2001)
17. International Organization for Standardization: ISO/IEC 18092: Information technology—Telecommunications and information exchange between systems—Near Field Communication—Interface and Protocol (NFCIP-1) (2004)
18. International Organization for Standardization: ISO/IEC 21481: Information technology—Telecommunications and information exchange between systems—Near Field Communication Interface and Protocol-2 (NFCIP-2) (2005)
19. International Organization for Standardization: ISO/IEC 7816–12: Identification cards—Integrated circuit cards—Part 12: Cards with contacts—USB electrical interface and operating procedures (2005)
20. International Organization for Standardization: ISO/IEC 7816–4: Identification cards—Integrated circuit cards—Part 4: Organization, security and commands for interchange (2005)
21. International Organization for Standardization: ISO/IEC 7816–3: Identification cards—Integrated circuit cards—Part 3: Cards with contacts—Electrical interface and transmission protocols (2006)
22. International Organization for Standardization: ISO/IEC 7816–2: Identification cards—Integrated circuit cards—Part 2: Cards with contacts—Dimensions and location of the contacts (2007)
23. International Organization for Standardization: ISO/IEC 14443–1: Identification cards—Contactless integrated circuit cards—Proximity cards—Part 1: Physical characteristics (2008)
24. International Organization for Standardization: ISO/IEC 14443–4: Identification cards—Contactless integrated circuit cards—Proximity cards—Part 4: Transmission protocol (2008)
25. International Organization for Standardization: ISO/IEC 14443–2: Identification cards—Contactless integrated circuit cards—Proximity cards—Part 2: Radio frequency power and signal interface (2010)
26. International Organization for Standardization: ISO/IEC 14443–3: Identification cards—Contactless integrated circuit cards—Proximity cards—Part 3: Initialization and anticollision (2011)
27. International Organization for Standardization: ISO/IEC 7816–1: Identification cards—Integrated circuit cards—Part 1: Cards with contacts—Physical characteristics (2011)
28. Japanese Standards Association: JIS X 6319–4: Specification of implementation for integrated circuit(s) cards—Part 4: High Speed proximity cards (2005)

29. Langer, J., Roland, M.: Anwendungen und Technik von Near Field Communication (NFC). Springer, Heidelberg (2010)
30. NFC Forum: NFC Data Exchange Format (NDEF). Technical specification, version 1.0 (2006)
31. NFC Forum: NFC Record Type Definition (RTD). Technical specification, version 1.0 (2006)
32. NFC Forum: Smart Poster Record Type Definition. Technical specification, version 1.0 (2006)
33. NFC Forum: Text Record Type Definition. Technical specification, version 1.0 (2006)
34. NFC Forum: URI Record Type Definition. Technical specification, version 1.0 (2006)
35. NFC Forum: Connection Handover. Technical specification, version 1.1 (2008)
36. NFC Forum: Type 1 Tag Operation Specification. Technical specification, version 1.1 (2011)
37. NFC Forum: Type 2 Tag Operation Specification. Technical specification, version 1.1 (2011)
38. NFC Forum: Type 3 Tag Operation Specification. Technical specification, version 1.1 (2011)
39. NFC Forum: Type 4 Tag Operation Specification. Technical specification, version 2.0 (2011)
40. Rankl, W., Effing, W.: Handbuch der Chipkarten, 4th edn. Carl Hanser Verlag, München (2002)
41. Roland, M.: Software Card Emulation in NFC-enabled Mobile Phones: Great Advantage or Security Nightmare? In: 4th International Workshop on Security and Privacy in Spontaneous Interaction and Mobile Phone Use. Newcastle, UK (2012). http://www.medien.ifi.lmu.de/iwssi2012/papers/iwssi-spmu2012-roland.pdf
42. Yeager, D.: Added NFC Reader support for two new tag types: ISO PCD type A and ISO PCD type B (2012). https://github.com/CyanogenMod/android_packages_apps_Nfc/commit/d41edfd794d4d0fedd91d561114308f0d5f83878
43. Yeager, D.: Added NFC Reader support for two new tag types: ISO PCD type A and ISO PCD type B (2012). https://github.com/CyanogenMod/android_external_libnfc-nxp/commit/34f13082c2e78d1770e98b4ed61f446beeb03d88

http://www.springer.com/978-3-319-15487-9

Security Issues in Mobile NFC Devices

Roland, M.

2015, XVIII, 185 p. 44 illus., 17 illus. in color., Hardcover
ISBN: 978-3-319-15487-9