NT4H2421Tx

NTAG 424 DNA TT – Secure NFC T4T compliant IC with Tag Tamper feature

Rev. 3.0 — 31 January 2019
465530

Product data sheet
COMPANY PUBLIC

1 General description

1.1 Introduction

NTAG 424 DNA TT (NT4H2421Tx) sets a new standard in secure NFC and IoT applications, introducing a new NTAG DNA chip generation with state-of-the-art features for security, status detection and privacy protection. It comes with AES-128 cryptographic operation and a new Secure Unique NFC Message (SUN) feature to generate tap-unique data authentication upon each read-out by an NFC enabled mobile device. This enables most advanced product protection in terms of product authenticity and integrity, as well a secure exclusive user experiences based on product status served in real time.

NTAG 424 DNA TT is fully compliant with the NFC Forum Type 4 Tag specification (Certification ID: 58569), with the contactless proximity protocol according to ISO/IEC14443-4 and the ISO/IEC 7816-4 based file system and command frames, to ensure maximum interoperability within the NFC infrastructure. Its NFC performance supports superior user interaction and an operating distance of up to 10 cm.

Using AES-128 cryptography, the tag generates a unique NFC authentication message (SUN) each time it is being tapped. An NFC mobile device reads this tap-unique URL with the SUN authentication message, sends it to the host where tag and message authentication take place, and returns the verification result. The SUN authentication mechanism is working on Android without a dedicated application and from iOS11 onwards using an application. This way, NTAG 424 DNA TT offers tag authentication, as well as data assurances on authenticity, integrity and even confidentiality, while also securing physical tag presence, see also Section 9.3.

In addition, NTAG 424 DNA TT comes with smart status awareness, detecting the status of a tamper loop. Users can instantly detect the once-opened status, by just reading the tag with an NFC enabled device. The chip irrevocably stores this event, and mirrors it into the NDEF message during startup.

The chip has a file-based memory structure of totally 416 bytes compliant to NFC Forum Type 4 Tag and ISO/IEC 7816-4 with a Capability Container (CC) file to specify the NFC Forum tag operation, an NDEF file as well as an extra data file to protect sensitive content. Configurable access rights per file support different use cases of brand product manufacturers and service providers to meet security and operational requirements. With 5 customer defined AES keys, NTAG 424 DNA TT enables advanced cryptographic functionality for CMAC, optionally combined with encrypted data, for SUN, mutual authentication (secure host or reader authentication) and for secure access to the NDEF file and the extra data file.

NTAG 424 DNA TT contains configurable features like optionally encrypting part of the NDEF file, and the fully encrypted communication mode to address privacy sensitive applications. The optional Random ID together with the encrypted chip UID/data that

can be mirrored in the NDEF file, enables compliance with latest user data protection regulations. Also the tamper status can be encrypted to add extra security.

Besides the standard AES-128 implementation, NTAG 424 DNA TT also offers an alternative AES-based protocol for authentication and secure messaging using a Leakage Resilient Primitive, or LRP, a wrapper around the AES cryptography to enhance side-channel attack resistance.

Thanks to its high input capacitance of 50 pF, NTAG 424 DNA TT is well suited for smart product applications requiring smaller footprint antennas, without compromise on performance. Smaller NFC tags can more easily be embedded in product labels, caps and closures, and other form factors.

2 Features and benefits

2.1 RF Interface & Communication Protocol

- Fully compliant to the NFC Forum Tag 4 Type technical specification [14]
- Fully compliant to NDEF data structure configurations [15]
- Contactless interface compliant to ISO/IEC 14443A-2/ -3/ -4, see [1] , [2], [3]
- Support of ISO/IEC 7816-4 communication frames for highest interoperability with mobile and wearables
- Low power consumption (Hmin) enabling operating distances of up to 10 cm
- Support of fast data rates: 106 kbit/s, 212 kbit/s, 424 kbit/s, and 848 kbit/s
- Support of double size (7-byte) Unique Identifiers (UID) and optionally Random ID (RID) according to ISO 14443-3 [2]
- Communication frame size to support up to 128 bytes
- Support of ISO 7816-4 wrapped commands

2.2 Memory Organization

- 416 bytes user memory
- Data retention of 50 years and write endurance of minimum 200.000 cycles
- File system compliant to ISO/IEC 7816-4 with one predefined Directory File (DF) and a set of Elementary Files (EF)
  - Three standard data files, one with 32 byte for the capability file, one with 256 bytes for NDEF storage and one with 128 bytes for protected data
- File system compliant with ISO 7816

2.3 Security and Privacy

- Common Criteria certification: EAL4 for both Hardware and Software
- Secure Unique NFC (SUN) message featuring integrity protection, authenticity and confidentiality
- The SUN feature is enabled by the Secure Dynamic Messaging (SDM) which is mirrored as text (ASCII encoded) into the NDEF message
- Incremental NFC Counter, which counts each tap
- Secure messaging compliant to standard AES according to NIST Special Publication 800-38A and 800-38B [5] [6]

  - Optional enhanced side channel attack protection using LRP wrapped AES operation according to [10]
- Five customer defined AES 128-bit keys including key versions
- Optional Random ID for enhanced privacy
- 3-pass mutual authentication
- Flexible access control configurable per file (EF)
  - Individual key configuration for Read (R) / Write (W) / ReadWrite (RW) / Configuration
- Configurable secure messaging communication mode
  - Plain communication
  - CMAC protected for message integrity protection
  - Full Enciphered plus CMAC for full encryption of complete data transferred through contactless interface
- ECC-based NXP originality signature
- AES-based originality keys leveraging the LRP wrapped AES authentication

2.4 Specific Features

- Tamper detection and secure mirroring into the NDEF message
- Frame-level oriented automatic anti-tearing mechanism
- High input capacitance (50 pF) for small form factor design

3 Applications

NTAG 424 DNA TT offers multi-layered security to enable a broad range of trusted applications that protect products, services and user experiences.

- **Advanced anti-counterfeiting**
  Verify authenticity of physical goods and identify sales outside authorized markets
- **Tamper detection and proof**
  Detect unauthorized product opening to secure product integrity
- **Secured exclusive user experiences based on product status**
  Enabled advanced user interactions, pre- and post-sale. Also reward customers with truly exclusive and personalized content, offers and privileges
- **Secured sensitive data applications**
  Protect sensitive product and user data, or trigger an action upon a verified incidence, e.g., payment
- **Secure authentication and configuration of closed loop devices**
  Authenticate consumables and parts, enable an integrity check, and automatically transfer device settings

4 Ordering information

Table 1. Ordering information

Type number	Package	Description	Version
NT4H2421TTDUD/02	FFC	8 inch wafer (sawn; 120 µm thickness; Au Bumps) [1] [2] 256 Byte NDEF Message, 128 Byte User Memory, C_i = 50 pF	-
NT4H2421TTDUF/02	FFC	8 inch wafer (sawn; 75 µm thickness; Au Bumps) [1] [2] 256 Byte NDEF Message, 128 Byte User Memory, C_i = 50 pF	-
NT4H2421TSDUD/02	FFC	Service version with preprogrammed NDEF message and keys 8 inch wafer (sawn; 75 µm thickness; Au Bumps) [1] [2] 256 Byte NDEF Message, 128 Byte User Memory, C_i = 50 pF	-
NT4H2421TSDUF/02	FFC	Service version with preprogrammed NDEF message and keys 8 inch wafer (sawn; 75 µm thickness; Au Bumps) [1] [2] 256 Byte NDEF Message, 128 Byte User Memory, C_i = 50 pF	-
NT4H2421TCDUD/02	FFC	Customer configurable version 8 inch wafer (sawn; 75 µm thickness; Au Bumps) [1] [2] 256 Byte NDEF Message, 128 Byte User Memory, C_i = 50 pF	-
NT4H2421TCDUF/02	FFC	Customer configurable version 8 inch wafer (sawn; 75 µm thickness; Au Bumps) [1] [2] 256 Byte NDEF Message, 128 Byte User Memory, C_i = 50 pF	-

[1] Delivered on film frame carrier with electronic fail die marking according to SECSII format.
[2] See [11]

5 Quick reference data

Table 2. Quick reference data

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
C_i	input capacitance		[1]	45.0	50	55.0	pF
f_i	input frequency			-	13.56	-	MHz
EEPROM characteristics							
t_{ret}	retention time	T_{amb} = 22 °C		50	-	-	year
$N_{endu}(W)$	write endurance[2]	T_{amb} = 22 °C		200.000	-	-	cycle
$t_{cy}(W)$	write cycle time	T_{amb} = 22 °C		-	1	-	ms

[1] T_{amb} = 22 °C; f_i = 13.56 MHz; 2 V RMS
[2] Write endurance of a single EEPROM cell

6 Block diagram

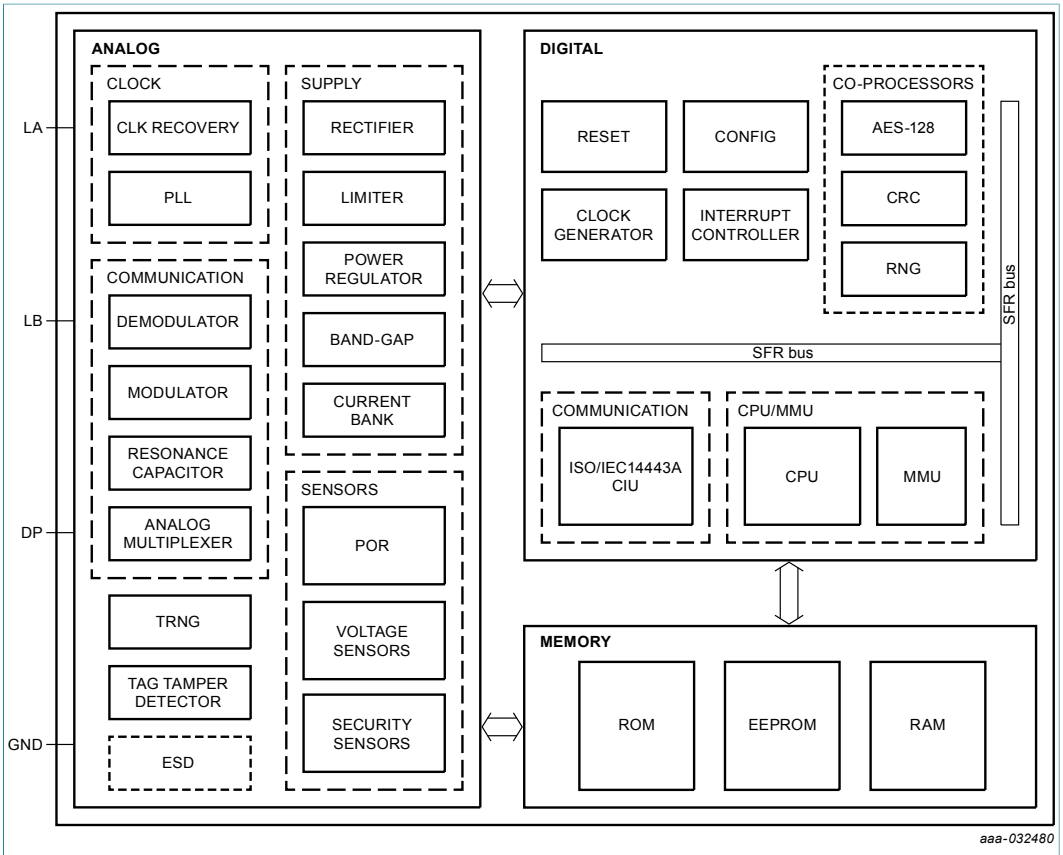


Figure 1. NTAG 424 DNA TT IC blocks

7 Pinning information

7.1 Pinning

The pinning for the NT4H2421Tx is shown in Figure 2 for a die.

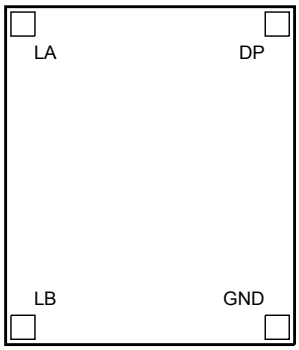


Figure 2. Pin configuration

Table 3. Pin allocation table

Pin	Symbol	
LA	LA	antenna coil connection LA
LB	LB	antenna coil connection LB
DP	DP	Detection Pin
GND	GND	Ground

8 Functional description

8.1 Interface initialization and protocol

NT4H2421Tx is fully compliant to ISO/IEC 14443-2 [1] radio frequency power and signal interface, the initialization and anti-collision is according to ISO/IEC 14443-3 [2] and it uses transmission protocol as specified in ISO/IEC 14443-4 [3] of PICC Type A.

8.1.1 ISO/IEC 14443 parameter values

This section describes the values for ISO/IEC 14443 activation and selection. Usage of Random ID can be changed using the SetConfiguration command.

Note, that any change in the ISO/IEC 14443 parameter values through SetConfiguration requires a power cycle to make those changes effective.

ATQA

ATQA value is 0344h, which denotes double size (7-byte) UID. However, NT4H2421Tx offers configuration of Random ID which is single size (4-byte). If the Random ID feature is enabled, then the ATQA is changed to 0304h. According to ISO/IEC 14443-3, the ATQA bytes are transmitted as LSB first.

SAK

For double size UID, the value of SAK1 in cascade level 1 is 04h, indicating that the UID is not complete. SAK2 in cascade level 2 is 20h, indicating UID complete and supporting ISO/IEC 14443-4. For single size UID which is used in the Random ID case, the value of SAK is 20h, indicating UID complete and supporting ISO/IEC 14443-4.

UID

The ISO/IEC 14443-3 compliant UID is programmed and locked during production. The first byte of the double size UID is fixed to 04h, indicating NXP as manufacturer.

ATS

The value of the ATS of NT4H2421Tx is as follows:

Table 4. ATS value

ATS Parameter	Value	Comment
TL	06h	Length of ATS
T0	77h	TA(1), TB(1), TC(1) present in ATS and frame size is 128 bytes
TA(1)	77h	Different communication speed can be set in each direction supports communication speeds 212, 424, 848 kbps in both directions
TB(1)	71h	Max frame waiting time is 38.66 ms, start frame guard time is 604 µs
TC(1)	02h	CID supported
T1	80h	Historical byte

8.1.2 Setting of higher communication speed

After receiving an ATS, a PPS request can be sent to the NT4H2421Tx to set up a higher communication speed up to 848 kbit/s according to ISO/IEC 14443-4 [3].

8.1.3 Half-duplex block transmission protocol

NT4H2421Tx uses half-duplex block transmission protocol as specified in ISO/IEC 14443-4. It is fully compliant to block format, frame waiting time, frame waiting time extension, protocol operation, and all rules or handling as in [3].

8.2 User memory

The file system in the user memory is according to ISO/IEC 7816-4 and shown in Figure 3. In the DF (application), there are 3 EFs (files) and 5 keys.

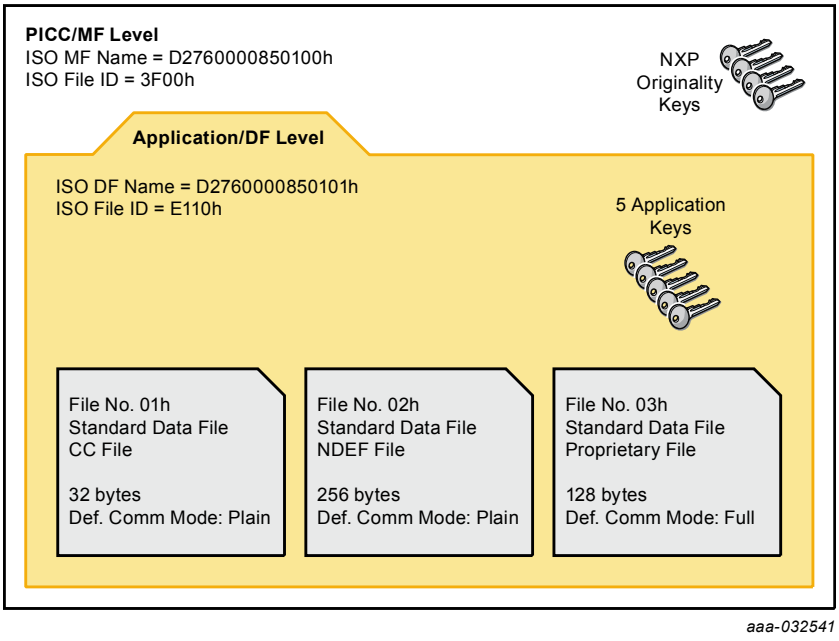


Figure 3. NTAG 424 DNA TT application

8.2.1 Application and file selection

The MF (PICC Level), the DF (application) and the EFs (files) can be selected using the ISOSelectFile command specified in ISO/IEC 7816-4 [4] and described in Section 11.10.1.

The PICC level refers to the card itself and has the ISO DF Name D2760000850100h and the File ID 3F00h. The only functionality available on PICC level is an LRP mode authentication using an OriginalityKey and the retrieval of the originality signature using the Read_Sig command, see Section 11.11.1.

8.2.2 Application Name and ID

NT4H2421Tx is pre-configured with one application. The DF name and File ID are as follows:

- DF Name: D2760000850101h
- File Identifier: E110h

8.2.3 Files

NT4H2421Tx stores user data into EF (files) of specific types. All files are statically created and cannot be deleted. The ChangeFileSettings command can be used to change the access rights configuration.

File access rights can be restricted with the keys of the application.

Table 5. File management

EF (File) Type	File type coding	File no.	File ID	File size	Example uses
StandardData file	00h	01h	E103h	32 bytes	CC file
		02h	E104h	256 bytes	NDEF message, SDM and mirroring supported
		03h	E105h	128 bytes	Proprietary file, storage of raw data

8.2.3.1 StandardData file

A StandardData file stores the data as raw data bytes. Data is accessed by chunk of byte at a certain offset in the StandardData file and with a certain byte length.

A StandardData file can be read with the ReadData and ISOReadBinary commands. The data can be written with the WriteData and ISOUpdateBinary commands. ISOReadBinary and ISOUpdateBinary are standard ISO/IEC 7816-4 interindustry commands, see [4].

A StandardData file is not covered by the transaction mechanism and therefore does not implement a transaction-based backup mechanism. Thus data are available to read as soon as they are written in the file. The writing operations of single frames up to 128 bytes with a WriteData or ISOUpdateBinary command are also tearing protected.

NT4H2421Tx is configured to hold the NDEF message in StandardData file No 02h, see Section 8.2.3. Enabling SDM is only possible for this file. At delivery, SDM is disabled.

8.2.3.2 Capability Container File

The Capability Container (CC) file is a StandardData file with respect to access rights management and data management. This file will hold the CC-file according to [14]. At delivery it will hold following content:

- CCLEN = 0017h, i.e. 23 bytes
- T4T_VNo = 20h, i.e. Mapping Version 2.0
- MLe = 0100h, i.e. 256 bytes
- MLc = 00FFh, i.e. 255 bytes
- NDEF-File_Ctrl_TLV
  - T = 04h, indicates the NDEF-File_Ctrl_TLV
  - L = 06h, i.e. 6 bytes
  - NDEF-File File Identifier = E104h
  - NDEF-File File Size = 0100h, i.e. 256 bytes

  - NDEF-File READ Access Condition = 00h, i.e. READ access granted without any security
  - NDEF-File WRITE Access Condition = 00h, i.e. WRITE access granted without any security
- Proprietary-File_Ctrl_TLV
  - T = 05h, indicates the Proprietary-File_Ctrl_TLV
  - L = 06h, i.e. 6 bytes
  - Proprietary-File File Identifier = E105h
  - Proprietary-File File Size = 0080h, i.e. 128 bytes
  - Proprietary-File READ Access Condition = 82h, i.e. Limited READ access, granted based on proprietary methods, after authentication with key 2h.
  - Proprietary-File WRITE Access Condition = 83h, i.e. Limited READWRITE access, granted based on proprietary methods, after authentication with key 3h.

8.2.3.3 File access rights management

File data is accessed with 3 different access rights Read, Write and ReadWrite.

In addition, an access right called Change is specified per file permitting ChangeFileSettings to change the file access rights.

An access right is granted if at least one condition associated to it is satisfied. Such conditions are called access conditions. Access conditions are associated with an access right and evaluated to decide whether the access right is granted or not. There are three kinds of access conditions:

- The access condition where a valid authentication with a given AppKey of the targeted application is needed to access related commands listed in Table 9. The access condition is satisfied if the current valid authentication has been performed with the given AppKey and not satisfied in all other cases. The AppKey is specified with its number.
- The free access condition meaning the related commands listed in Table 9 can be accessed without an active authentication.
- The no access condition meaning no access to related commands listed in Table 9.

The access conditions are specified on 4 bits as defined in the following table.

Table 6. Access condition

Condition value	Description
0h..4h	key number of a AppKey
Eh	free access
Fh	no access or RFU

The set of access conditions is coded on 2 bytes as shown in the following table. RFU access conditions are expected to be set to Fh (for future extensibility).

Table 7. Set of Access condition

Bit index	Description	Value
15..12	Read	access condition as in Table 6
11..8	Write	access condition as in Table 6
7..4	ReadWrite	access condition as in Table 6

Bit index	Description	Value
3..0	Change	access condition as in Table 6

The default access conditions for the files in NT4H2421Tx is shown below in .

Table 8. Default file access rights

File No.	File Type	Read	Write	ReadWrite	Change
01h	StandardData File	Eh	0h	0h	0h
02h	StandardData File	Eh	Eh[1]	Eh[1]	0h
03h	StandardData File	2h	3h	3h	0h

[1] Write and ReadWrite access rights for the NDEF File (File No. 02h) should be changed after personalization in order to prevent unauthorized changes in the NDEF File.

The mapping of access rights to applicable commands is shown in Table 9.

Table 9. Command list associated with access rights

Access Right	Command
Read	ReadData ISOReadBinary
Write	WriteData ISOUpdateBinary
ReadWrite	ReadData ISOReadBinary WriteData ISOUpdateBinary
Change	ChangeFileSettings
SDMMetaRead	-
SDMFileRead	ReadData ISOReadBinary
SDMCtrRet	GetFileCounters

A command listed in Table 9 is accepted if at least one access condition associated with an access right granting access to it is satisfied. If authenticated and the only access conditions satisfied are the free access Eh ones, then the CommMode.Plain is to be applied.

If not authenticated, Secure Dynamic Messaging will be applied if access is granted via SDMFileRead, even if there is free access via one of the other access rights. SDMFileRead is not affecting the regular secure messaging, i.e. if authenticated.

8.2.3.4 SDM related access rights

A StandardData file can be associated with the following Secure Dynamic Messaging access rights: SDMMetaRead, SDMFileRead and SDMCtrRet. SDMCtrRet is interpreted as the access rights defined according to Table 6 and grants access to the GetFileCounters command. The others have a different interpretation.

The SDMMetaRead access does not define access to certain commands, but it defines the mirroring of PICCData, i.e. whether the PICCData will be mirrored in plain, encrypted or not at all, see also Section 9.3.3. This is interpreted according to Table 10.

Note that it is still possible to enable plain PICCData mirroring of the UID even if Random ID is enabled. For privacy protection, it is strongly recommended to use only encrypted PICCData mirroring for the UID if Random ID is enabled.

Table 10. SDMMetaRead values

Condition value	Description
0h..4h	SDMMetaReadKey: key number of an AppKey used to encrypt the PICCData before mirroring
Eh	Plain PICCData mirroring
Fh	No PICCData mirroring

The SDMFileRead access right, if related with an AppKey, grants free access to ReadData and ISOReadBinary. The targeted AppKey is used for the Secure Dynamic Messaging, see also Section 9.3. SDMFileRead is interpreted according to Table 10.

Table 11. SDMFileRead values

Condition value	Description
0h..4h	SDMFileReadKey: free access, key number of an AppKey that is to be applied for the Secure Dynamic Messaging
Eh	RFU
Fh	No Secure Dynamic Messaging for Reading

Note that SDMFileRead is not influenced by Read or ReadWrite access rights on the NDEF file. If SDMFileRead is granted, the Secure Dynamic Messaging will always be applied in not authenticated state. If SDMFileRead access right is set to Fh, it is still possible to freely read the file if Read or ReadWrite access right are set to Eh. In this case, plain mirroring of the PICCData, see Section 9.3.3, is still applied if the card is configured for that.

8.2.3.5 Communication modes

NT4H2421Tx supports three communication modes as defined in Table 12. As shown in the table, the different communication modes can be represented by two bits. This representation is used at several places in the document.

Table 12. Supported communication modes

Communication mode	Bit Representation	Explanation
CommMode.Plain	X0b	No protection: message is transmitted in plain text
CommMode.MAC	01b	MAC protection for integrity and authenticity
CommMode.Full	11b	Full protection for integrity, authenticity and confidentiality, also referred to as "Full Protection" mode

The communication mode defines the level of security for the communication between PCD and PICC after mutual authentication. At application level, the communication

mode is defined by the command itself, as specified in Table 22. The specified communication mode is applied if there is an active authentication regardless of whether this authentication is required by the command or not.

At file level, the communication mode is defined by the file. The specified communication mode is applied if there is an active authentication. Note, if the only valid access condition for a certain access right is free access (Eh) under an active authentication, CommMode.Plain has to be applied, see also Section 8.2.3.3.

The commands for authentication have their own secure messaging rules, as indicated by N/A (not applicable) in Section 11.2. The ChangeKey command is always executed in Full Protection mode.

If there is no active authentication, the command and response are sent in plain (or the command is rejected in the case an authentication is required).

The default communication mode per file is shown in below.

Table 13. Default communication modes per file

EF (File) Type	File no.	Default communication mode
StandardData file	01h	CommMode.Plain
	02h	CommMode.Plain
	03h	CommMode.Full

8.2.4 Keys

Application keys and PICC keys and their usage are defined in Table 14 and Table 15 respectively. They are used to manage the security of the application.

Table 14. Keys at application level

Key Identifier	Key number	Change Key	Can be used for Authentication
Addressable keys:			
AppMasterKey	00h	AppMasterKey	yes
AppKey	00h..04h	AppMasterKey	yes
SDMMetaReadKey	00h..04h	AppMasterKey	yes
SDMFileReadKey	00h..04h	AppMasterKey	yes
TTStatusKey	00h..04h	AppMasterKey	yes

8.2.4.1 AppMasterKey

The AppMasterKey always has the key number 00h. A successful authentication with the AppMasterKey is required to change any application key including the AppMasterKey itself with the ChangeKey command.

8.2.4.2 AppKey

The application of the NT4H2421Tx includes 5 AES 128-bit keys with key numbers 0, 1, 2, 3, 4. Furthermore, key number "E" (means free) and key number "F" (means never) can be assigned for access rights management.

The transport value of these 5 keys is 16 bytes of 00h, and can be changed by authentication with key number 0 in transport configuration.

Remark: It is highly recommended to change all 5 keys at personalization, even if not all keys are used in the application.

8.2.4.3 SDMMetaReadKey

The SDMMetaReadKey is one of the 5 AppKey. Which key is used is configured via Section 11.7.1 by adjusting the SDMMetaRead access rights, see Section 8.2.3.4. SDMMetaReadKey is used to encrypt PICCData before mirroring.

As the SDMMetaReadKey refers to an AppKey, it is changeable with ChangeKey with an active authentication with the AppMasterKey.

As the SDMMetaReadKey refers to an AppKey, it is available for authentication.

8.2.4.4 SDMFileReadKey

The SDMFileReadKey is one of the 5 AppKey. Which key is used is configured via Section 11.7.1 by adjusting the SDMFileRead access rights, see Section 8.2.3.4. SDMFileReadKey is used for Secure Dynamic Messaging.

As the SDMFileReadKey refers to an AppKey, it is changeable with ChangeKey with an active authentication with the AppMasterKey.

As the SDMFileReadKey refers to an AppKey, it is available for authentication.

8.2.4.5 TTStatusKey

The TTStatusKey is one of the 5 AppKey. Which key is used is configured via SetConfiguration Option 07h, when enabling the Tag Tamper Protection feature. Once configured to an AppKey, authentication with this TTStatusKey will be required for GetTTStatus, see Section 11.9.1.

As the TTStatusKey refers to an AppKey, it is changeable with ChangeKey with an active authentication with the AppMasterKey.

As the TTStatusKey refers to an AppKey, it is available for authentication.

8.2.4.6 OriginalityKey

The authentication procedure for AES keys can be used to authenticate to one of the four OriginalityKey and check whether the PICC is a genuine NXP product. NT4H2421Tx supports targeting the OriginalityKey with the LRP authentication using AES. For details on the authentication command, see Section 9.2. The following variants can be used:

- AuthenticateLRPFirst, see Section 9.2.5
- AuthenticateLRPNonFirst, see Section 9.2.6

Table 15. Keys at PICC level

Key Identifier	Key number	Can be used for Authentication
Addressable keys:		
OriginalityKey1	01h	yes
OriginalityKey2	02h	yes
OriginalityKey3	03h	yes
OriginalityKey4	04h	yes

8.2.4.7 Key version

A 1-byte key version (00h to FFh) is assigned to each key. This key version can be used to distinguish the keys of a specific application if the same application uses different keys or key versions.

The key version is set with the ChangeKey command and can be retrieved using the GetVersion command.

8.3 Native Command Format

NT4H2421Tx always communicates in ISO/IEC 7816-4 wrapped mode as described in Section 8.4. Nevertheless it is important to understand the basic format of native commands which consist of the following parts.

A command as sent by the PCD consists of the concatenation of:

- the command code (Cmd)
- zero, one or more header fields (CmdHeader)
- zero, one or more data fields (CmdData)

The response as sent by the PICC consists of the concatenation of:

- the return code (RC)
- zero, one or more data fields (RespData)

NT4H2421Tx supports the APDU message structure according to ISO/IEC 7816-4 [4] for:

- wrapping of the native command format into a proprietary ISO/IEC 7816-4 APDU
- a subset of the standard ISO/IEC 7816-4 commands (ISOSelectFile, ISOReadBinary, ISOUpdateBinary)

Remark: Communication via native ISO/IEC7816-4 commands without wrapping is not supported.

On the native command interface, plain command parameters consisting of multiple bytes are represented least significant byte (LSB) first, similar as for ISO/IEC 14443 parameters during the activation, see [2]. For cryptographical parameters and keys (including the random numbers exchanged during authentication, the TI and the computed MACs), this does not hold. For these, the representation on the interface maps one-to-one to the most significant byte (MSB) first notation used in this specification.

Note that within this document, the 'Xh' prefix indicates hexadecimal integer notation, i.e. not reflecting the byte order representation on the command interface at all.

8.4 ISO/IEC7816-4 Communication frame

NT4H2421Tx uses ISO/IEC 7816-4 [4] type APDUs for command-response pair for both, wrapping of native commands as outlined in Section 8.3 and standard ISO/IEC 7816-4 commands.

Note that for all parameters of standard ISO/IEC 7816-4 commands, the representation on the interface is most significant byte (MSB) first notation. As data like the 2-byte ISO/IEC 7816-4 file identifiers, are in different order for the wrapped native commands, this needs to be taken into account.

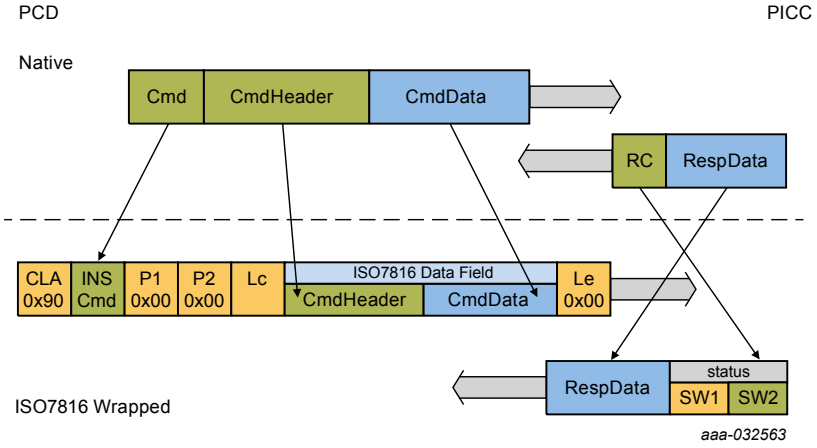


Figure 4. ISO/IEC 7816-4 command response pair

Table 16. ISO/IEC 7816-4 command fields

Field	Description	Length
Command header	Class byte (CLA)	1
	Instruction (INS)	1
	Parameters (P1,P2)	2
Lc field	Length of command data field (Lc), absent if no data field is present	1
Command data field	Absent if no data is sent in the command	Lc
Le field	Expected response length. If Le is 00h, then all available data is sent back for ISO/IEC 7816-4 standard commands. For wrapped commands, Le must always be set to 00h.	1

Note that the maximum number of bytes in the command data field, indicated by Lc, cannot exceed 255 bytes since only short length (1 byte) is supported in NT4H2421Tx, see [4]. This includes overhead for secure messaging.

The maximum number of bytes in the response data field, indicated by Le, cannot exceed 256 bytes since only short length (1 byte) is supported in NT4H2421Tx, see [4]. This includes overhead for secure messaging.

Table 17. ISO/IEC 7816-4 response fields

Field	Description	Length
Response data field	Response data if any, absent if no response data	up to Le
Response trailer	status byte (SW1SW2)	2

The field length and presence might vary for different commands, refer to the specific command description in Section 11.

8.5 Command Chaining

NT4H2421Tx supports standard ISO/IEC 14443-4 [3] command chaining in the following cases:

- the PICC supports ISO/IEC 14443-4 chaining to allow larger command or response frames than the supported buffer size for variants of the following commands:
  - Native commands wrapped into ISO/IEC 7816-4 APDU: ReadData, WriteData, see Section 11.
  - Standard ISO/IEC 7816-4 commands: ISOReadBinary, ISOUpdateBinary i.e. every command where larger frame size can occur.
- the PICC will automatically split a response in several frames to fit with the FSD frame size supported by the PCD and communicated in the RATS.

When a PCD applies ISO/IEC 14443-4 chaining, see [3], it needs to assure the reassembled INF field containing the command header (i.e. ISO/IEC 7816-4 header bytes and/or (Cmd || CmdHeader)) fits within the PICC's buffer (FSC) communicated in the ATS. If not, the PICC may respond with LENGTH_ERROR.

The ISO/IEC 14443-4 chaining does not influence the secure messaging. This means that the secure messaging mechanisms are applied as if the command or response would have been sent in a single large frame. With regards to command execution, commands are handled as if they were received in one large frame, except for write commands where the total frame size can be larger than the supported FSC (WriteData and ISOUpdateBinary). In this case, command execution is started before the complete command is received.

Note that this is important for StandardData files, where the file may end up in an undefined state if the command is not completed successfully, see Section 8.6.

8.6 Backup management

NT4H2421Tx supports hardware based anti-tearing on single frame write operations. This means that blocks up to 128 bytes are either completely written of not changed at all.

8.7 Product originality

NT4H2421Tx supports two types of originality check: one based on an LRP authentication with AES originality keys and another one with a static signature verification with an ECC public key. For detail of the ECC originality check, refer to originality check commands.

The AES-based originality verification using AuthenticateLRPFirst or AuthenticateLRPNonFirst is done with one of the OriginalityKey described in Section 8.2.4.

9 Secure Messaging

Prior to data transmission a mutual three-pass authentication can be done between PICC and PCD which results in the generation of session keys used in the secure messaging.

There are three secure messaging types available in the NT4H2421Tx including Secure Dynamic Messaging. One is using AES-128 and is referred to as AES mode in this data sheet. The other one is using AES-128 with a Leakage Resilient Primitive (LRP) wrapper, also referred to as LRP mode. The LRP mode can be permanently enabled using the SetConfiguration command. After this switch, it is not possible to revert back to AES mode.

Compared to AES mode, the LRP mode has the advantage that it provides strong resistance against side-channel and fault attacks. It serves as a replacement for AES as it only uses standard cryptographic constructions based on AES without any proprietary cryptography. Thus, LRP can be seen as an alternative for AES which is itself based on AES, and is provably as secure as AES, but comes with better properties w.r.t. implementation security, see also [10]. The PCD requires the same LRP mode implementation.

To improve the resistance against side-channel attacks and especially card only attacks for the AES mode, NT4H2421Tx provides a limit for unsuccessful authentication attempts. Every unsuccessful authentication is counted in the TotFailCtr. The parameters TotFailCtrLimit TotFailCtrDecr can be configured as described in Section 11.5.1 using the "Failed authentication counter configuration".

Each unsuccessful authentication is counted internally in the total failed authentication counter TotFailCtr. After reaching the TotFailCtrLimit, see Section 11.5.1, the related key cannot be used for authentication anymore.

In addition, after reaching a limit of *consecutive* unsuccessful authentication attempts, the NT4H2421Tx starts to slow down the authentication processing in order to hamper attacks. This is done by rejecting any authentication command with a AUTHENTICATION_DELAY response. The response time of a single AUTHENTICATION_DELAY response is depending on the FWT, see Section 8.1.1, and is about 65% of the maximum response time specified by FWT. The error response is sent until the total authentication delay time is reached which is equal to the sum of the frame delay times. The total authentication delay time increases with each unsuccessful authentication attempt up to a maximum value, only a successful authentication restores the full operational speed.

Changing a blocked AES key by authenticating with the AppMasterKey and using the ChangeKey command makes the referenced key accessible again. If the AppMasterKey itself is blocked, no recovery is possible.

Each successful AES authentication decrements the TotFailCtr by a value of TotFailCtrDecr.

The AES and LRP authentications are initiated by commands sharing the same command code (First Authentication and Non-First Authentication variants). These authentication protocols are both AES-based, but differ with regards to the actual protocol applied and the subsequent secure messaging mode they initiate. An overview of the different modes is given in Figure 5.

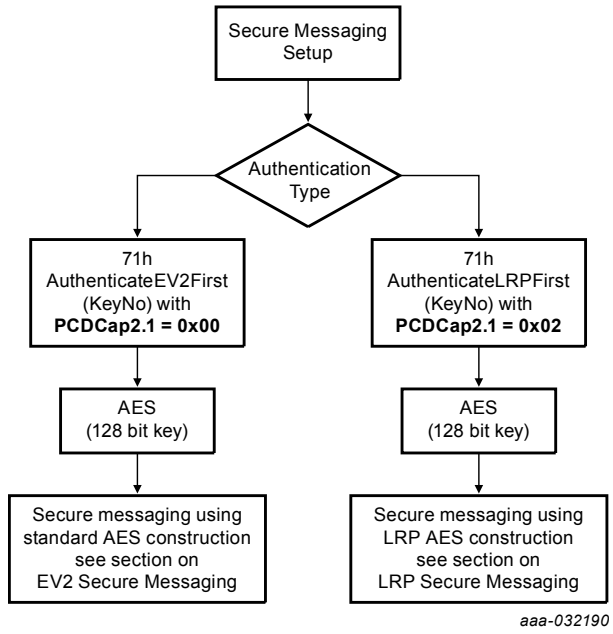


Figure 5. NTAG 424 DNA TT secure messaging setup

A First Authentication can be executed at any time whether the PICC is in authenticated or not authenticated state.

The Non-First Authentication can only be executed while the card is in authenticated state after a successful First or Non-First Authentication.

Correct application of First Authentication and Non-First Authentication allows to cryptographically bind all messages within a transaction together by the transaction identifier established in a First Authentication, see Section 9.1.1, and a command counter, see Section 9.1.2, even if multiple authentications are required.

The following table specifies when to authenticate using First Authentication and when to use Non-First Authentication.

Table 18. When to use which authentication command

Purpose	First Authentication	Non-First Authentication
First authentication (i.e. when not in any authenticated state) with any key different from OriginalityKey	Allowed	Not Allowed
Subsequent authentication (i.e. when in any authenticated state) with any key different from OriginalityKey	Allowed, recommended not to use.	Allowed, recommended to use.
Any LRP authentication with OriginalityKey	Allowed	Allowed

The AuthenticateEV2First initiates a standard AES authentication and secure messaging, see Section 9.1. The other variant AuthenticateLRPFirst initiates an AES authentication and secure messaging based on the Leakage Resilient Primitive (LRP), see Section 9.2.

The negotiation between those two variants is done using the capabilities of the First Authentication and the return message of the first part, where a PCD can distinguish between standard AES authentication and LRP authentication based on the message length.

On First Authentication, the PCD can choose between AES and LRP secure messaging by setting bit 1 of PCDCap2.1 in the issued command. The PD is configured for either AES or LRP secure messaging by the setting of bit 1 from PDCap2.1. This setting is defined with SetConfiguration, see Section 11.5.1.

If the PCD chooses for AES secure messaging, it sends PCDCap2.1 equaling 00h (or no PCDCap2 at all). A NT4H2421Tx will accept the authentication if its PDCap2.1 bit 1 is not set, i.e. the NT4H2421Tx is configured for AES secure messaging. The command is interpreted as AuthenticateEV2First, see Section 9.1.5 for detailed specification. If PDCap2.1 bit 1 is set, i.e. the NT4H2421Tx is configured for LRP secure messaging, the authentication request is rejected.

If the PCD chooses for LRP secure messaging, it sends PCDCap2.1 equaling 02h. NTAG 424 DNA TT will accept the authentication if its PDCap2.1 bit 1 is set, i.e. the NT4H2421Tx is configured for LRP secure messaging. The command is interpreted as AuthenticateLRPFirst and replied with 18 bytes, i.e. ADDITIONAL_FRAME, followed by an additional AuthMode indicating LRP secure messaging, and 16 bytes of data, see Section 11.4.3 for detailed specification. If PDCap2.1 bit 1 is not set, i.e. the NT4H2421Tx is configured for AES secure messaging, the authentication request is also accepted, but responded with 17 bytes, i.e. the AuthenticateEV2First response composed of ADDITIONAL_FRAME, followed by 16 bytes of data, allowing the PCD to fall back to standard AES authentication as well.

With Non-First Authentication, the PCD cannot choose between standard AES and LRP. If authenticated using AES mode, AuthenticateEV2NonFirst will be applied, see Section 9.1.6. If authenticated with LRP mode, AuthenticateLRPNonFirst will be applied, see Section 11.4.4. If not authenticated at all, e.g. if targeting one of the originality keys, only AuthenticateLRPNonFirst is supported.

Below table provides possible negotiation outcomes on FirstAuthentication.

Table 19. Secure messaging mode negotiation

PCD		PD				
Mode	**PCDCap2.1**	**PDCap2.1 (Mode)**	**RC**	**resp PDCap2.1**	**resp PCDCap2.1**	**comment**
Requesting EV2 Secure Messaging	00h	00h (AES)	ADDITIONAL_FRAME: 17-byte response without AuthMode	00h	00h	Matching, AES SM accepted and selected
		02h (LRP)	PERMISSION_DENIED	N/A	N/A	No match, AES SM rejected
Requesting LRP Secure Messaging	02h	00h (AES)	ADDITIONAL_FRAME	00h	02h	No match, PD replies with AES response, allowing a PCD to fall back.
		02h (LRP)	ADDITIONAL_FRAME: 18-byte response with AuthMode set to 01h	02h	02h	Matching, LRP SM accepted and selected

9.1 AES Secure Messaging

The AES Secure Messaging is managed by AuthenticateEV2First and AuthenticateEV2NonFirst.

Note that AuthenticateEV2First and AuthenticateEV2NonFirst can also be used to start LRP Secure Messaging, as defined in Section 9.2. This is done with the PCDCap2 sent in First Authentication and the return code, see Section 9 and Section 9.2.5 for details.

9.1.1 Transaction Identifier

In order to avoid interleaving of transactions from multiple PCDs toward one PICC, the Transaction Identifier (TI) is included in each MAC that is calculated over commands or responses. The TI is generated by the PICC and communicated to the PCD with a successful execution of an AuthenticateEV2First command, see Section 11.4.1. The size is 4 bytes and these 4 bytes can hold any value. The TI is treated as a byte array, so there is no notion of MSB and LSB.

9.1.2 Command Counter

A command counter is included in the MAC calculation for commands and responses in order to prevent e.g. replay attacks. It is also used to construct the Initialization Vector (IV) for encryption and decryption.

Each command, besides few exceptions, see below, is counted by the command counter CmdCtr which is a 16-bit unsigned integer. Both sides count commands, so the actual value of the CmdCtr is never transmitted. The CmdCtr is reset to 0000h at PCD and PICC after a successful AuthenticateEV2First authentication and it is maintained as long as the PICC remains authenticated. In cryptographic calculations, the CmdCtr is represented LSB first. Subsequent authentications using AuthenticateEV2NonFirst do not affect the CmdCtr. Subsequent authentications using the AuthenticateEV2First will reset the CmdCtr to 0000h. The CmdCtr is increased between the command and response, for all communication modes.

When a MAC on a command is calculated at PCD side that includes the CmdCtr, it uses the current CmdCtr. The CmdCtr is afterwards incremented by 1. At PICC side, a MAC appended at received commands is checked using the current value of CmdCtr. If the MAC matches, CmdCtr is incremented by 1 after successful reception of the command, and before sending a response.

For CommMode.Full, the same holds for both the MAC and encryption IV calculation, i.e. the non-increased value is used for the command calculations while the increased value is used for the response calculations.

If the CmdCtr holds the value FFFFh and a command maintaining the active authentication arrives at the PICC, this leads to an error response and the command is handled like the MAC was wrong.

Command chaining, see Section 8.5, does not affect the counter. The chained command is considered as a single command, just as for the other aspects of secure messaging, and thus the related counter is increased only once.

9.1.3 MAC Calculation

MACs are calculated using the underlying block cipher according to the CMAC standard described in [6]. Padding is applied according to the standard.

The MAC used in NT4H2421Tx is truncated by using only the 8 even-numbered bytes out of the 16 bytes output as described [6] when represented in most-to-least-significant order.

Initialization Vector for MACing

The initialization vector used for the CMAC computation is the zero byte IV as prescribed [6].

9.1.4 Encryption

Encryption and decryption are calculated using AES-128 according to the CBC mode of NIST SP800-38A [5].

Padding is applied according to Padding Method 2 of ISO/IEC 9797-1 [7], i.e. by adding always 80h followed, if required, by zero bytes until a string with a length of a multiple of 16 byte is obtained. Note that if the plain data is a multiple of 16 bytes already, an additional padding block is added. The only exception is during the authentication itself (AuthenticateEV2First and AuthenticateEV2NonFirst), where no padding is applied at all.

The notation *E(key, message)* is used to denote the encryption and *D(key, message)* for decryption.

Initialization Vector for Encryption

When encryption is applied to the data sent between the PCD and the PICC, the Initialization Vector (IV) is constructed by encrypting with SesAuthENCKey according to the ECB mode of NIST SP800-38A [5] the concatenation of:

- a 2-byte label, distinguishing the purpose of the IV: A55Ah for commands and 5AA5h for responses
- Transaction Identifier TI
- Command Counter CmdCtr (LSB first)
- Padding of zeros acc. to NIST SP800-38B [6]

This results in the following IVs:

IV for CmdData = *E(SesAuthENCKey; A5h* || *5Ah* || TI || CmdCtr || *0000000000000000h)*

IV for RespData = *E(SesAuthENCKey;5Ah* || *A5h* || TI || CmdCtr || *0000000000000000h)*

When an encryption or decryption is calculated, the CmdCtr to be used in the IV are the current values. Note that this means that if CmdCtr = *n* before the reception of a command, after the validation of the command CmdCtr = *n* + *1* and that value will be used in the IV for the encryption of the response.

For the encryption during authentication (both AuthenticateEV2First and AuthenticateEV2NonFirst), the IV will be 128 bits of 0.

9.1.5 AuthenticateEV2First Command

This section defines the Authentication, which is mandatory to be used first in a transaction when using Secure Messaging, see Table 18. In this procedure both, the PICC as well as the PCD show in an encrypted way that they possess the same secret, i.e. the same key. This authentication is supported with AES keys.

The authentication consists of two parts: AuthenticateEV2First - Part1 and Section 9.1.6 - Part2. Detailed command definition can be found in Section 11.4.1. The protocol cannot

be interrupted by other commands. On any command different from AuthenticateEV2First - Part2 received after the successful execution of the first part, the PICC aborts the ongoing authentication.

During this authentication phase, the PICC accepts messages from the PCD that are longer than the lengths derived from this specification as long as LenCap is correct. This feature is to support the upgradability to following generations of NT4H2421Tx. The PCD rejects answers from the PICC when they don't have the proper length. Note that if present, PCDcap2:1:Bit1 must not be set, otherwise LRP authentication is targeted, see Section 9.2.5.

Upon reception of AuthenticateEV2First, the PICC validates the targeted key. If the key does not exist, AuthenticateEV2First is rejected.

The PICC generates a random 16-byte challenge *RndB* and send this encrypted to the PCD, according to Section 9.1.4. Additionally, the PICC resets CmdCtr to zero and generate a random Transaction Identifier (TI).

Upon reception of the AuthenticateEV2First response from the PICC, the PCD also generates a random 16-byte challenge *RndA*. The PCD encrypts, on his turn, the concatenation of *RndA* with *RndB'*, which is the received challenge after decryption and rotating it left by one byte. Within AuthenticateEV2First - Part2, this is sent to the PICC. Upon reception of AuthenticateEV2First - Part2, the PICC decrypts the second message and validates the received *RndB'*. If not as expected, the command is rejected. Else it generates *RndA'* by rotating left the received *RndA* by one byte. This is returned together with the generated TI. Also, the PICC sends 12 bytes of capabilities to the PCD: 6 bytes of PICC capabilities PDcap2 and 6 bytes of PCD capabilities PCDcap2 that were received on the command (sent back for verification).

The use of those capabilities, and the negotiation process is described in Section 9. Note that part of PDCap will be configurable with SetConfiguration. PCDcap2 is used to refer both to the value sent from the PCD to the PICC and to the value used in the encrypted response message from the PICC to the PCD where in this case the PCDcap2 is the adjusted version of the originally sent PCDcap2: i.e. truncated or padded with zero bytes to a length of 6 bytes if needed.

On successful execution of the authentication protocol, the session keys SesAuthMACKey and SesAuthENCKey are generated according to Section 9.1.7. The PICC is in EV2 authenticated state and the Secure Messaging is activated. On any failure during the protocol or if one of the OriginalityKey were targeted, the PICC ends up in not authenticated state.

If there is a mismatch between the capabilities expected by the PCD and the capabilities presented by the PICC to the PCD (both the PDcap2 and the echoed/adjusted PCDcap2), it is the responsibility of the PCD to take the proper actions based on the application the PCD is running. This decision is outside the scope of this specification.

9.1.6 AuthenticateEV2NonFirst Command

This section defines the Non-First Authentication, which is recommended to be used if Secure Messaging is already active, see Table 18. In this procedure both, the PICC as well as the PCD show in an encrypted way that they possess the same secret, i.e. the same key. This authentication is supported with AES keys.

The authentication consists of two parts: AuthenticateEV2NonFirst - Part1 and AuthenticateEV2NonFirst - Part2. Detailed command definition can be found in Section 11.4.2. This command is rejected if there is no active authentication, except if the

targeted key is the OriginalityKey. For the rest, the behavior is exactly the same as for AuthenticateEV2First, except for the following differences:

- No *PCDcap2* and *PDcap2* are exchanged and validated.
- Transaction Identifier TI is not reset and not exchanged.
- Command Counter CmdCtr is not reset.

After successful authentication, the PICC remains in EV2 authenticated state. On any failure during the protocol, the PICC ends up in not authenticated state.

9.1.7 Session Key Generation

At the end of a valid authentication with AuthenticateEV2First or AuthenticateEV2NonFirst, both the PICC and the PCD generate two session keys for secure messaging, as shown in Figure 6:

- SesAuthMACKey for MACing of messages
- SesAuthENCKey for encryption and decryption of messages

Note that these identifiers are also used in context of the LRP protocol, though the actual calculation of the session keys is different, see Section 9.2.7.

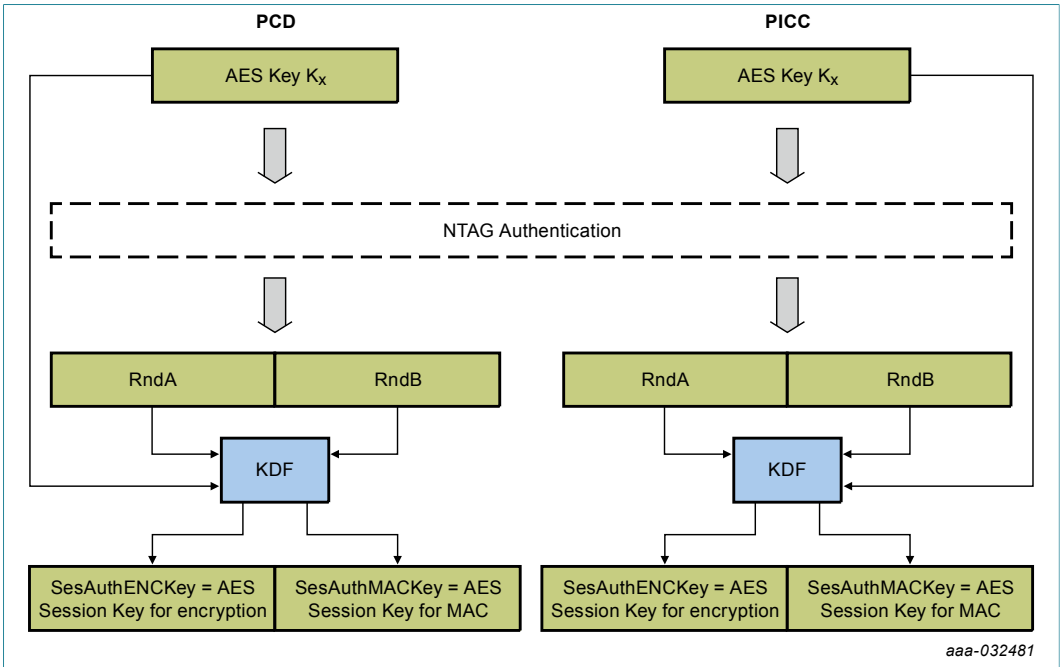


Figure 6. Session key generation for Secure Messaging

The session key generation is according to NIST SP 800-108 [8] in counter mode.

The Pseudo Random Function PRF(key; message) applied during the key generation is the CMAC algorithm described in NIST Special Publication 800-38B [6]. The key derivation key is the key Kx that was applied during authentication. As the authentications are restricted to target AES keys, the generated session keys are also of AES.

The input data is constructed using the following fields as defined by [8]. Note that NIST SP 800-108 allows defining a different order than proposed by the standard as long as it is unambiguously defined.

- a 2-byte label, distinguishing the purpose of the key: 5AA5h for MACing and A55Ah for encryption
- a 2-byte counter, fixed to 0001h as only 128-bit keys are generated.
- a 2-byte length, fixed to 0080h as only 128-bit keys are generated.
- a 26-byte context, constructed using the two random numbers exchanged, RndA and RndB

First, the 32-byte input session vectors SV_x are derived as follows:

SV1 = A5h||5Ah||00h||01h||00h||80h||RndA[15..14]||
(RndA[13..8] # RndB[15..10])||RndB[9..0]||RndA[7..0]
SV2 = 5Ah||A5h||00h||01h||00h||80h||RndA[15..14]||
(RndA[13..8] # RndB[15..10])||RndB[9..0]||RndA[7..0]

with # being the XOR-operator.

Then, the 16-byte session keys are constructed as follows:

SesAuthENCKey = PRF(K_x, SV1)
SesAuthMACKey = PRF(K_x, SV2)

9.1.8 Plain Communication Mode

The command and response data is not secured. The data is sent in plain, see Figure 7, i.e. as defined in the command specification tables, see Section 11.

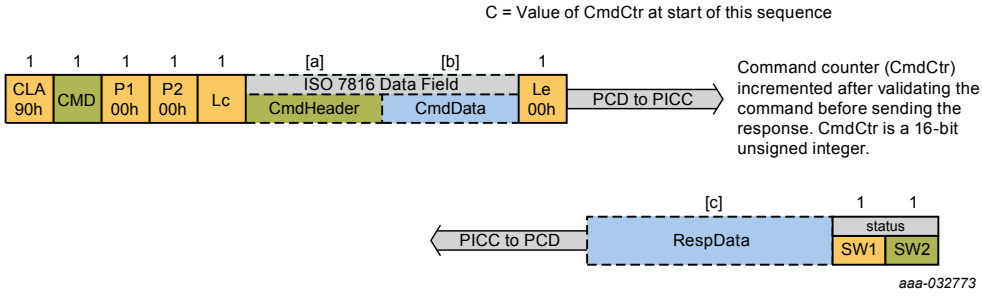


Figure 7. Plain Communication Mode

However, note that, as the PICC is in authenticated state (EV2 authenticated state or LRP authenticated state), the command counter CmdCtr is still increased as defined in Section 9.1.2.

This allows the PCD and PICC to detect any insertion and/or deletion of commands sent in CommMode.Plain on any subsequent command that is sent in CommMode.MAC (e.g. CommitTransaction) or CommMode.Full.

9.1.9 MAC Communication Mode

The Secure Messaging applies MAC to all commands listed as such in Section 11.2.

In case MAC is to be applied, the following holds. The MAC is calculated using the current session key SesAuthMACKey. MAC calculation is done as defined in Section 9.1.3.

For commands, the MAC is calculated over the following data (according to the definitions from Section 8.3) in this order:

- Command, Cmd

- Command Counter CmdCtr
- Transaction Identifier TI
- Command header - CmdHeader (if present)
- Command data - CmdData (if present)

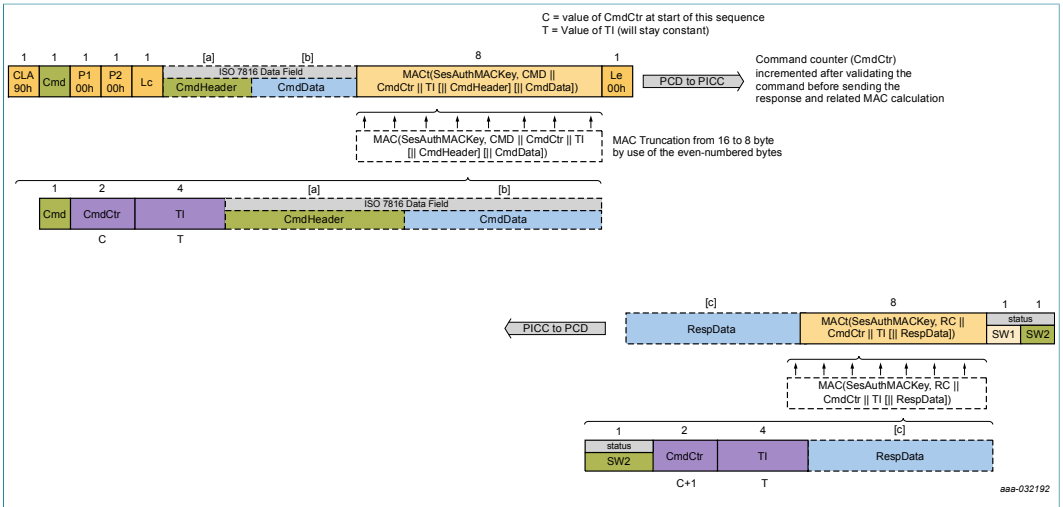
For responses, the MAC is calculated over the following data in this order:

- Return code - RC
- Command Counter - CmdCtr (The already increased value)
- Transaction Identifier - TI
- Response data - RespData (if present)

CmdCtr is the Command Counter as defined in Section 9.1.2. Note that the CmdCtr is increased between the computation of the MAC on the command and the MAC on the response. TI is the Transaction Identifier, as defined in Section 9.1.1. The other input parameters are as defined in Section 8.3. The calculation is illustrated in Figure 8.

In case of command chaining, the MAC calculation is not interrupted. The MAC is calculated over the data including the complete data field (i.e. either CmdData or RespData of all frames) at once. The MAC is always transmitted by appending to the unpadded plain command. If necessary, an additional frame is sent. If a MAC over the command is received, the PICC verifies the MAC and rejects commands that do not contain a valid MAC by returning INTEGRITY_ERROR.

In this case, the ongoing command and transaction are aborted (see also Section 11). The authentication state is immediately lost and the error return code is sent without a MAC appended. Note that any other error during the command execution has the same consequences.



CLA, P1, P2, LC, Le and SW1 not included in secure messaging calculation. SW2 is the return code (RC) and appended in the beginning for secure messaging calculation

Figure 8. Secure Messaging: MAC Communication mode

9.1.10 Full Communication Mode

The Secure Messaging applies encryption (CommMode.Full) to all commands listed as such in Section 11.2. In case CommMode.Full is to be applied, the following holds. The encryption/decryption is calculated using the current session key SesAuthENCKey. Calculation is done as defined in Section 9.1.4 over either the command or the response

data field (i.e. CmdData or RespData). Note that none of the commands have a data field in both the command and the response frame.

After the encryption, the command and response frames are handled as with MAC. This means that additionally a MAC is calculated and appended for transmission using the current session key SesAuthMACKey. This is exactly done as specified for MAC in Section 9.1.9, replacing the plain CmdData or RespData by the encrypted field: *E(SesAuthENCKey; CmdData)* or *E(SesAuthENCKey; RespData)*. The complete calculation is illustrated in Figure 9. In case of command chaining, the encryption/ decryption is applied over the complete data field (i.e. of all frames). If necessary, due to the padding or the MAC added, an additional frame is sent. If encryption of the command is required, after the MAC verification as described for MAC, the PICC verifies and removes the padding bytes. Commands without a valid padding are also rejected by returning INTEGRITY_ERROR.

In this case, the ongoing command and transaction are aborted (see also Section 11). The authentication state is immediately lost and the error return code is sent without a MAC appended. Note that any other error during the command execution has the same consequences.

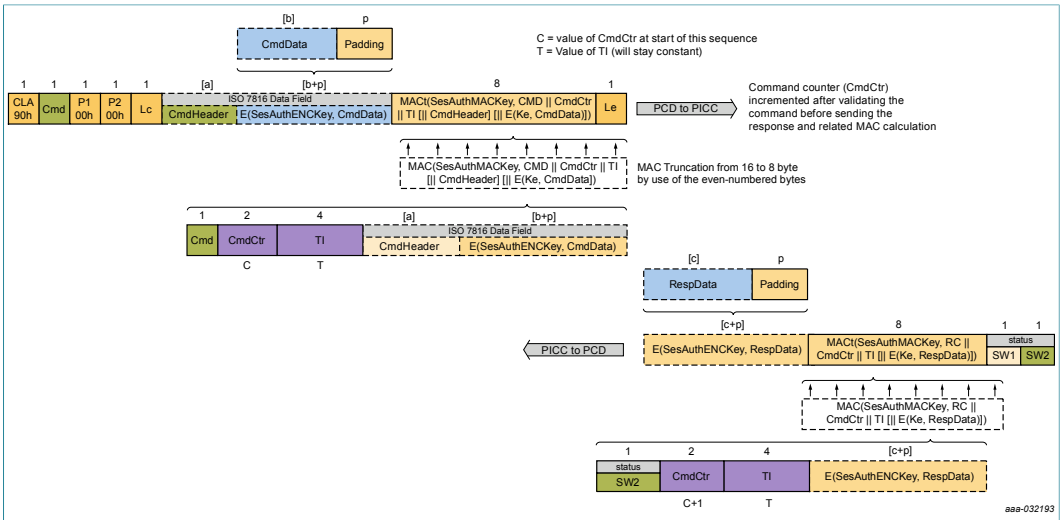


Figure 9. Secure Messaging: CommMode.Full

9.2 LRP Secure Messaging

The LRP Secure Messaging is using AES-128 to construct a Leakage Resilient Primitive. This way, it allows side-channel resistant implementation.

Like the AES secure messaging, this secure messaging mode is managed by commands with the same command code as AuthenticateEV2First and AuthenticateEV2NonFirst. To distinguish and ease the descriptions, they are renamed for the LRP case into AuthenticateLRPFirst and AuthenticateLRPNonFirst. The recommendations of Section 9 on when to use one or the other command also apply for LRP secure messaging.

9.2.1 Transaction identifier

The Transaction Identifier (TI) is treated exactly in the same way by LRP secure messaging as defined for AES secure messaging, see Section 9.1.1.

9.2.2 Command counter

The Command counter (CmdCtr) is treated exactly in the same way by LRP secure messaging as defined for AES secure messaging, see Section 9.1.2.

9.2.3 MAC calculation

MACs are computed by using a CMAC construction on top of the LRP primitive. This is specified in [10]. This document uses the following notation where the right hand refers to the notation of [10].

$MAC_{LRP}(key,message) = CMAC\_LRP(4,key, Len(message),message)$

Note that in the LRP context a key is not purely a single value, but rather consists of the associated set of plain texts, an updated key and in context of CMAC also the subkeys K1 and K2. Therefore K1 and K2 are not shown (contrary to [10]) as they can be calculated inside.

$MACt_{LRP}(key, message)$ denotes the CMAC after truncation to 8 bytes which is identical to the truncation of the AES secure messaging i.e. the even-numbered bytes are retained in most-to-least-significant order, see Section 9.1.3.

The initialization vector used for the CMAC computation is the zero byte IV as prescribed [10].

9.2.4 Encryption

Encryption and decryption are calculated using a Leakage Resilient Indexed CodeBook (LRICB) construction on top of the LRP primitive: *LRICB*of [10].

For this purpose an Encryption Counter is maintained: EncCtr is a 32-bit unsigned integer as Input Vector (IV) for encryption/decryption. The EncCtr is reset to 000000000h at PCD and PICC when starting an authentication with AuthenticateLRPFirst or AuthenticateLRPNonFirst targeting LRP. The counter is incremented during each encryption/decryption of each 16-byte block. i.e. for 64-byte encryption/decryption the EncCtr is increased by 5 due to 4 blocks of 16-byte of data plus one block of padding. Note that for AuthenticateLRPFirst the value 00000000h is already used for the response of part 2, so the actual secure messaging starts from 00000001h. For AuthenticateLRPNonFirst, secure messaging starts from 00000000h as the counter is not used during the authentication. EncCtr is further maintained as long as the PICC remains in LRP authenticated state. Note that for the key stream calculation [10], the counter is represented MSB first.

Padding is applied according to Padding Method 2 of ISO/IEC 9797-1 [7], i.e. by adding always 80h followed, if required, by zero bytes until a string with a length of a multiple of 16 bytes is obtained. Note that if the plain data is a multiple of 16 bytes already, an additional padding block is added. The only exception is during the authentication itself (AuthenticateLRPFirst and AuthenticateLRPNonFirst), where no padding is applied at all.

The notation $E_{LRP}(key, plaintext)$ is used to denote the encryption, i.e. *LRICBEnc* of [10] and $D_{LRP}(key, ciphertext)$ for the complementary decryption operation. Note that in the LRP context a key is not purely a single value, but rather consists of the associated set of plain texts and updated key. Also, as specified in [10], the EncCtr is updated as part of the operation.

Note that the EncCtr cannot overflow. Due to the supported file sizes, the CmdCtr will always expire before.

Note that the MSB representation of EncCtr is different from other counter representations in this specification, but allows saving some AES calculations in the key stream generation.

9.2.5 AuthenticateLRPFirst command

The AuthenticateLRPFirst command reuses the same command code as AuthenticateEV2First. The distinction is made via the PCDCap2.1 parameter, as explained in Section 9.

The AuthenticateLRPFirst command is fully compliant with the mutual three-pass authentication of ISO/IEC 9798-4 [7].

The authentication consists of two parts: AuthenticateLRPFirst - Part1 and AuthenticateLRPFirst Part2. Detailed command definition can be found in Section 11.4.3.

The protocol cannot be interrupted by other commands. On any command different from AuthenticateLRPFirst - Part2 received after the successful execution of the first part, the PICC aborts the ongoing authentication.

During this authentication phase, the PICC accepts messages from the PCD that are longer than the lengths derived from this specification as long as *LenCap* is correct. This feature is to support the upgradability to following generations of NTAG 424 DNA TT.

Apart from bit 1 of PCDCap2.1, which need to be set to 1 for AuthenticateLRPFirst resulting into 020000000000h, the content of PCDCap2 is not interpreted by the PICC. The PCD rejects answers from the PICC when they don't have the proper length.

Upon reception of AuthenticateLRPFirst, the PICC validates the targeted key. If the key does not exist, AuthenticateLRPFirst is rejected. At PICC level, the only available key is the OriginalityKey.

The PICC generates a random 16-byte challenge *RndB* and send this in plain to the PCD. Additionally, the PICC and PCD reset both CmdCtr and EncCtr to zero and generate a random TI.

Upon reception of the AuthenticateLRPFirst response from the PICC, the PCD also generates a random 16-byte challenge *RndA*. Now the PCD calculates the session keys SesAuthMACKey and SesAuthENCKey, as specified in Section 9.2.7. As explained there for LRP, a session key consists of a set of plain texts and an updated key.

Then the PCDResponse computes a MAC over the concatenation of *RndA* with *RndB*, applying the SesAuthMACKey with the algorithm defined in Section 9.2.3. Note that MACs are not truncated during the authentication. Within AuthenticateLRPFirst - Part2, the concatenation of *RndA* and this MAC is sent to the PICC.

Upon reception of AuthenticateLRPFirst - Part2, the PICC validates the received MAC. If not as expected, the command is rejected. Else it encrypts the generated TI concatenated with 12 bytes of capabilities to the PCD: 6 bytes of PICC capabilities *PDCap2* and 6 bytes of PCD capabilities *PCDCap2* that were received on the command (sent back for verification). Encryption is done according to Section 9.2.4, applying SesAuthENCKey.

Note that part of PDCap is configurable with SetConfiguration. *PCDCap2* is used to refer both to the value sent from the PCD to the PICC and to the value used in the encrypted response message from the PICC to the PCD where in this case the *PCDCap2* is the adjusted version of the originally sent *PCDCap2*: i.e. truncated or padded with zero bytes to a length of 6 bytes if needed. After that encryption, the PICCResponse will also compute a MAC over the concatenation of *RndB*, *RndA* and the encrypted data.

9.2.6 AuthenticateLRPNonFirst command

This section defines the LRP Non-First Authentication, which is recommended to be used if LRP Secure Messaging is already active, see Table 18.

The authentication consists of two parts: AuthenticateLRPNonFirst - Part1 and AuthenticateLRPNonFirst Part2. Detailed command definition can be found in Section 11.4.4.

This command is rejected if there is no active LRP authentication, except if the targeted key is the OriginalityKey.

For the rest, the behavior is exactly the same as for AuthenticateLRPFirst, except for the following differences:

- PCDCap2 and PDCap2 are not exchanged and validated
- TI is not reset and not exchanged
- CmdCtr is not reset

Note that EncCtr is reset to zero also on AuthenticateLRPNonFirst.

After successful authentication, the PICC remains in LRP authenticated state, except if the OriginalityKey was targeted. In that case, the PICC is in not authenticated state. On any failure during the protocol, the PICC ends up in not authenticated state.

9.2.7 Session key generation

Next to the algorithms for MAC calculation and encryption, one of the major differences between the LRP secure messaging and the AES secure messaging is that the session keys are generated and already applied during the authentication with AuthenticateLRPFirst or AuthenticateLRPNonFirst.

Also for the LRP protocol, two keys are generated:

- SesAuthMACKey for MACing of messages
- SesAuthENCKey for encryption and decryption of messages

During the authentication, the SesAuthMACKey is used for both AuthenticateLRPFirst and AuthenticateLRPNonFirst. SesAuthENCKey is only used for AuthenticateLRPFirst.

Being LRP keys, this section shows how both the plain texts and the updated key [10] related to these session keys are computed. In the remainder of the document, when the session key is applied in the LRP context the combination of those plain texts and updated key is meant.

The session key generation is according to NIST SP 800-108 [8] in counter mode.

The Pseudo Random Function *PRF(key; message)* applied during the key generation is the CMAC algorithm on top of the LRP primitive. This is specified in [10], see also Section 9.2.3. The key derivation key is the key *Kx* that was applied during authentication. Note that from this key a set of plaintexts and updated key is computed, so the static key is only used in this derivation. The generated session keys are AES keys. The input data is constructed using the following fields as defined by [8]. Note that NIST SP 800-108 allows defining a different order than proposed by the standard as long as it is unambiguously defined.

- a 2-byte counter, fixed to 0001h as only 128-bit keys are generated

- a 2-byte length, fixed to 0080h as only 128-bit keys are generated
- a 26-byte context, constructed using the two random numbers exchanged, *RndA* and *RndB*
- a 2-byte label: 9669h

Firstly, the 32-byte input session vector *SV* is derived as follows:

SV = 00h || 01h || 00h || 80h || RndA[15::14] ||
(RndA[13::8] # RndB[15::10]) || RndB[9::0] || RndA[7::0] || 96h || 69h

with # being the XOR-operator.

Then, the session key material is constructed as follows:

AuthSPT = generatePlaintexts(4; K_x)
{AuthUpdateKey} = generateUpdatedKeys(1; K_x)
SesAuthMasterKey = MACLRP (K_x; SV)
SesAuthSPT = generatePlaintexts(4; SesAuthMasterKey)
{SesAuthMACUpdateKey; SesAuthENCUpdateKey} = generateUpdatedKeys(2; SesAuthMasterKey)

with *generatePlaintexts* and *generateUpdatedKeys* the functions from [10]. Note that the output of *generateUpdatedKeys* is shown in the order that the keys are generated. The actual SesAuthMACKey then consists for LRP of the set of plaintexts SesAuthSPT (consisting of 16 16-byte values) and SesAuthMACUpdateKey. The SesAuthENCKey consists of the same set of plaintexts SesAuthSPT and SesAuthENCUpdateKey.

9.2.8 Plain communication mode

For CommMode.Plain, command processing in LRP authenticated state is identical to AES secure messaging in EV2 authenticated state, see Section 9.1.8.

9.2.9 MAC communication mode

For MAC, apart from using the LRP MAC algorithm, as specified in Section 9.2.3, the command processing in LRP authenticated state is identical to AES secure messaging in EV2 authenticated state, see Section 9.1.9. The calculation is illustrated in Figure 10.

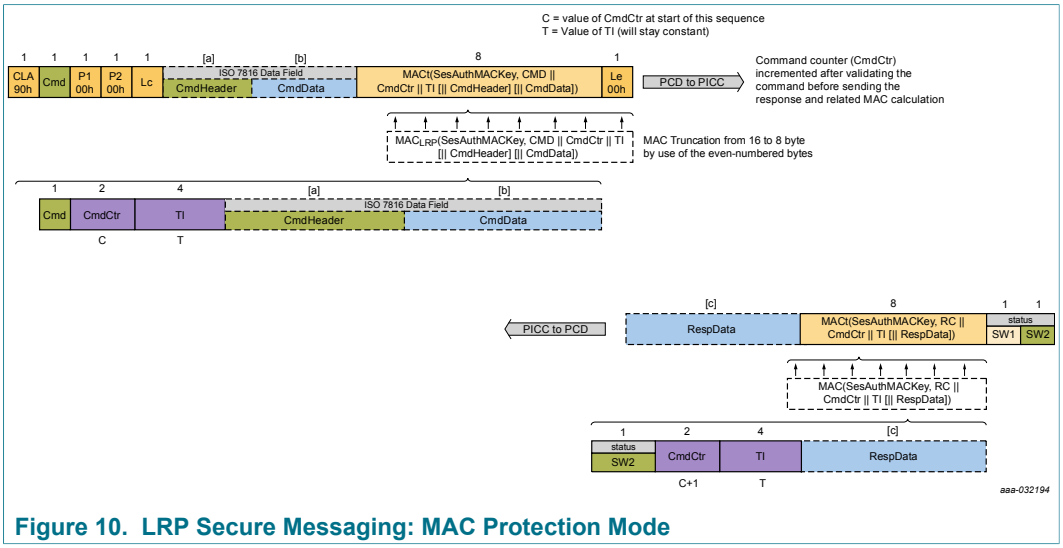


Figure 10. LRP Secure Messaging: MAC Protection Mode

9.2.10 Full communication mode

For CommMode.Full, apart from using the LRP encryption and MAC algorithm, as specified in Section 9.2.4, the command processing in LRP authenticated state is identical to AES secure messaging in EV2 authenticated state, see Section 9.1.10. This is as well illustrated in Figure 11.

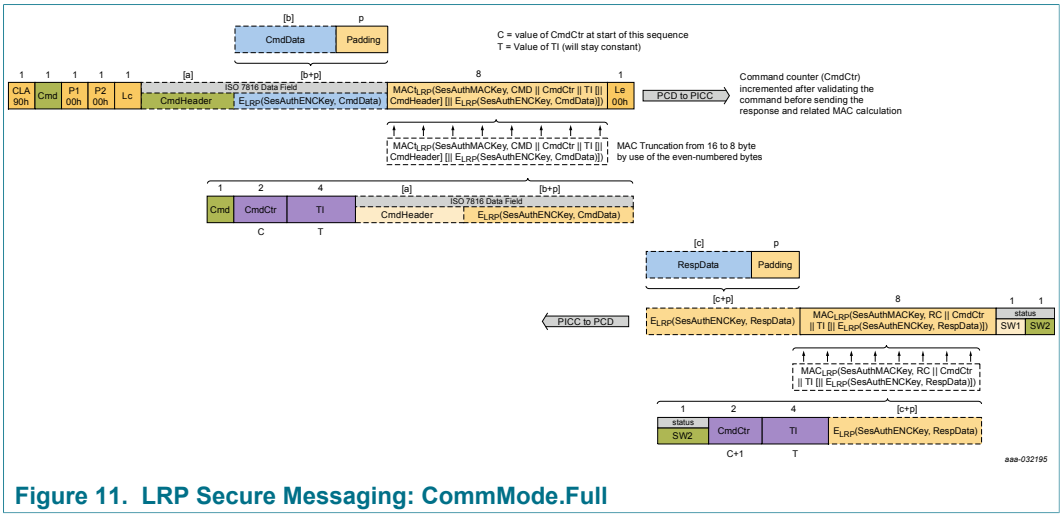


Figure 11. LRP Secure Messaging: CommMode.Full

9.3 Secure Dynamic Messaging

The Secure Dynamic Messaging (SDM) allows for confidential and integrity protected data exchange, without requiring a preceding authentication. NT4H2421Tx supports SDM for reading from one of the StandardData files on the PICC. Secure Dynamic Messaging allows adding security to the data read, while still being able to access it with standard NDEF readers. The typical use case is an NDEF holding a URI and some meta-data, where SDM allows this meta-data to be communicated confidentiality and integrity protected toward a backend server.

When using SDM, residual risks coming with the Secure Dynamic Messaging for Reading have to be taken into account. As SDM allows free reading of the secured message, i.e. without any up-front reader authentication, anybody can read out the message. This means that also a potential attacker is able to read out and store one ore multiple messages, and play them at a later point in time to the verifier.

If this residual risk is not acceptable for the system's use case, the legacy mutual authentication (using challenge response protocol) and subsequent secure messaging should be applied. This would require using an own application and operating outside a standard NDEF read operation.

Other risk mitigation may be applied for SDM to limit the residual risk, without completely removing it:

- Track SDMReadCtr per tag at the verifying side. Reject SDMReadCtr values that have been seen before or that are played out-of-order. This is a minimum requirement any verifier should implement.
- Limit the time window of an attacker by requiring tags to be presented regularly (e.g. at least once a day) in combination with the previous mitigation.

- Read out the SDM-protected file more than once. This does not protect against attackers that have read out the valid tag also multiple times and play the received responses in the same sequence.

9.3.1 SDM Read Counter

In order to allow replay detection by the party validating the data read, a read counter is associated with the file for which Secure Dynamic Messaging is enabled.

SDMReadCtr is a 24-bit unsigned integer. The SDMReadCtr is reset to 000000h when enabling SDM with ChangeFileSettings. In cryptographic calculations and represented with binary encoding on the external interface, the SDMReadCtr is represented LSB first. When represented with ASCII encoding on the contactless interface, it is represented MSB first. Note that this is in line with the NFC counter representation in [13]

In not Authenticated state, the SDMReadCtr is incremented by 1 before calculating the response of the first read command, ReadData or ISOReadBinary, if successful. On subsequent read commands targeting the same file, the SDMReadCtr is not increased, and the current value is used. As soon as a different command has been received, the counter is incremented again on a subsequent read command. Also when varying between ReadData and ISOReadBinary, the counter is incremented on each first instance of the read command type. Note that the SDMReadCtr is not incremented when authenticated.

If the SDMReadCtr reaches the SDMReadCtrLimit (see Section 9.3.2) or the value FFFFFFh (if SDMReadCtrLimit is not enabled) and a first read command arrives at the PICC, an error is being returned. Command chaining, see Section 8.5, does not additionally affect the counter increase. The chained command is considered as a single command.

SDMReadCtr can be retrieved via the mirroring as part of the PICCData, see Section 9.3.3, or it can be retrieved via GetFileCounters.

9.3.2 SDM Read Counter Limit

In order to allow limiting the number of reads that can be done with a single device applying Secure Dynamic Messaging, an optional SDM Read Counter Limit can be configured. There are two main use cases:

- limit the number of usages from the card side. Note that typically this can also be controlled from the backend verifying the SDM for Read protected message.
- limit the number of traces that can be collected on the symmetric crypto processing. This way potential side channel attacks can be mitigated, see also the Failed Authentication Counter feature for the mutual authentication. In this case, it is recommended to have the configured limit aligned with TotFailCtrLimit.

The number of reads that can be executed for an SDM configured file can be limited by setting an SDM Read Counter Limit (SDMReadCtrLimit). This is an unsigned integer of 3 bytes, related with SDMReadCtr. On the interface, the SDMReadCtrLimit is represented LSB first. The SDMReadCtrLimit can be enabled by setting a customized value with ChangeFileSettings. It can be retrieved with GetFileSettings.

Once the SDMReadCtr equals the SDMReadCtrLimit, no reading of the file with ReadData or ISOReadBinary in not authenticated state can be executed. Note that if authenticated, reading is always possible even if SDMReadCtrLimit is reached, applying the regular secure messaging. If the SDMReadCtrLimit is disabled with ChangeFileSettings, this is also equivalent to putting it to the maximum value: FFFFFFh.

9.3.3 PICCData

The PICCData holds metadata of the targeted PICC and file, consisting of the UID and/or the SDMReadCtr. Whether PICCData is transmitted in plain or encrypted depends on the configuration of the SDMMetaRead access rights on the file, see Section 8.2.3.4. If the SDMMetaRead access right is configured for free access (Eh), PICCData is plain and is defined according to Table 20.

ASCII mirroring is reflected by the function EncodeASCII(), which means that each hexadecimal character of the hexadecimal representation will be ASCII encoded using capitals. For example, the UID 04E141124C2880h becomes: 30h 34h 45h 31h 34h 31h 31h 32h 34h 43h 32h 38h 38h 30h.

Table 20. PICCData: plain encoding and lengths

Mode	PICCData Value	Length with 7-byte UID
ASCII	EncodeASCII(UID)	UIDLength = 14 (i.e. 2*UIDLen)
ASCII	EncodeASCII(SDMReadCtr)	SDMReadCtrLength = 6 (i.e. 2*3)

Note that the SDMReadCtr, as defined in Section 9.3.1, is represented MSB first for ASCII case. If the SDMMetaRead access right is configured for an application key, PICCData will be encrypted as defined in Section 9.3.4. In this case, the input plaintext for the encryption is always in binary encoding, while the output ciphertext will be ASCII encoded.

The PICCData is mirrored within the file. This is configured with ChangeFileSettings via the related offsets.

In case of plain mirroring (i.e. access right SDMMetaRead = Eh):

- UIDOffset configures the UID mirroring position. It is only given if UID mirroring is enabled.
- SDMReadCtrOffset configures the SDMReadCtr mirroring position. It is only given if SDMReadCtr mirroring is enabled. Note that it is possible to enable the SDMReadCtr but without mirroring by putting SDMReadCtrOffset to FFFFFFh. In this case it can be retrieved with the GetFileCounters command.

If UID and SDMReadCtr are mirrored within the file, they shall not overlap:

- UIDOffset ≥ SDMReadCtrOffset + SDMReadCtrLength OR SDMReadCtrOffset ≥ UIDOffset + UIDLength.

In case of encrypted mirroring (i.e. SDMMetaRead = 0h..4h), PICCDataOffset configures the PICCData mirroring. The encryption is outlined in Section 9.3.4.

If the PICCData is mirrored within the file, the mirroring shall always be applied in not authenticated state, independently of whether Secure Dynamic Messaging applies. This means it will also be applied if reading the file with free access due to Read or ReadWrite access right. Note that if authenticated, no mirroring is done, i.e. the regular secure messaging is always applied on the static file data.

With NT4H2421Tx, PICCData is always ASCII encoded.

When both the UID and SDMReadCtr are mirrored, “x” (78h) is used as a separator character with NTAG2x [13]. This can be achieved by leaving one byte space between the placeholders defined by UIDOffset and SDMReadCtrOffset, and writing “x” (78h) in the static file data.

9.3.4 Encryption of PICCData

In case of encrypted PICCData mirroring (both binary and ASCII), PICCDataTag specifies what metadata is mirrored, together with the length of the UID if mirrored, as defined in Table 21.

Table 21. PICCDataTag

Bit	Value	Description
Bit7	-	UID mirroring
	0	disabled
	1	enabled
Bit6	-	SDMReadCtr mirroring
	0	disabled
	1	enabled
Bit5-4	00	RFU
Bit3-0	-	UID Length
	0h	RFU (if UID is not mirrored)
	7h	7 byte UID

The format of the plain text is: *PICCDataTag [|| UID] [|| SDMReadCtr].*

To ensure that the encrypted PICCData cannot be abused for tracking purposes, random padding is added to the actual plain text input.

The random padding is generated for the response of the first read command, ReadData or ISOReadBinary. On subsequent read commands targeting the same file the same random padding is reused. This allows for reading the file in chunks, where a chunk border might even be in the middle of the encrypted PICCData. As soon as a different command has been received, fresh random padding is generated on a subsequent read command. Also when varying between ReadData and ISOReadBinary, fresh random padding is generated.

The key applied for encryption of PICCData is the SDMMetaReadKey as defined by the SDMMetaRead access right.

9.3.4.1 AES mode encryption

Encryption and decryption of the PICCData are calculated using the underlying block cipher according to the CBC mode of NIST SP800-38A [5], applying zero byte IV. Therefore PICCData is defined as follows:

PICCData = E(SDMMetaReadKey; PICCDataTag [|| UID] [|| SDMReadCtr] || RandomPadding)

with PICCDataTag as defined in Section 9.3.3, and RandomPadding being a random byte string generated by the PICC to make the input 16 bytes long. Note that because of the ASCII encoding the required placeholder length doubles.

9.3.4.2 LRP mode encryption

Encryption and decryption of the PICCData are calculated using a leakage resilient indexed codebook (LRICB) construction on top of the LRP primitive: LRICB of [13].

For this operation, the LRP key material is constructed as follows:

SDMMetaReadSPT = generatePlaintexts(4; SDMMetaReadKey)

{SDMMetaReadUpdateKey} = generateUpdatedKeys(1; SDMMetaReadKey)

As input counter for the CTR construction, the 8-byte random PICCRand, generated by the PICC, is used. Apart from the counter applied, this is identical to the encryption used for LRP secure messaging, see Section 9.2.4.

Therefore PICCData is defined as follows:

PICCData = PICCRand || E_{LRP} (SDMMetaReadKey; PICCDataTag [|| UID] [|| SDMReadCtr] || RandomPadding)

with PICCDataTag as defined in Section 9.3.3, and PICCRand being an 8-byte long random byte string generated by the PICC. RandomPadding is applied like for AES mode.

The required placeholder length in the NDEF message is 48 bytes due to ASCII encoding.

Note that due to the different sizes of encrypted PICCData with AES mode and LRP mode, the Secure Dynamic Messaging and mirroring will be disabled when switching from AES mode to LRP mode with SetConfiguration.

9.3.5 Tag Tamper Status

When the Tag Tamper Protection feature is enabled, see Section 10, it is possible to mirror the tag tamper status TTStatus consisting of TTPermStatus and TTCurrStatus. They are mirrored concatenated in the following order: TTPermStatus || TTCurrStatus. They can be mirrored in plain or encrypted.

For the latter, TTStatusOffset needs to be positioned within the placeholder for the plain data that serves as input for SDMENCFileData, see Section 9.3.7. In this case, the static file data is replaced by the dynamic statuses before applying the encryption. Note however, that either both status bytes are plain or both are encrypted. As the TTPermStatus and TTCurrStatus are already ASCII encoded, no ASCII encoding is applied on top, and only a 2-byte placeholder is used. Where the status is mirrored within the file, is configured with ChangeFileSettings, see Section 11.7.1 via TTStatusOffset. The restrictions on this offset shall be that it may not be overlapping with any PICCData mirrored or with the SDMMAC.

If the TTPermStatus is mirrored within the file, the mirroring shall always be applied in not authenticated state, independently of whether Secure Dynamic Messaging applies. This means it will also be applied if reading the file with free access due to Read or ReadWrite access right. Note that if authenticated, no mirroring is done, i.e. the regular secure messaging is always applied on the static file data.

9.3.6 SDMENCFileData

SDM for Reading supports mirroring (part of the) file data encrypted. This part is called the SDMENCFileData.

If the SDMFileRead access right is configured for an application key, part of the file data can optionally be encrypted as defined in Section 9.3.7 when being read out in not authenticated state.

In this case, the input plaintext for the encryption is always in binary encoding, while the output ciphertext is ASCII encoded.

Note that if authenticated, no Secure Dynamic Messaging is applied, i.e. the regular secure messaging is always applied on the static file data.

The SDMENCFileData (if any) is always mirrored within the file. This is configured with ChangeFileSettings, see Section 11.7.1 via SDMENCOffset and SDMENCLength. If the SDMFileRead access right is disabling Secure Dynamic Messaging for reading (i.e. set to Fh), SDMENCOffset and SDMENCLength are not present in ChangeFileSettings.

If PICCData is mirrored within the file, SDMENCFileData shall not overlap with it. Depending on what is exactly mirrored, the following holds:

- SDMENCOffset ≥ PICCDataOffset + PICCDataLength OR PICCDataOffset ≥ SDMENCOffset + SDMENCLength.
- SDMENCOffset ≥ UIDOffset + UIDLength OR UIDOffset ≥ SDMENCOffset + SDMENCLength.
- SDMENCOffset ≥ SDMReadCtrOffset + SDMReadCtrLength OR SDMReadCtrOffset ≥ SDMENCOffset + SDMENCLength.

It shall be ensured that SDMENCOffset + SDMENCLength is smaller than or equal to the file size. Note that as the SDMMAC is as well mirrored into the file, additional conditions apply, see Section 9.3.8. The SDMENCLength shall be a multiple of 32 bytes for the ASCII encoding. With NT4H2421Tx, only ASCII encoding is supported.

9.3.7 Encryption of SDMENCFileData

The key applied for the encryption is a session key SesSDMFileReadENCKey derived from the application key defined by the SDMFileRead access right as specified in Section 9.3.10.

From user point of view, the SDMENCOffset and SDMENCLength define a placeholder within the file where the plain data is to be stored when writing the file.

For ASCII encoding, only the first half of the placeholder is used for storing the plain data, the second half is ignored for constructing the returned data when reading with SDM. For example, if targeting to encrypt 2 AES blocks, i.e. 32 bytes, a placeholder of 64 bytes is reserved via SDMENCOffset and SDMENCLength. The first 32 bytes hold the plaintext, and the next 32 bytes are ignored when reading with Secure Dynamic Messaging.

9.3.7.1 AES mode encryption

Encryption and decryption of the SDMENCFileData are calculated using the underlying block cipher according to the CBC mode of NIST SP800-38A [5]. The following IV is applied:

IV = E(SesSDMFileReadENCKey; SDMReadCtr||00000000000000000000000000h)

with SDMReadCtr LSB first.

For applying SDM with ASCII encoding, the SDMENCFileData is defined as follows:

SDMENCFileData = E(SesSDMFileReadENCKey; StaticFileData[SDMENCOffset:: SDMENCOffset + SDMENCLength=2 - 1])

with StaticFileData being the current file data as written in the placeholder. The file configuration ensures via SDMENCLength that the input is a multiple of 16 bytes, so no padding is applied.

Note that it is possible via the read command parameters to read-only part of the file. If the SDMENCFileData is partially read as per the issued offset and length, a truncated part of the ciphertext will be returned. As truncation might happen in the middle of an AES block, this means subsequent read commands to fetch the remainder of the file might be required to be able to decrypt.

9.3.7.2 LRP mode encryption

Encryption and decryption of the SDMENCFileData are calculated using a leakage resilient indexed codebook (LRICB) construction on top of the LRP primitive: LRICB of [10].

As input counter for the CTR construction, a 6-byte counter is used, consisting of the concatenation of SDMReadCtr and three zero bytes: SDMReadCtr || 000000h. SDMReadCtr is LSB first. After concatenation the 6-byte are treated as unsigned integer for the counting.

Apart from the counter applied, this is identical to the encryption used for LRP secure messaging, see Section 9.2.4.

If applying SDM with ASCII encoding, the SDMENCFileData is defined as follows:

SDMENCFileData = E_{LRP} *(SesSDMFileReadENCKey; StaticFileData[SDMENCOffset ... SDMENCOffset + SDMENCLength/2 - 1])*

with StaticFileData being the file data as it was written in the placeholder. The file configuration ensures via SDMENCLength that the input is a multiple of 16 bytes. No padding is applied.

9.3.8 SDMMAC

SDM for Reading supports calculating a MAC over the response data. This message authentication code is called the SDMMAC.

If SDMFileRead access right is configured for an application key, a MAC will be calculated as defined in Section 9.3.9 when being read out in no authenticated state.

The SDMMAC is to be mirrored within the file via SDMMACOffset. This is configured with ChangeFileSettings, see Section 11.7.1.

If SDMMAC is mirrored within the file, it is limited to start only after SDMENCFileData, i.e. SDMMACOffset ≥ SDMENCOffset + SDMENCLength. The SDMMACInputOffset must ensure that the complete SDMENCFileData is included in the MAC calculation.

As the mirrored SDMMAC is ASCII encoded, the output size doubles to 16 bytes.

It shall be ensured that SDMMACOffset + SDMMACLength is smaller or equal than the file size. Note that if authenticated, no Secure Dynamic Messaging is applied and the placeholder data at SDMMACOffset is not replaced, i.e. the regular secure messaging is always applied on the static file data.

The SDMMACInputOffset will define from which position in the file the MAC calculation starts. If SDMMAC is mirrored within the file, SDMMACInputOffset must be smaller than or equal to SDMMACOffset.

MACing is mandatory if the SDMFileRead access right is configured for an application key. If the SDMFileRead access right is disabling Secure Dynamic Messaging for reading (i.e. set to Fh), SDMMACOffset and SDMMACInputOffset are not present in ChangeFileSettings.

With NT4H2421Tx, only ASCII encoding is supported. SDMMAC is always mirrored within the file.

9.3.9 MAC Calculation

The key applied for the MAC calculation is a session key SesSDMFileReadMACKey derived from the application key defined by the SDMFileRead access right, as specified in Section 9.3.10.

9.3.9.1 AES mode MAC calculation

The 8-byte SDMMAC is calculated using AES according to the CMAC standard described in NIST Special Publication 800-38B [6] applying the same truncation as the AES mode secure messaging, see Section 9.1.3.

The SDMMAC is defined as follows:

SDMMAC = MACt (SesSDMFileReadMACKey; DynamicFileData[SDMMACInputOffset ... SDMMACOffset - 1])

with DynamicFileData being the file data as how it is put on the contactless interface, i.e. replacing any placeholders by the dynamic data.

9.3.9.2 LRP mode MAC calculation

The 8-byte SDMMAC is calculated using a CMAC construction on top of the LRP primitive. This is specified in [10].

It is identical to the MAC calculation of LRP secure messaging, see Section 9.2.3, also applying the same truncation.

Therefore SDMMAC is defined as follows:

SDMMAC = $MACt_{LRP}$ (SesSDMFileReadMACKey; DynamicFileData[SDMMACInputOffset ... SDMMACOffset - 1])

with DynamicFileData being the file data as how it is put on the contactless interface, i.e. replacing any placeholders by the dynamic data.

9.3.10 SDM Session Key Generation

For Secure Dynamic Messaging for reading, the following session keys are calculated:

- SesSDMFileReadMACKey for MACing of file data.
- SesSDMFileReadENCKey for encryption of file data

The session key generation is according to NIST SP 800-108 [8] in counter mode.

The pseudo random function applied during the key generation is the CMAC algorithm described in NIST Special Publication 800-38B [6]. The key derivation key is the SDMFileReadKey as configured with the SDMFileRead access right.

9.3.10.1 AES mode session key generation for SDM

The input data is constructed using the following fields as defined by [8]. Note that NIST SP 800-108 allows defining a different order than proposed by the standard as long as it is unambiguously defined.

- a 2-byte label, distinguishing the purpose of the key: 3CC3h for MACing and C33Ch for encryption.

- a 2-byte counter, fixed to 0001h as only 128-bit keys are generated.
- a 2-byte length, fixed to 0080h as only 128-bit keys are generated.
- a context, constructed using the UID and/or SDMReadCtr, followed by zero-byte padding if needed.

Firstly, the input session vectors SV x are derived as follows:

SV1 = C3h || 3Ch || 00h || 01h || 00h || 80h || UID || SDMReadCtr

SV2 = 3Ch || C3h || 00h || 01h || 00h || 80h [|| UID] [|| SDMReadCtr] [|| ZeroPadding]

Whether or not the UID and/or SDMReadCtr are included in session vector SV2, depends on whether they are mirrored, see Section 9.3.3. Note that in case of encrypting file data, mirroring of both is mandatory.

Therefore they are always included in SV1.

Padding with zeros is done up to a multiple of 16 bytes. So in case of 7-byte UID and both elements are mirrored, no padding is added. Then, the 16-byte session keys are constructed as follows:

SesSDMFileReadENCKey = MAC(SDMFileReadKey; SV1)

SesSDMFileReadMACKey = MAC(SDMFileReadKey; SV2)

9.3.10.2 LRP mode session key generation for SDM

The input data is constructed using the following fields as defined by [8]. Note that NIST SP 800-108 allows defining a different order than proposed by the standard as long as it is unambiguously defined.

- a 2-byte counter, fixed to 0001h as only 128-bit keys are generated.
- a 2-byte length, fixed to 0080h as only 128-bit keys are generated.
- a context, constructed using the UID and/or SDMReadCtr, followed by zero-byte padding if needed.
- a 2-byte label: 1EE1h

Firstly, the input session vector SV is derived as follows:

SV = 00h || 01h || 00h || 80h [|| UID] [|| SDMReadCtr] [|| ZeroPadding] || 1Eh || E1h

Whether or not the UID and/or SDMReadCtr are included in the session vectors, depends on whether they are mirrored, see Section 9.3.3. Note that in case of encrypting file data, mirroring of both is mandatory. Padding with zeros is done up to a multiple of 16 bytes. So in case of 7-byte UID and both elements are mirrored, no padding is added. Then, the session key material is constructed as follows:

SDMFileReadSPT = generatePlaintexts(4; SDMFileReadKey)

{SDMFileReadUpdateKey} = generateUpdatedKeys(1; SDMFileReadKey)

SesSDMFileReadMasterKey = MAC_{LRP} (SDMFileReadKey; SV)

SesSDMFileReadSPT = generatePlaintexts(4; SesSDMFileReadMasterKey)

{SesSDMFileReadMACUpdateKey; SesSDMFileReadENCUpdateKey} = generateUpdatedKeys(2; SesSDMFileReadMasterKey)

with generatePlaintexts and generateUpdatedKeys the functions from [10]. Note that the output of generateUpdatedKeys is shown in the order that the keys are generated.

The actual SesSDMFileReadMACKey then consists for LRP of the set of plaintexts SesSDMFileReadSPT (consisting of 16 16-byte values) and SesSDMFileReadMACUpdateKey. The SesSDMFileReadENCKey consists of the same set of plaintexts SesSDMFileReadSPT and SesSDMFileReadENCUpdateKey.

9.3.11 Output Mapping Examples

The following figure shows an example with the static file content and how it will be read.

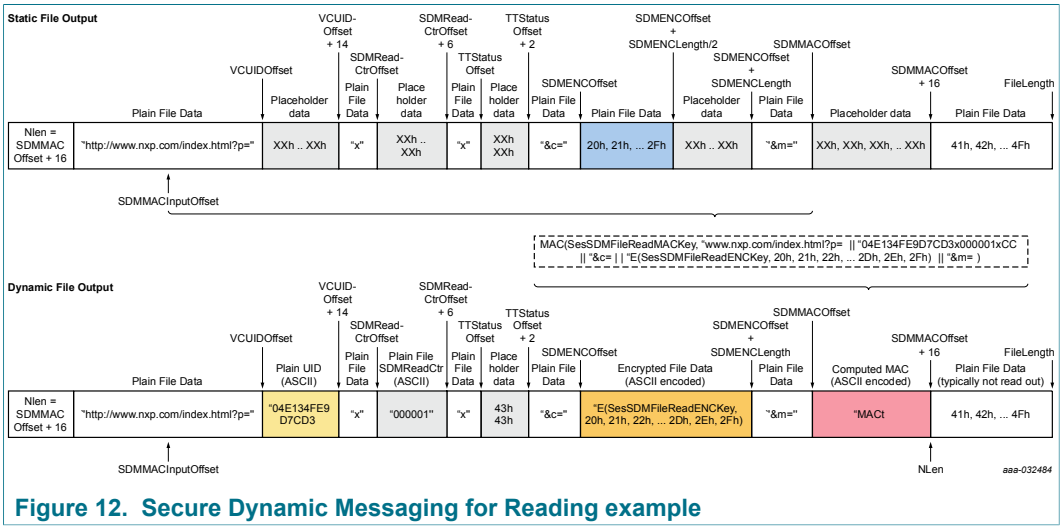


Figure 12. Secure Dynamic Messaging for Reading example

10 Tag Tamper Protection

NT4H2421Tx offers an NFC Forum-compliant solution to reflect e.g. if the sealing of a product is opened. This solution works without a dedicated application on a mobile phone. It only requires the capability of reading out NFC Forum Type 4 Tag [14]. NT4H2421Tx contains four pads, where two are used for antenna connection and the other two will connect a detection wire as illustrated in Figure 13. During the execution of the first command that is sent after ISO/IEC 14443-4 activation, the IC checks the tag tamper wire. If opened, this status will be recorded as permanent status in NVM. The result can be reflected in the NDEF message. Next to this, NT4H2421Tx also supports a specific command that triggers a measurement and returns both the permanent and current status. This chapter describes how the feature is enabled, when the check is executed and how the status can be retrieved.

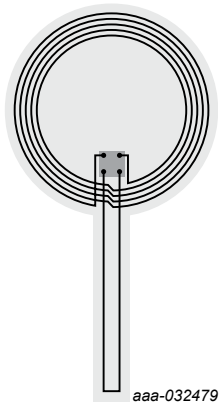


Figure 13. Tag Tamper illustration

10.1 Enabling the Tag Tamper feature

At delivery, NT4H2421Tx has the tag tamper feature disabled. The feature can be enabled by SetConfiguration with Option 07h, see Section 11.5.1. Once enabled, the NT4H2421Tx starts Tag Tamper measurements, see Section 10.2 from the next activation onwards. A measurement can always be triggered by issuing a GetTTStatus command.

When enabling the Tag Tamper feature using SetConfiguration, the access right related with TTStatusKey for GetTTStatus for Tag Tamper Status retrieval can be set, see Section 10.3.2. Enabling the Tag Tamper feature is permanent, it cannot be disabled once enabled.

10.2 Tag Tamper measurements

NT4H2421Tx maintains a permanent Tag Tamper status TTPermStatus. The TTPermStatus and the current status TTCurrStatus can be mirrored together in the NDEF file, and included in the Secure Dynamic Messaging, see Section 9.3. Both statuses can also be retrieved with GetTTStatus. The following values are supported:

- *Close*: 43h, i.e. ASCII encoding of ‘C’. This is the initial value when the seal is still closed.
- *Open*: 4Fh, i.e. ASCII encoding of ‘O’. This is the value when the seal was opened.
- *Invalid*: 49h, i.e. ASCII encoding of ‘I’. This is the value when the feature has not been enabled yet and no measurements are executed.

Once the feature has been enabled, NT4H2421Tx measures whether the seal is opened by applying a measurement on the detection wire. As the NTAG 424 DNA TT is a passive tag, it cannot trigger measurements itself. The measurement will be done, each time the tag is powered and booted. If the current TTPermStatus is still set to Close, the measurement will only be done after complete ISO/IEC 14443-4 activation and during processing of the first command.

If a measurement detects the seal to be opened, the TTPermStatus is updated to Open. Once set to Open, the measurement on boot will not be triggered anymore. The TTPermStatus cannot be reset to Close anymore.

The following commands also trigger a measurement once the feature is enabled:

- GetTTStatus, see Section 10.3.2.

- ReadData and ISOReadBinary, if required for the Secure Dynamic Messaging and Mirroring, see Section 9.3.5.

10.3 Tag Tamper status retrieval

10.3.1 Mirroring in the NDEF message

The TTPermStatus and the TTCurrStatus can be mirrored within the NDEF messaging. In this way, the status can be protected by the Secure Dynamic Messaging, as defined in Section 9.3 and more specifically Section 9.3.5. To enable this mirroring, the configuration needs to be done with ChangeFileSettings, see Section 11.7.1.

10.3.2 GetTTStatus command

The GetTTStatus command supports retrieving of the permanent and current Tag Tamper Status: TTPermStatus and TTCurrStatus. For the latter, once the feature has been enabled, a Tag Tamper measurement as defined in Section 10 will be triggered. If detected as open, the TTPermStatus will be updated to Open.

Under active authentication, the command GetTTStatus requires CommMode.Full.

11 Command set

11.1 Introduction

This section contains the full command set of NTAG 424 DNA TT.

Remark: In the figures and tables, always CommMode.Plain is presented and the field length is valid for the plain data length. For the CommMode.MAC and CommMode.Full, the cryptogram needs to be calculated according to the secure messaging Section 9, then data field needs to fill with the cryptogram (Plain; CMAC; encrypted data with CMAC). Communication mode and condition are mentioned in the command description.

11.2 Supported commands and APDUs

Table 22. APDUs

Command	C-APDU (hex)						R-APDU (hex)			Communication mode
INS	CLA	INS	P1	P2	Lc	Data	Le	Data	SW1SW2 Succ essful	
AuthenticateEV2First - Part1	90	71	00	00	XX	Data	00	Data	91AF	N/A (command specific)
AuthenticateEV2First - Part2	90	AF	00	00	20	Data	00	Data	9100	
AuthenticateEV2NonFirst - Part1	90	77	00	00	XX	Data	00	Data	91AF	N/A (command specific)
AuthenticateEV2NonFirst - Part2	90	AF	00	00	20	Data	00	Data	9100	
AuthenticateLRPFirst - Part1	90	71	00	00	XX	Data	00	Data	91AF	N/A (command specific)
AuthenticateLRPFirst - Part2	90	AF	00	00	20	Data	00	Data	9100	
AuthenticateLRPNonFirst - Part1	90	77	00	00	XX	Data	00	Data	91AF	N/A (command specific)
AuthenticateLRPNonFirst - Part2	90	AF	00	00	20	Data	00	Data	9100	
ChangeFileSettings	90	5F	00	00	XX	Data	00	-	9100	CommMode.Full
ChangeKey	90	C4	00	00	XX	Data	00	-	9100	CommMode.Full
GetCardUID	90	51	00	00	-	-	00	Data	9100	CommMode.Full
GetFileCounters	90	F6	00	00	-	File ID	00	Data	9100	CommMode.Full
GetFileSettings	90	F5	00	00	01	File number	00	Data	9100	CommMode.MAC
GetKeyVersion	90	64	00	00	01	Key number	00	Data	9100	CommMode.MAC
GetTTStatus	90	F7	00	00	-	-	00	Data	9100	CommMode.Full
GetVersion - Part1	90	60	00	00	-	-	00	Data	91AF	CommMode.MAC[1]
GetVersion - Part2	90	AF	00	00	-	-	00	Data	91AF	CommMode.MAC
GetVersion - Part3	90	AF	00	00	-	-	00	Data	9100	CommMode.MAC
ISOReadBinary	00	B0	XX	XX	-	-	XX	Data	9000	CommMode.Plain
ReadData	90	AD	00	00	XX	Reference	00	Data	9100	Comm. mode of targeted file
Read_Sig	90	3C	00	00	01	00	00	Data	9100	CommMode.Full
ISOSelectFile	00	A4	XX	XX	XX	Data to send	XX	FCI	9000	CommMode.Plain
SetConfiguration	90	5C	00	00	XX	Data	00	-	9100	CommMode.Full

Command	C-APDU (hex)								R-APDU (hex)	Communication mode
ISOUpdateBinary	00	D6	XX	XX	XX	Data to write	-	-	9000	CommMode.Plain
WriteData	90	8D	00	00	XX	Data	00	-	9100	Comm. mode of targeted file

[1] MAC on command and returned with the last response, calculated over all 3 responses

11.3 Status word

Table 23. SW1 SW2 for CLA byte 0x90

SW1 SW2	Name	Description
0x9100	OPERATION_OK	Successful operation.
0x911C	ILLEGAL_COMMAND_CODE	Command code not supported.
0x911E	INTEGRITY_ERROR	CRC or MAC does not match data. Padding bytes not valid.
0x9140	NO_SUCH_KEY	Invalid key number specified.
0x917E	LENGTH_ERROR	Length of command string invalid.
0x919D	PERMISSION_DENIED	Current configuration / status does not allow the re- quested command.
0x919E	PARAMETER_ERROR	Value of the parameter(s) invalid.
0x91AD	AUTHENTICATION_DELAY	Currently not allowed to authenticate. Keep trying until full delay is spent.
0x91AE	AUTHENTICATION_ERROR	Current authentication status does not allow the re- quested command.
0x91AF	ADDITIONAL_FRAME	Additionaldata frame is expected to be sent.
0x91BE	BOUNDARY_ERROR	Attempt toread/write data from/to beyond the file's/record's limits. Attempt to exceed the limits of a value file.
0x91CA	COMMAND_ABORTED	Previous Command was not fully completed. Not all Frames were requested or provided by the PCD.
0x91F0	FILE_NOT_FOUND	Specified file number does not exist.

Table 24. SW1 SW2 for CLA byte 0x00

SW1 SW2	Description
0x6700	Wrong length; no further indication
0x6982	Security status not satisfied
0x6985	Conditions of use not satisfied
0x6A80	Incorrect parameters in the command data field
0x6A82	File or application not found
0x6A86	Incorrect parameters P1-P2
0x6A87	Lc inconsistent with parameters P1-P2
0x6C00	Wrong Le field
0x6CXX	Wrong Le field; SW2 encodes the exact number of avail- able data bytes.
0x6D00	Instruction code not supported or invalid
0x6E00	Class not supported
0x9000	Normal processing (no further qualification)

11.4 Authentication commands

Authentication with the defined key is required to access the protected file according to access rights. Based on successful authentication session keys are generated, which are used for secure messaging between the terminal and NT4H2421Tx.

Remark: Default FWI settings for authentication and secure messaging are set according to GSMA specification v2.0 to the value 7h. This value is stored in the User ATS.

11.4.1 AuthenticateEV2First

This command initiates an authentication based on standard AES. After this authentication, AES secure messaging is applied. This authentication command is used to authenticate for the first time in a transaction and can always be used within a transaction. AuthenticateEV2First starts a transaction with a Transaction Identifier (TI) and AuthenticateEV2NonFirst continues the transaction with that TI. This 3-pass challenge-response-based mutual authentication command is completed in two parts:

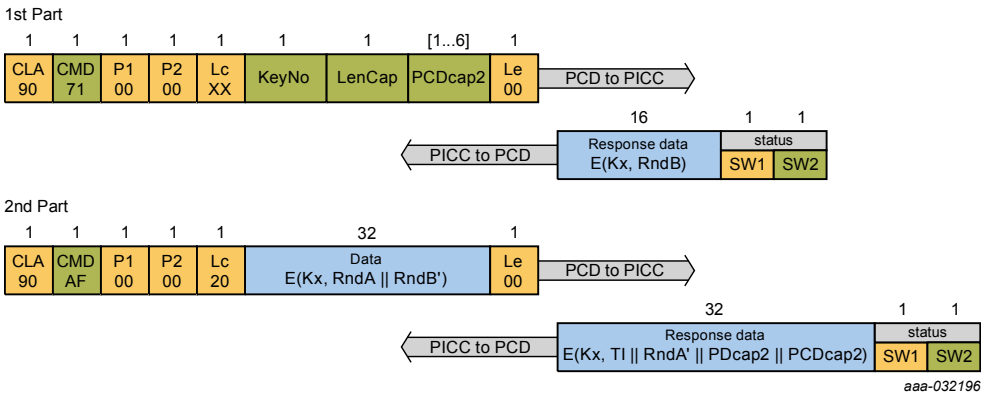


Figure 14. AuthenticateEV2First command protocol

Table 25. Command parameters description - AuthenticateEV2First - Part1

Name	Length	Value	Description
Command Header Parameters			
CMD	1	71h	Command code.
KeyNo	1		Targeted authentication key
	Bit 7-6	00b	RFU
	Bit 5-0	0h to 4h	Key number
LenCap	1	00h to 06h	Length of the PCD Capabilities. [This value should be set to 00h].
PCDcap2.1	[1]	-	Capability vector of the PCD.
	Bit 7-2	Full range	RFU, can hold any value
	Bit 1	0b	EV2 secure messaging
	Bit 0	Full range	RFU, can hold any value
PCDcap2.2-6	[1..5]	Full range	Capability vector of the PCD. All other bytes but PCDcap2.1 are optional, RFU and can hold any value. [If LenCap set to 00h, no PCDcap2 present]

Name	Length	Value	Description
Command Data Parameters			
-	-	-	No data parameters

Table 26. Response data parameters description - AuthenticateEV2First - Part1

Name	Length	Value	Description
E(Kx, RndB)	16	Full range	Encrypted PICC challenge The following data, encrypted with the key Kx referenced by KeyNo: - RndB: 16 byte random from PICC
SW1SW2	2	91AFh 91XXh	successful execution Refer to Table 26

Table 27. Return code description - AuthenticateEV2First - Part1

Status	Value	Description
COMMAND_ABORTED	CAh	Chained command or multiple pass command ongoing.
LENGTH_ERROR	7Eh	Command size not allowed.
PARAMETER_ERROR	9Eh	Parameter value not allowed.
NO_SUCH_KEY	40h	Targeted key does not exist
PERMISSION_DENIED	9Dh	Targeted key not available for authentication.
		Targeted key not enabled.
		Targeting EV2 authentication and secure messaging, while not allowed by configuration (PDCap2.1.Bit1 is '1').
AUTHENTICATION_DELAY	ADh	Currently not allowed to authenticate. Keep trying until full delay is spent.

Table 28. Command parameters description - AuthenticateEV2First - Part2

Name	Length	Value	Description
CMD	1	AFh	Additional frame
E(Kx, RndA || RndB')	32	Full range	Encrypted PCD challenge and response
			The following data, encrypted with the key Kx referenced by KeyNo: - RndA: 16 byte random from PCD. - RndB': 16 byte RndB rotated left by 1 byte

Table 29. Response data parameters description - AuthenticateEV2First - Part2

Name	Length	Value	Description
E(Kx, TI || RndA' || PDcap2 || PCDcap2)	32	Full range	Encrypted PICC response The following data encrypted with the key referenced by KeyNo: - TI: 4 byte Transaction Identifier - RndA': 16 byte RndA rotated left by 1 byte. - PDcap2: 6 byte PD capabilities - PCDcap2: 6 byte PCD capabilities
SW1SW2	2	9100h 91XXh	successful execution Refer to Table 30

Table 30. Return code description - AuthenticateEV2First - Part2

Status	Value	Description
LENGTH_ERROR	7Eh	Command size not allowed.
AUTHENTICATION_ERROR	AEh	Wrong RndB'
MEMORY_ERROR	EEh	Failure when reading or writing to non-volatile memory.

11.4.2 AuthenticateEV2NonFirst

The AuthenticateEV2NonFirst command can be used only if there is a valid authentication with AuthenticateEV2First. It continues the transaction with the transaction started by previous AuthenticateEV2First command. It starts a new session. The scheme of transaction and sessions within the transaction have been designed to protect any possible sophisticated replay attacks

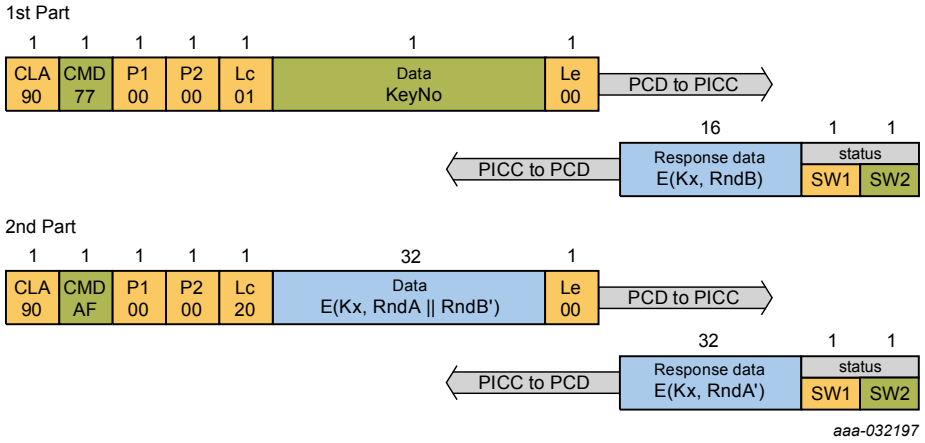


Figure 15. AuthenticateEV2NonFirst command protocol

Table 31. Command parameters description - AuthenticateEV2NonFirst - Part1

Name	Length	Value	Description
Command Header Parameters			
CMD	1	77h	Command code.
KeyNo	1		Targeted authentication key
	Bit 7-6	0	RFU
	Bit 5-0	0h to 04h	Key number
Command Data Parameters			
-	-	-	No data parameters

Table 32. Response data parameters description - AuthenticateEV2NonFirst - Part1

Name	Length	Value	Description
E(Kx, RndB)	16	Full range	Encrypted PICC challenge The following data, encrypted with the key Kx referenced by KeyNo: - RndB (16 byte): Random number from the PICC.
SW1SW2	2	91AFh 91XXh	successful execution Refer to Table 33

Table 33. Return code description - AuthenticateEV2NonFirst - Part1

Status	Value	Description
COMMAND_ABORTED	CAh	Chained command or multiple pass command ongoing.
LENGTH_ERROR	7Eh	Command size not allowed.
PARAMETER_ERROR	9Eh	Parameter value not allowed.
NO_SUCH_KEY	40h	Targeted key does not exist
PERMISSION_DENIED	9Dh	In not authenticated state and not targeting OriginalityKey
		Targeted key not available for authentication.
		Targeted key not enabled.
AUTHENTICATION_DELAY	ADh	Currently not allowed to authenticate. Keep trying until full delay is spent.
MEMORY_ERROR	EEh	Failure when reading or writing to non-volatile memory.

Table 34. Command parameters description - AuthenticateEV2NonFirst - Part2

Name	Length	Value	Description
CMD	1	AFh	Additional frame
E(Kx, RndA || RndB')	32	Full range	Encrypted PCD challenge and response
			The following data, encrypted with the key Kx referenced by KeyNo: - RndA: 16 byte random from PCD. - RndB': 16 byte RndB rotated left over 1 byte.

Table 35. Response data parameters description - AuthenticateEV2NonFirst - Part2

Name	Length	Value	Description
E(Kx, RndA')	16	Full range	Encrypted PICC challenge and response The following data, encrypted with the key Kx referenced by KeyNo: - RndA: 16 byte random from PCD. - RndB': 16 byte RndB rotated left over 1 byte.
SW1SW2	2	9100h 91XXh	successful execution Refer to Table 36

Table 36. Return code description - AuthenticateEV2NonFirst - Part2

Status	Value	Description
LENGTH_ERROR	7Eh	Command size not allowed.
AUTHENTICATION_ERROR	AEh	Wrong RndB'
MEMORY_ERROR	EEh	Failure when reading or writing to non-volatile memory.

11.4.3 AuthenticateLRPFirst

Authentication for LRP secure messaging. The AuthenticationLRPFirst is intended to be the first in a transaction and recommended. LRP secure messaging allows side-channel resistant implementations.

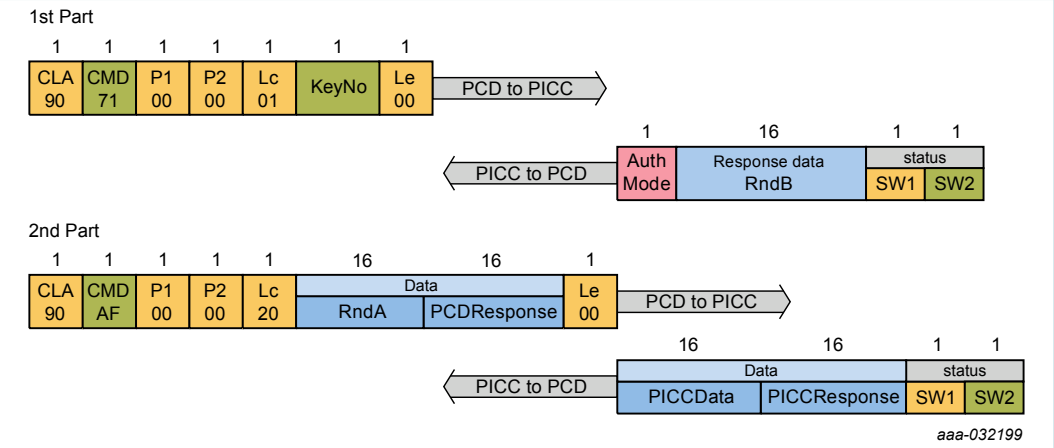


Figure 16. AuthenticateLRPFirst command protocol

Table 37. Command parameters description - AuthenticateLRPFirst - Part1

Name	Length	Value	Description
Command Header Parameters			
CMD	1	71h	Command code.
KeyNo	1		Targeted authentication key
	Bit 7-6	00b	RFU
	Bit 5-0	0h..4h	Key number
LenCap	1	1h..6h	Length of the PCD Capabilities.
PCDcap2.1	1	-	Capability vector of the PCD.
	Bit 7-2	Full range	RFU, can hold any value
	Bit 1	1b	LRP secure messaging
	Bit 0	Full range	RFU, can hold any value
PCDcap2.2-6	[1..5]	Full range	Capability vector of the PCD. All other bytes but PCDcap2.1 are optional, RFU and can hold any value.
Command Data Parameters			
-	-	-	No data parameters

Table 38. Response data parameters description - AuthenticateLRPFirst - Part1

Name	Length	Value	Description
AuthMode	1	01h	LRP Mode
RndB	16	Full range	PICC challenge RndB

Name	Length	Value	Description
SW1SW2	2	91AFh 91XXh	successful execution Refer to Table 39

Table 39. Return code description - AuthenticateLRPFirst - Part1

Status	Value	Description
COMMAND_ABORTED	CAh	Chained command or multiple pass command ongoing.
LENGTH_ERROR	7Eh	Command size not allowed.
PARAMETER_ERROR	9Eh	Parameter value not allowed.
NO_SUCH_KEY	40h	Targeted key does not exist.
PERMISSION_DENIED	9Dh	Targeted key is locked as related TotFailCtr is equal to or bigger than the TotFailCtrLimit.
AUTHENTICATION_DELAY	ADh	Currently not allowed to authenticate. Keep trying until full delay is spent.
MEMORY_ERROR	EEh	Failure when reading or writing to non-volatile memory.

Table 40. Command parameters description - AuthenticateLRPFirst - Part2

Name	Length	Value	Description
CMD	1	AFh	Additional frame
RndA	16	Full range	PCD challenge
PCDResponse	16	Full range	PCD response MAC_{LRP} *(SesAuthMACKey;RndA* || *RndB)*

Table 41. Response data parameters description - AuthenticateLRPFirst - Part2

Name	Length	Value	Description
PICCData	16	Full range	Encrypted PICC data, E_{LRP} (SesAuthENCKey; TI; PDCap2; PCDCap2)
PICCResponse	16	Full range	PICC response to the challenge MAC_{LRP} *(SesAuthMACKey; RndB* || *RndA* || *PICCData)*
SW1SW2	2	9100h 91XXh	successful execution Refer to Table 42

Table 42. Return code description - AuthenticateLRPFirstPart2

Status	Value	Description
LENGTH_ERROR	7Eh	Command size not allowed.
AUTHENTICATION_ERROR	AEh	Wrong PCDResp
MEMORY_ERROR	EEh	Failure when reading or writing to non-volatile memory.

11.4.4 AuthenticateLRPNonFirst

Consecutive authentication for LRP secure messaging. After this authentication, LRP secure messaging is used. This authentication is intended to be the following authentication in a transaction.

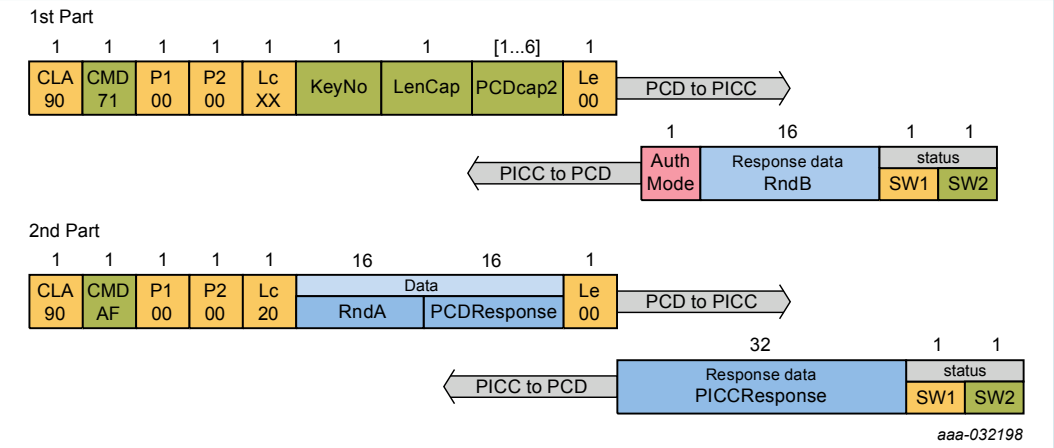


Figure 17. AuthenticationLRPNonFirst command protocol

Table 43. Command parameters description - AuthenticateLRPNonFirst - Part1

Name	Length	Value	Description
Command Header Parameters			
CMD	1	71h	Command code.
KeyNo	1		Targeted authentication key
	Bit 7-6	00b	RFU
	Bit 5-0	00h to 04h	Key Number
Command Data Parameters			
-	-	-	No data parameters

Table 44. Response data parameters description - AuthenticateLRPNonFirst - Part1

Name	Length	Value	Description
AuthMode	1	01h	LRP
RndB	16	Full range	PICC random RndB
SW1SW2	2	91AFh 91XXh	successful execution Refer to Table 45

Table 45. Return code description - AuthenticateLRPNonFirst - Part1

Status	Value	Description
COMMAND_ABORTED	CAh	Chained command or multiple pass command ongoing.
LENGTH_ERROR	7Eh	Command size not allowed.
PARAMETER_ERROR	9Eh	Parameter value not allowed.

Status	Value	Description
NO_SUCH_KEY	40h	Targeted key does not exist
PERMISSION_DENIED	9Dh	In not authenticated state and not targeting OriginalityKeys
		Targeted key not available for authentication.
		Targeted key not enabled.
AUTHENTICATION_DELAY	ADh	Currently not allowed to authenticate. Keep trying until full delay is spent.
MEMORY_ERROR	EEh	Failure when reading or writing to non-volatile memory.

Table 46. Command parameters description - AuthenticateLRPNonFirst - Part2

Name	Length	Value	Description
CMD	1	AFh	Additional frame
RndA	16	Full range	PCD challenge RndA
PCDResp	16	Full range	PCD response to the challenge MAC_{LRP} *(SesAuthMACKey;RndA* || *RndB)*

Table 47. Response data parameters description - AuthenticateLRPNonFirst - Part2

Name	Length	Value	Description
PICCResp	16	Full range	PICC response to the challenge MAC_{LRP} *(SesAuthMACKey;RndB* || *RndA)*
SW1SW2	2	9100h 91XXh	successful execution Refer to Table 48

Table 48. Return code description - AuthenticateLRPNonFirst - Part2

Status	Value	Description
LENGTH_ERROR	7Eh	Command size not allowed.
AUTHENTICATION_ERROR	AEh	Wrong PCDResp
MEMORY_ERROR	EEh	Failure when reading or writing to non-volatile memory.

11.5 Memory and configuration commands

11.5.1 SetConfiguration

With the SetConfiguration command, the application attributes can be configured. It requires an authentication with the AppMasterKey and CommMode.Full.

The command consists of an option byte and a data field with a size depending on the option. The option byte specifies the nature of the data field content.

In the below table “No change” references are used with configurations that are persistent. This means that the associated configuration is left as it is already in the card and its value is not changed.

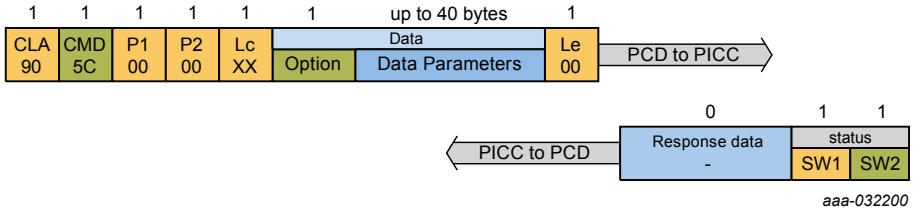


Figure 18. SetConfiguration command protocol

Table 49. Command parameters description - SetConfiguration

Name	Length	Value	Description
Command Header Parameters			
Cmd	1	5Ch	Command code.
Option	1	-	Configuration Option. It defines the length and content of the Data parameter. The Option byte is transmitted in plain text, whereas the Data is always transmitted in CommMode.Full.
		00h	PICC configuration.
		04h	Secure Messaging Configuration.
		05h	Capability data.
		07h	Tag Tamper Configuration
		0Ah	Failed authentication counter setting
		0Bh	HW configuration
		Other values	RFU
Command Data Parameters			
Data	Up to 10 bytes	-	Data content depends on option values.
		Full range	Data content depends on option value as defined in setConfigOptionsList Table.

Table 50. SetConfigOptionList

Option	Data Length	Field	Length/bitindex	Description
00h	1 byte	PICC Configuration		
		PICCConfig	Bit 7-2	RFU
			Bit 1	UseRID configuration for Random. Random ID is disabled at delivery time. 1b: Enable Random UID 0b: No change
			Bit 0	RFU
04h	2 bytes	Secure Messaging Configuration		
		SMConfig	Bit 15 to 3	RFU
			Bit 2	Secure messaging configuration for StandardData file 0b: No Change 1b: disable chained writing with WriteData command in CommMode.MAC and CommMode.Full
			Bit 1-0	RFU
05h	10 bytes	Capability data, consisting of PDCap2		
			4 bytes	RFU
			1 byte	User configured PDCap2.1 Bit 7 to 2: RFU Bit 1: 1b means enable LRP mode. This change is permanent, LRP mode cannot be disabled afterwards. Bit 1: 0b means no change
			3 bytes	RFU
			1 byte	User configured PDCap2.5
			1 byte	User configured PDCap2.6
07h	2 bytes	Tag Tamper Configuration		
		TTConfig	1 byte	Bit 7 to 1: RFU Bit 0: 1b Enabling Tag Tamper feature, see Section 10. Bit 0: 0b means no change
		TTStatusKey	1 byte	TTStatusKey 00h .. 04h: targeted key will be required for GetTTStatus 0Eh: free access to GetTTStatus (default) 0Fh: no access to GetTTStatus
0Ah	5 bytes	Failed authentication counter configuration		

Option	Data Length	Field	Length/bitindex	Description
		FailedCtrOption	1 byte	Bit 7 to 1: RFU Bit 0: Set to 0b for disabling Bit 0: Set to 1b for enabling [default]
		TotFailCtrLimit	2 bytes	configurable limit, encoded as 2-byte unsigned integer (LSB first), must be bigger than 0000h. Default value: 1000. When disabling, this value is ignored
		TotFailCtrDecr	2 bytes	configurable decrement value, encoded as 2-byte unsigned integer (LSB first). Default value: 10. When disabling, this value is ignored.
0Bh	1 byte	HW configuration		
		HW Option	1 byte	Bit 7 to 1: RFU Bit 0: Set to 0b for Standard back modulation Bit 0: Set to 1b for Strong back modulation (default)[1]

[1] note that it is strongly recommended to leave the default setting, specifically for antennas smaller than Class1

Table 51. Response data parameters description - SetConfiguration

Name	Length	Value	Description
Response data	0	-	No response data
SW1SW2	2	9100h 91XXh	successful execution Refer to Table 52

Table 52. Return code description - SetConfiguration

Status	Value	Description
COMMAND_ABORTED	CAh	Chained command or multiple pass command ongoing.
INTEGRITY_ERROR	1Eh	Invalid cryptogram (padding or CRC). Invalid secure messaging MAC.
LENGTH_ERROR	7Eh	Command size not allowed. Option 00h: Data length is not 1 Option 04h: Data length is not 2 Option 05h: Data length is not 10 Option 07h: Data length is not 2 Option 0Ah: Data length is not 5 Option 0Bh: Data length is not 1

Status	Value	Description
PARAMETER_ERROR	9Eh	Parameter value not allowed. Option 00h: Data bit 7-2 or bit 0 not set to 0b. Option 07h: targeting non-existing key Unsupported option (i.e. Reserved).
PERMISSION_DENIED	9Dh	Option 00h, 04h, 05h, 07h, 0Ah, 0Bh: not supported / allowed at PICC level
AUTHENTICATION_ERROR	AEh	Option 00h, 04h, 05h, 07h, 0Ah, 0Bh: No active authentication with AppMasterKey.
MEMORY_ERROR	EEh	Failure when reading or writing to non-volatile memory.

11.5.2 GetVersion

The GetVersion command returns manufacturing related data of NTAG 424 DNA TT (NT4H2421Tx). No parameters are required for this command.

Remark: This command is only available after ISO/IEC 14443-4 activation.

The version data is return over three frames. Part1 returns the hardware-related information, Part2 returns the software-related information and Part3 and last frame returns the production-related information. This command is freely accessible without secure messaging as soon as the PD is selected and there is no active authentication.

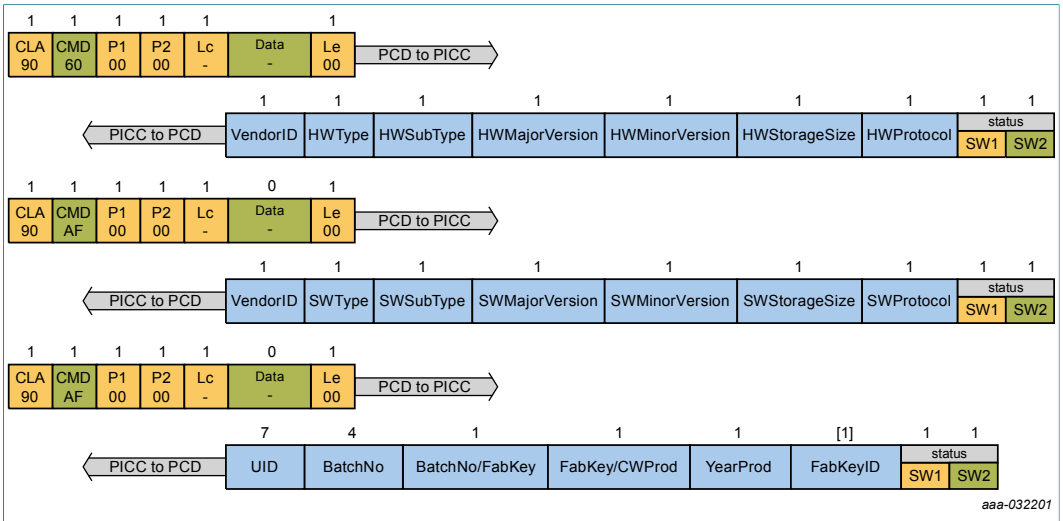


Figure 19. GetVersion command protocol

Part 1

Table 53. Command parameters description - GetVersion - Part1

Name	Length	Value	Description
Command Header Parameters			
Cmd	1	60h	Command code.
Command Data Parameters			
-	-	-	No data parameters

Table 54. Response data parameters description - GetVersion - Part1

Name	Length	Value	Description
VendorID	1	04h	Vendor ID
HWType	1	04h	HW type for NTAG
HWSubType	1	-	HW subtype
		X8h	50 pF + Tag Tamper
		0Xh	Strong back modulation
		8Xh	Standard back modulation
HWMajorVersion	1	30h	HW major version number

Name	Length	Value	Description
HWMinorVersion	1	00h	HW minor version number
HWStorageSize	1	-	HW storage size
		11h	256 B<storage size< 512 B
		other values	RFU
HWProtocol	1	05h	HW communication protocol type
SW1SW2	2	91AFh 91XXh	successful execution Refer to Table 59

Part 2

Table 55. Command parameters description - GetVersion - Part2

Name	Length	Value	Description
CMD	1	AFh	Additional frame request.
Data	0	-	No data parameters:

Table 56. Response data parameters description - GetVersion - Part2

Name	Length	Value	Description
VendorID	1	04h	Vendor ID
SWType	1	04h	SW type for NTAG
SWSubType	1	02h	SW subtype
SWMajorVersion	1	01h	SW major version number
SWMinorVersion	1	02h	SW minor version number
SWStorageSize	1	-	SW storage size
		11h	256 B<storage size< 512 B
		other values	RFU
SWProtocol	1	05h	SW communication protocol type
SW1SW2	2	91AFh 91XXh	successful execution Refer to Table 59

Part 3

Table 57. Command parameters description - GetVersion - Part3

Name	Length	Value	Description
CMD	1	AFh	Additional frame request.
Data	0	-	No data parameters:

Table 58. Response data parameters description - GetVersion - Part3

Name	Length	Value	Description
UID	7	-	UID
		All zero	if configured for RandomID
		Full range	UID if not configured for RandomID
BatchNo	4	Full range	Production batch number
BatchNo/FabKey	1		
	Bit 7-4	Full range	Production batch number
	Bit 3-0	0h	Default FabKey, other values RFU
FabKey/CWProd	1		
	Bit 7	0b	Default FabKey, other values RFU
	Bit 6-0	01h..52h	Calendar week of production
YearProd	1	Full range	Year of production
FabKeyID	[1]	1Fh..FFh	Optional, present for customized configurations when FabKey = 1Fh
SW1SW2	2	9100h 91XXh	successful execution Refer to Table 59

Table 59. Return code description - GetVersion

Status	Value	Description
COMMAND_ABORTED	CAh	Chained command or multiple pass command ongoing.
INTEGRITY_ERROR	1Eh	Invalid secure messaging MAC (only).
LENGTH_ERROR	7Eh	Command size not allowed.
MEMORY_ERROR	EEh	Failure when reading or writing to non-volatile memory.

11.5.3 GetCardUID

GetCardUID command is required to get the 7-byte UID from the card. In case "Random ID" at activation is configured, encrypted secure messaging is applied for this command and response. An authentication with any key needs to be performed prior to the command GetCardUID. This command returns the UID and gives the opportunity to retrieve the UID, even if the Random ID is used.

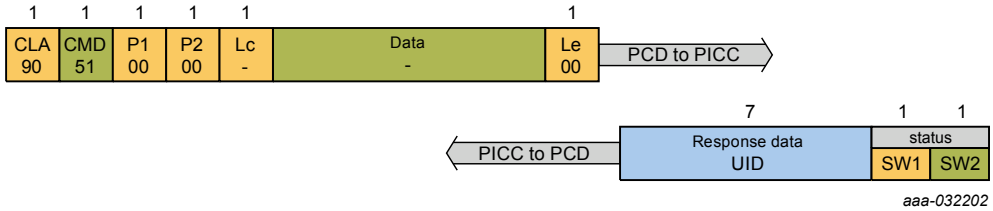


Figure 20. GetCardUID command protocol

Table 60. Command parameters description - GetCardUID

Name	Length	Value	Description
Command Header Parameters			
Cmd	1	51h	Command code.
Command Data Parameters			
-	-	-	No data parameters

Table 61. Response data parameters description - GetCardUID

Name	Length	Value	Description
UID	7	Full range	UID of the NT4H2421Tx
SW1SW2	2	9100h 91XXh	successful execution Refer to Table 62

Table 62. Return code description - GetCardUID

Status	Value	Description
COMMAND_ABORTED	CAh	Chained command or multiple pass command ongoing.
INTEGRITY_ERROR	1Eh	Invalid secure messaging MAC (only).
LENGTH_ERROR	7Eh	Command size not allowed.
AUTHENTICATION_ERROR	AEh	No active authentication
MEMORY_ERROR	EEh	Failure when reading or writing to non-volatile memory.

11.6 Key management commands

NT4H2421Tx provides the following command set for Key Management.

11.6.1 ChangeKey

The ChangeKey command is used to change the application keys. Authentication with application key number 0 is required to change the key. CommMode.Full is applied for this command. Note that the cryptogram calculations for changing key number 0 and other keys are different.

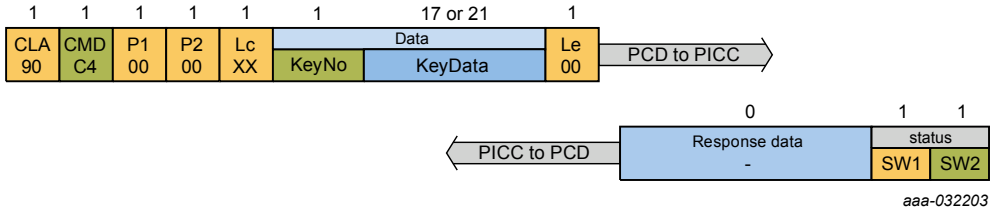


Figure 21. ChangeKey command protocol

Table 63. Command parameters description - ChangeKey

Name	Length	Value	Description
Command Header Parameters			
Cmd	1	C4h	Command code.
KeyNo	1	-	Key number of the key to be changed.
	Bit 7-6	00b	RFU
	Bit 5-0		Key Number
		0h..4h	The application key number
Command Data Parameters			
KeyData	17 or 21		New key data.
		full range (17-byte length)	if key 0 is to be changed NewKey || KeyVer
		full range (21-byte length)	if key 1 to 4 are to be changed (NewKey XOR OldKey) || KeyVer || CRC32NK[1]

[1] The CRC32NK is the 4-byte CRC value computed according to IEEE Std 802.3-2008 (FCS Field) over NewKey [9]

Table 64. Response data parameters description - ChangeKey

Name	Length	Value	Description
Response data	0	-	No response data
SW1SW2	2	9100h 91XXh	successful execution Refer to Table 65

Table 65. Return code description - ChangeKey

Status	Value	Description
COMMAND_ABORTED	CAh	Chained command or multiple pass command ongoing.
INTEGRITY_ERROR	1Eh	Integrity error in cryptogram or invalid secure messaging MAC (Secure Messaging).
LENGTH_ERROR	7Eh	Command size not allowed.
PARAMETER_ERROR	9Eh	Parameter value not allowed.
NO_SUCH_KEY	40h	Targeted key does not exist
PERMISSION_DENIED	9Dh	At PICC level, targeting any OriginalityKey which cannot be changed
AUTHENTICATION_ERROR	AEh	At application level, missing active authentication with AppMasterKey while targeting any AppKey.
MEMORY_ERROR	EEh	Failure when reading or writing to non-volatile memory.

11.6.2 GetKeyVersion

The GetKeyVersion command retrieves the current key version of any key. Key version can be changed with the ChangeKey command together with the key.

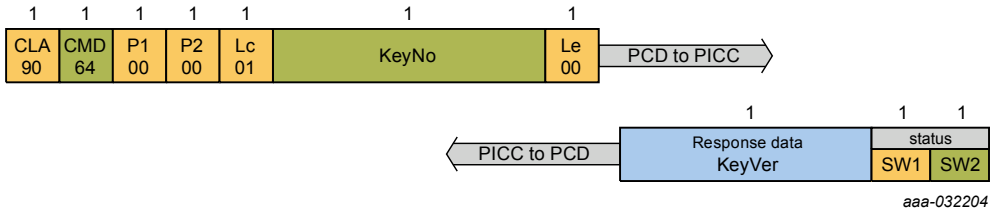


Figure 22. GetKeyVersion command protocol

Table 66. Command parameters description - GetKeyVersion

Name	Length	Value	Description
Command Header Parameters			
Cmd	1	64h	Command code.
KeyNo	1	-	Key number of the targeted key
	Bit 7-4	00h	RFU
	3 to 0	00h to 04h	Application key number
Command Data Parameters			
-	-	-	No data parameters

Table 67. Response data parameters description - GetKeyVersion

Name	Length	Value	Description
KeyVer	1	-	Key version of the targeted key
		00h	[if targeting disabled keys]
		00h	[if targeting OriginalityKey]
		Full range	[else]
SW1SW2	2	9100h 91XXh	successful execution Refer to Table 68

Table 68. Return code description - GetKeyVersion

Status	Value	Description
COMMAND_ABORTED	CAh	Chained command or multiple pass command ongoing.
INTEGRITY_ERROR	1Eh	Invalid secure messaging MAC (only).
LENGTH_ERROR	7Eh	Command size not allowed.
PARAMETER_ERROR	9Eh	Parameter value not allowed.
NO_SUCH_KEY	40h	Targeted key does not exist.
MEMORY_ERROR	EEh	Failure when reading or writing to non-volatile memory.

11.7 File management commands

The NT4H2421Tx provides the following command set for File Management functions.

11.7.1 ChangeFileSettings

The ChangeFileSettings command changes the access parameters of an existing file. The communication mode can be either CommMode.Plain or CommMode.Full based on current access right of the file.

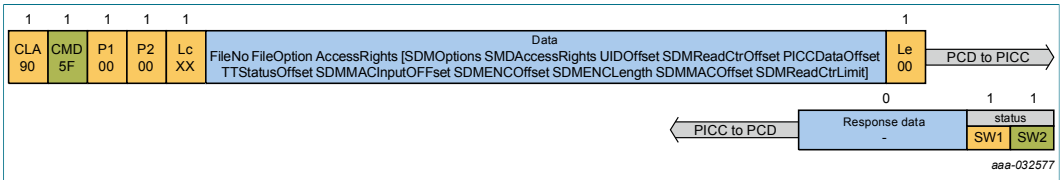


Figure 23. ChangeFileSettings command protocol

Table 69. Command parameters description - ChangeFileSettings

Name	Length	Value	Description
Command Header Parameters			
Cmd	1	5Fh	Command code.
FileNo	1	-	File number of the targeted file.
	Bit 7-5		RFU
	Bit 4-0		File number
Command Data Parameters			
FileOption	1	-	Options for the targeted file.
	Bit 7	0b	RFU
	Bit 6		Secure Dynamic Messaging and Mirroring
		0b	disabled
		1b	enabled
	Bit 5-2	0000b	RFU
	Bit 1-0		CommMode (see Table CommunicationModes)
AccessRights	2	-	Set of access conditions for the first set in the file (see Section 8.2.3.3).
SDMOptions	[1]	-	[Optional, present if FileOption[Bit 6] set] SDM Options
	Bit 7	-	UID (only for mirroring)
		0b	disabled
		1b	enabled
	Bit 6	-	SDMReadCtr
		0b	disabled
		1b	enabled

Name	Length	Value	Description
	Bit 5	-	SDMReadCtrLimit
		0b	disabled
		1b	enabled
	Bit 4	-	SDMENCFileData
		0b	disabled
		1b	enabled
	Bit 3	-	TTStatus
		0b	disabled
		1b	enabled
	Bit 2-1	00b	RFU
	Bit 0	-	Encoding mode
		1b	ASCII
SDMAccessRights	[2]	-	[Optional, present if FileOption[Bit 6] set] SDM Access Rights
	Bit 15- 12	-	SDMMetaRead access right
		0h..4h	Encrypted PICCData mirroring using the targeted AppKey
		Eh	Plain PICCData mirroring
		Fh	No PICCData mirroring
	Bit 11- 8	-	SDMFileRead access right
		0h..4h	Targeted AppKey
		Fh	No SDM for Reading
	Bit 7-4	Fh	RFU
	Bit 3-0	-	SDMCtrRet access right
		0h..4h	Targeted AppKey
		Eh	Free
		Fh	No Access
UIDOffset	[3]	-	[Optional, present if ((SDMOptions[Bit 7] = 1b) AND (SDMMetaRead access right = Eh)] Mirror position (LSB first) for UID
		0h .. (FileSize - UIDLength)	Offset within the file
SDMReadCtrOffset	[3]	-	[Optional, present if ((SDMOptions[Bit 6] = 1b) AND (SDMMetaRead access right = Eh)] Mirror position (LSB first) for SDMReadCtr
		0h .. (FileSize - SDMReadCtrLength)	Offset within the file

Name	Length	Value	Description
		FFFFFFh	No SDMReadCtr mirroring
PICCDataOffset	[3]	-	[Optional, present if SDMMetaRead access right =0h..4h] Mirror position (LSB first) for encrypted PICCData
		0h .. (FileSize - PICCDataLength)	Offset within the file
TTStatusOffset	[3]	-	[Optional, present if (SDMOptions[Bit 3] = 1b)] Mirror position (LSB first) for TTStatus
		0h .. (FileSize-2)	Offset within the file
SDMMACInputOffset	[3]	-	[Optional, present if SDMFileRead access right != Fh] Offset in the file where the SDM MAC computation starts (LSB first)
		0h .. (SDMMACOffset)	Offset within the file
SDMENCOffset	[3]	-	[Optional, present if ((SDMFileRead access right != Fh) AND (SDMOptions[Bit 4] = 1b))] SDMENCFileData mirror position (LSB first)
		SDMMACInputOffset .. (SDMMACOffset - 32)	Offset within the file
SDMENCLength	[3]	-	[Optional, present if ((SDMFileRead access right != Fh) AND (SDMOptions[Bit 4] = 1b))] Length of the SDMENCFileData (LSB first)
		32 .. (SDMMACOffset - SDMENCOffset)	Offset within the file, must be multiple of 32
SDMMACOffset	[3]	-	[Optional, present if SDMFileRead access right != Fh] SDMMAC mirror position (LSB first)
		SDMMACInputOffset .. (FileSize - 16)	[if (SDMFileRead access right != Fh) AND (SDMOptions[Bit 4] = 0b)] Offset within the file
		(SDMENCOffset + SDMENCLength) .. (FileSize- 16)	[if (SDMFileRead access right != Fh) AND (SDMOptions[Bit 4] = 1b)] Offset within the file
SDMReadCtrLimit	[3]	Full range	[Optional, present if SDMOptions[Bit 5] = 1b] SDMReadCtrLimit value (LSB first)

Table 70. Response data parameters description - ChangeFileSettings

Name	Length	Value	Description
Response data	0	-	No response data
SW1SW2	2	9100h 91XXh	successful execution Refer to Table 71

Table 71. Return code description - ChangeFileSettings

Status	Value	Description
COMMAND_ABORTED	CAh	Chained command or multiple pass command ongoing.
INTEGRITY_ERROR	1Eh	Integrity error in cryptogram. Invalid Secure Messaging MAC (only).
LENGTH_ERROR	7Eh	Command size not allowed.
PARAMETER_ERROR	9Eh	Parameter value not allowed.
		Targeted key for one of the access conditions in AccessRights or SDMAccessRights does not exist.
		Trying to set access right SDMMetaRead to a value different than Fh, while both UID and SDMReadCtr mirroring are disabled.
		Trying to set access right SDMMetaRead to Fh, while enabling UID mirroring.
		Trying to set access right SDMCtrRet to a value different from Fh, while SDMReadCtr is disabled.
		SDMMAC and UID mirroring are overlapping, i.e. the following condition is not satisfied: (SDMMACOffset ≥ UIDOffset + UIDLength) OR (UIDOffset ≥ SDMMACOffset + SDMMACLength)
		SDMMAC and SDMReadCtr mirroring are overlapping, i.e. the following condition is not satisfied: (SDMMACOffset ≥ SDMReadCtrOffset + SDMReadCtrLength) OR (SDMReadCtrOffset ≥ SDMMACOffset + SDMMACLength)
		SDMMAC and PICCData mirroring are overlapping, i.e. the following condition is not satisfied: (SDMMACOffset ≥ PICCDataOffset + PICCDataLength) OR (PICCDataOffset ≥ SDMMACOffset + SDMMACLength)
		SDMMAC and TTStatus mirroring are overlapping, i.e. the following condition is not satisfied: (SDMMACOffset ≥ TTStatusOffset + 2) OR (TTStatusOffset ≥ SDMMACOffset + SDMMACLength)
		SDMENCFileData and UID mirroring are overlapping, i.e. the following condition is not satisfied: (SDMENCOffset ≥ UIDOffset + UIDLength) OR (UIDOffset ≥ SDMENCOffset + SDMENCLength)

Status	Value	Description
		SDMENCFileData and SDMReadCtr mirroring are overlapping, i.e. the following condition is not satisfied: (SDMENCOffset ≥ SDMReadCtrOffset + SDMReadCtrLength) OR (SDMReadCtrOffset ≥ SDMENCOffset + SDMENCLength)
		SDMENCFileData and PICCData mirroring are overlapping, i.e. the following condition is not satisfied: (SDMENCOffset ≥ PICCDataOffset + PICCDataLength) OR (PICCDataOffset ≥ SDMENCOffset + SDMENCLength)
		TTStatus and UID mirroring are overlapping, i.e. the following condition is not satisfied: (TTStatusOffset ≥ UIDOffset + UIDLength) OR (UIDOffset ≥ TTStatusOffset + 2
		TTStatus and SDMReadCtr mirroring are overlapping, i.e. the following condition is not satisfied: (TTStatusOffset ≥ SDMReadCtrOffset + SDMReadCtrLength) OR (SDMReadCtrOffset ≥ TTStatusOffset + 2)
		TTStatus and PICCData mirroring are overlapping, i.e. the following condition is not satisfied: (TTStatusOffset ≥ PICCDataOffset + PICCDataLength) OR (PICCDataOffset ≥ TTStatusOffset + 2) (DFCMD.2495) PARAMETER_ERROR
		TTStatus is part of SDMENCFileData but not part of the plaintext placeholder, i.e. following condition is satisfied: (TTStatusOffset + 1 ≥ SDMENCOffset + SDMENCLength/2) AND ((TTStatusOffset<SDMENCOffset + SDMENCLength)
		UID and SDMReadCtr mirroring are overlapping, i.e. the following condition is not satisfied: (UIDOffset ≥ SDMReadCtrOffset + SDMReadCtrLength) OR (SDMReadCtrOffset ≥ UIDOffset + UIDLength)
		Enabling Secure Dynamic Messaging encryption (SDMOptions[Bit 4] set to 1b) is not possible if access right SDMFileRead = Fh.
		Enabling Secure Dynamic Messaging encryption (SDMOptions[Bit 4] set to 1b) is not allowed if not both SDMReadCtr and UID are mirrored (i.e. SDMOptions[Bit 7] and SDMOptions[Bit 6] must be set to 1b)
		Trying to set a SDMReadCtrLimit while not enabling SDMReadCtr.
		Trying to set a SDMReadCtrLimit which is smaller or equal to the current SDMReadCtr.
PERMISSION_DENIED	9Dh	PICC level (MF) is selected.
		access right Change of targeted file has access conditions set to Fh.
		Enabling Secure Dynamic Messaging (FileOption Bit 6 set to 1b) is only allowed for FileNo 02h.

Status	Value	Description
FILE_NOT_FOUND	F0h	File with targeted FileNo does not exist for the targeted application.
AUTHENTICATION_ERROR	AEh	File access right Change of targeted file not granted as there is no active authentication with the required key while the access conditions is different from Fh.
MEMORY_ERROR	EEh	Failure when reading or writing to non-volatile memory.

11.7.2 GetFileSettings

The GetFileSettings command allows getting information on the properties of a specific file. The information provided by this command depends on the type of the file which is queried.

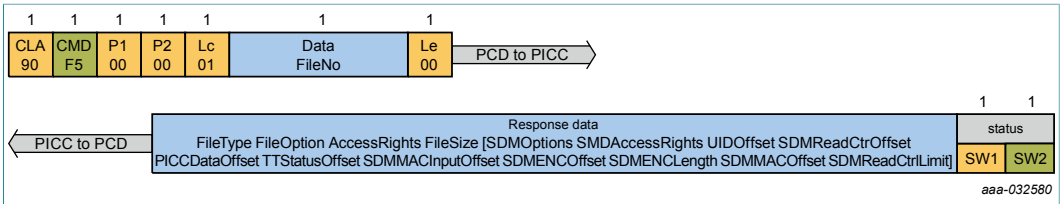


Figure 24. GetFileSettings command protocol

Table 72. Command parameters description - GetFileSettings

Name	Length	Value	Description
Command Header Parameters			
Cmd	1	F5h	Command code.
FileNo	1	-	File number of the targeted file.
	Bit 7-5		RFU
	Bit 4-0		File number
Command Data Parameters			
-	-	-	No data parameters

Table 73. Response data parameters description - GetFileSettings

Name	Length	Value	Description
FileType	1	-	File Type of the targeted file.
		00h	StandardData File
		Other values	RFU
FileOption	1	-	Options for the targeted file.
	Bit 7		RFU
	Bit 6	-	Secure Dynamic Messaging and Mirroring
		0b	disabled
		1b	enabled
	Bit 5-2	0000b	RFU

Name	Length	Value	Description
	Bit 1-0		CommMode (see Table CommunicationModes)
AccessRights	2	-	Set of access conditions for the 1st set in the file (see Section 8.2.3.3).
FileSize	3	-	File size of the targeted file.
SDMOptions	[1]	-	[Optional, present if FileOption[Bit 6] set] SDM Options, see Table 69
SDMAccessRights	[2]	-	[Optional, present if FileOption[Bit 6] set] SDM Access Rights, see Table 69
UIDOffset	[3]	-	[Optional, present if ((SDMOptions[Bit 7] = 1b) AND (SDMMetaRead access right = Eh)] Mirror position (LSB first) for UID, see Table 69
SDMReadCtrOffset	[3]	-	[Optional, present if ((SDMOptions[Bit 6] = 1b) AND (SDMMetaRead access right = Eh)] Mirror position (LSB first) for SDMReadCtr, see Table 69
PICCDataOffset	[3]	-	[Optional, present if SDMMetaRead access right =0h..4h] Mirror position (LSB first) for encrypted PICCData, see Table 69
TTStatusOffset	[3]	-	[Optional, present if (SDMOptions[Bit 3] = 1b)] Mirror position (LSB first) for TTStatus, see Table 69
SDMMACInputOffset	[3]	-	[Optional, present if SDMFileRead access right != Fh] Offset in the file where the SDM MAC computation starts (LSB first), see Table 69
SDMENCOffset	[3]	-	[Optional, present if ((SDMFileRead access right != Fh) AND (SDMOptions[Bit 4] = 1b))] SDMENCFileData mirror position (LSB first), see Table 69
SDMENCLength	[3]	-	[Optional, present if ((SDMFileRead access right != Fh) AND (SDMOptions[Bit 4] = 1b))] Length of the SDMENCFileData (LSB first), see Table 69
SDMMACOffset	[3]	-	[Optional, present if SDMFileRead access right != Fh] SDMMAC mirror position (LSB first), see Table 69

Name	Length	Value	Description
SDMReadCtrLimit	[3]	-	[Optional, present if SDMOptions[Bit 5] = 1b] SDMReadCtrLimit value (LSB first), see Table 69
SW1SW2	2	9100h 91XXh	successful execution Refer to Table 74

Table 74. Return code description - GetFileSettings

Status	Value	Description
COMMAND_ABORTED	CAh	Chained command or multiple pass command ongoing.
INTEGRITY_ERROR	1Eh	Invalid secure messaging MAC (only).
LENGTH_ERROR	7Eh	Command size not allowed.
PARAMETER_ERROR	9Eh	Parameter value not allowed.
PERMISSION_DENIED	9Dh	PICC level (MF) is selected.
FILE_NOT_FOUND	F0h	File with targeted FileNo does not exist for the targeted application.
MEMORY_ERROR	EEh	Failure when reading or writing to non-volatile memory.

11.7.3 GetFileCounters

The GetFileCounters command supports retrieving of the current values associated with the SDMReadCtr related with a StandardData file after enabling Secure Dynamic Messaging, see Section 9.3 and Section 11.7.1.

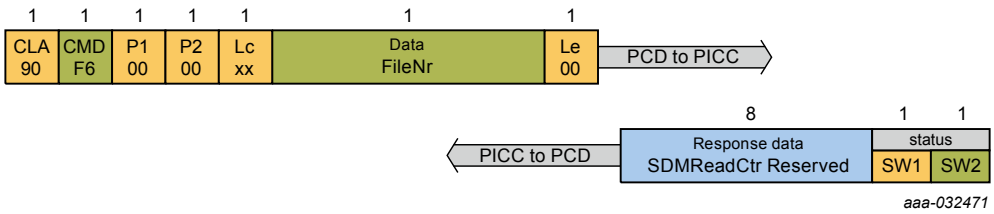


Figure 25. GetFileCounters command protocol

Table 75. Command parameters description - GetFileCounters

Name	Length	Value	Description
Command Header Parameters			
Cmd	1	F6h	Command code.
FileNo	1	-	File number of the targeted file.
	Bit 7-5	000b	RFU
	Bit 4-0	Limited range	File number
Command Data Parameters			
-	-	-	No data parameters

Table 76. Response data parameters description - GetFileCounters

Name	Length	Value	Description
SDMReadCtr	3	Full Range	Current SDMReadCtr of the targeted file (LSB first).
Reserved	2	0000h	RFU
SW1SW2	2	9100h 91XXh	successful execution Refer to Table 77

Table 77. Return code description - GetFileCounters

Status	Value	Description
COMMAND_ABORTED	CAh	Chained command or multiple pass command ongoing
INTEGRITY_ERROR	1Eh	Invalid secure messaging MAC
LENGTH_ERROR	7Eh	Command size not allowed
PARAMETER_ERROR	9Eh	Parameter value not allowed
PERMISSION_DENIED	9Dh	PICC level (MF) is selected.
		Targeted file has no Secure Dynamic Messaging enabled.
		Targeted file has SDMCtrRet access right set to Fh.
FILE_NOT_FOUND	F0h	Targeted file does not exist in the targeted application
AUTHENTICATION_ERROR	AEh	SDMCtrRet access right not granted while different from Fh.
FILE_NOT_FOUND	F0h	File with targeted FileNo does not exist for the targeted application.
MEMORY_ERROR	EEh	Failure when reading or writing to non-volatile memory

11.8 Data management commands

The NT4H2421Tx provides the following command set for Data Management functions.

11.8.1 ReadData

The ReadData command allows reading data from StandardData Files. The Read command requires a preceding authentication either with the key specified for Read or ReadWrite access, see the access rights section Section 8.2.3.3. Depending on the communication mode settings of the file secure messaging is applied, see Section 8.2.3.5.

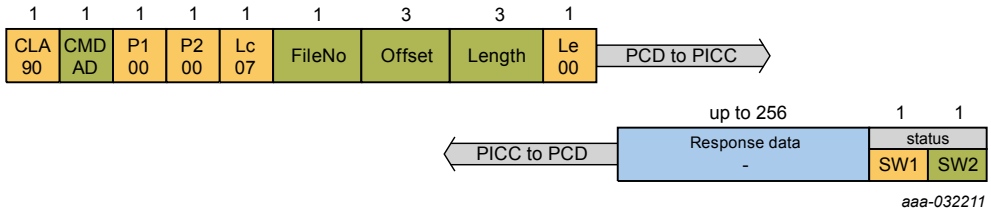


Figure 26. ReadData command protocol

Table 78. Command parameters description - ReadData

Name	Length	Value	Description
Command Header Parameters			
Cmd	1	ADh	Command code.
FileNo	1	-	File number of the targeted file.
	Bit 7-5	000b	RFU
	Bit 4-0		File number
		Full Range	
Offset	3	000000h .. (FileSize - 1)	Starting position for the read operation.
Length	3	-	Number of bytes to be read.
		000000h	Read the entire StandardData file, starting from the position specified in the offset value. Note that the short length Le limits response data to 256 byte including secure messaging (if applicable).
		000001h .. (FileSize - Offset)	
Command Data Parameters			
-	-	-	No data parameters

Table 79. Response data parameters description - ReadData

Name	Length	Value	Description
Response data	up to 256 byte including secure messaging	Full Range	Data read from the file
SW1SW2	2	9100h 91XXh	successful execution Refer to Table 80

Table 80. Return code description - ReadData

Status	Value	Description
COMMAND_ABORTED	CAh	Chained command or multiple pass command ongoing
INTEGRITY_ERROR	1Eh	Invalid secure messaging MAC (only)
		SMDRdCtr overflow
LENGTH_ERROR	7Eh	Command size not allowed
PARAMETER_ERROR	9Eh	Parameter value not allowed
PERMISSION_DENIED	9Dh	PICC level (MF) is selected.
		Read, ReadWrite and SDMFileRead (if SDM is enabled) access right of targeted StandardData file only have access conditions set to Fh.
		Targeted file cannot be read in not authenticated state as the related SDMReadCtr is equal or bigger than its SDMReadCtrLimit.
FILE_NOT_FOUND	F0h	Targeted file does not exist in the targeted application
AUTHENTICATION_ERROR	AEh	Read, ReadWrite and SDMFileRead (if SDM enabled) access right of targeted file not granted while at least one of the access conditions is different from Fh.
MEMORY_ERROR	EEh	Failure when reading or writing to non-volatile memory

11.8.2 WriteData

The WriteData command allows writing data to StandardData Files. NT4H2421Tx supports tearing protection for data that is sent within one communication frame to the file. Consequently, when using ISO/IEC 14443-4 chaining to write to a StandardData file, each frame itself is tearing protected but an interrupted chaining can lead to inconsistent files. Using single-frame WriteData commands instead of using the chaining can enable better control of the overall write process.

Depending on the communication mode settings of the Data file, data needs to be sent with either CommMode.Plain, CommMode.MAC or CommMode.Full. All cryptographic operations are done in CBC mode. In case of CommMode.MAC or CommMode.Full, the validity of data is verified by the PICC by checking the MAC. If the verification fails, the PICC stops further user memory programming and returns an Integrity Error to the PCD. As a consequence of the Integrity Error, any transaction, which might have begun, is automatically aborted. This can lead to the same situation as described above for an interrupted WriteData using chained communication.

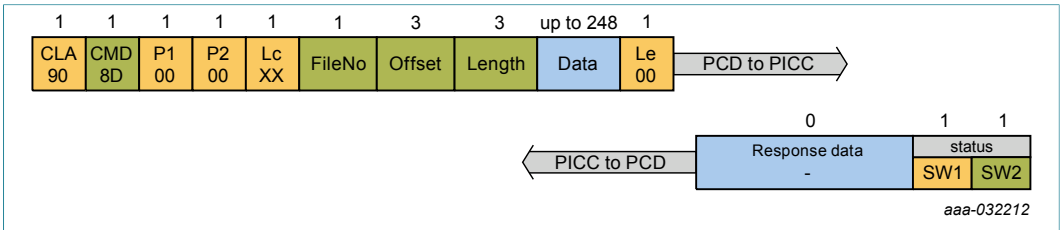


Figure 27. WriteData command protocol

Table 81. Command parameters description - WriteData

Name	Length	Value	Description
Command Header Parameters			
Cmd	1	8Dh	Command code.
FileNo	1	-	File number of the targeted file.
	Bit 7-5	000b	RFU
	Bit 4-0		File number
		Full range	
Offset	3	000000h .. (FileSize - 1)	Starting position for the write operation.
Length	3	000001h .. (FileSize - Offset)	Number of bytes to be written.
Command Data Parameters			
Data	up to 248 byte including secure messaging	Full range	Data to be written.

Table 82. Response data parameters description - WriteData

Name	Length	Value	Description
No response data parameters defined for this command			
Response data	0	-	No response data
SW1SW2	2	9100h 91XXh	successful execution Refer to Table 83

Table 83. Return code description - WriteData

Status	Value	Description
COMMAND_ABORTED	CAh	Chained command or multiple pass command ongoing.
INTEGRITY_ERROR	1Eh	Invalid secure messaging MAC or encryption padding.
LENGTH_ERROR	7Eh	Command size not allowed.
PARAMETER_ERROR	9Eh	Parameter value not allowed.
PERMISSION_DENIED	9Dh	PICC level (MF) is selected.
		Write and ReadWrite of targeted file only have access conditions set to Fh.
		Targeting a StandardData file with a chained command in MAC or Full while this is not allowed.
FILE_NOT_FOUND	F0h	Targeted file does not exist in the targeted application.
AUTHENTICATION_ERROR	AEh	Write and ReadWrite of targeted file not granted while at least one of the access conditions is different from Fh.
BOUNDARY_ERROR	BEh	Attempt to write beyond the file boundary as set during creation.
MEMORY_ERROR	EEh	Failure when reading or writing to non-volatile memory.

11.9 Tag Tamper protection commands

11.9.1 GetTTStatus

The GetTTStatus command supports retrieving of the the permanent and current Tag Tamper Status: TTPermStatus and TTCurrStatus. For the latter, once the feature has been enabled, a Tag Tamper measurement as defined in Section 10.2 will be triggered. If detected as open, the TTPermStatus will be updated to Open. Encrypted secure messaging is applied for this command and response.

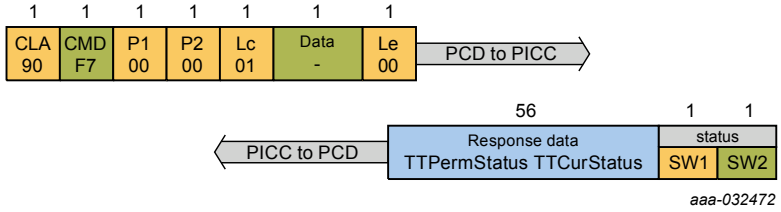


Figure 28. GetTTStatus command protocol

Table 84. Command parameters description - GetTTStatus

Name	Length	Value	Description
Command Header Parameters			
Cmd	1	F7h	Command code.
Command Data Parameters			
-	-	-	No data parameters

Table 85. Response data parameters description - GetTTStatus

Name	Length	Value	Description
TTPermStatus	1	-	TTPermStatus
		43h	Close
		4Fh	Open
		49h	Invalid
TTCurrStatus	1	-	TTCurrStatus
		43h	Close
		4Fh	Open
		49h	Invalid
SW1SW2	2	9100h 91XXh	successful execution Refer to Table 86

Table 86. Return code description - GetTTStatus

Status	Value	Description
COMMAND_ABORTED	CAh	Chained command or multiple pass command ongoing.
INTEGRITY_ERROR	1Eh	Invalid secure messaging MAC
LENGTH_ERROR	7Eh	Command size not allowed.
PERMISSION_DENIED	9Dh	PICC level (MF) is selected. TTStatusKey is configured for no access (0Fh).
AUTHENTICATION_ERROR	AEh	No active authentication with TTStatusKey while different from 0Eh and 0Fh.
MEMORY_ERROR	EEh	Failure when reading or writing to non-volatile memory.

11.10 Inter-industry standard commands

NT4H2421Tx provides the following ISO/IEC 7816-4 wrapped commands.

11.10.1 ISOSelectFile

This command is implemented in compliance with ISO/IEC 7816-4. It selects either the PICC level, an application or a file within the application.

If P1 is set to 00h, 01h or 02h, selection is done by a 2-byte ISO file identifier. For PICC level / MF selection, 3F00h or empty data has to be used. For Deticated application File (DF) and Elementary File (EF) selection, data holds the 2-byte ISO/IEC 7816-4 file identifier.

If P1 is set to 04h, selection is done by Deticated File (DF) name which can be up to 16 bytes. The registered ISO DF name is D2760000850100h. When selecting this DF name, the PICC level (or MF) is selected. For selecting the application immediately, the ISO/IEC 7816-4 DF name D2760000850101h can be used.

P2 indicates whether or not File Control Information (FCI) is to be returned in case of application selection. NT4H2421Tx does not support FCI and thus never returns any data, but does support both selection options to achieve broadest compatibility. The number of bytes requested by Le up to the complete file data will be returned in plain. There is no specific FCI template format checked, i.e. the data stored in the file will be sent back as is. In case of PICC level or file selection, FCI data is never returned. For NT4H2421Tx, no FCI will be returned as the pre-installed application does not contain such a file.

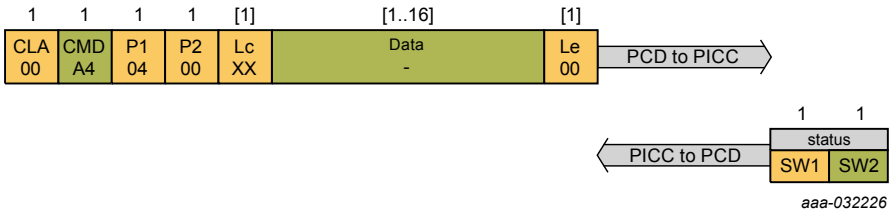


Figure 29. ISOSelectFile command protocol

Table 87. Command parameters description - ISOSelectFile

Name	Length	Value	Description
CLA	1	00h	
INS	1	A4h	
P1	1	-	Selection Control
		00h	Select MF, DF or EF, by file identifier
		01h	Select child DF
		02h	Select EF under the current DF, by file identifier
		03h	Select parent DF of the current DF
		04h	Select by DF name, see [3]
P2	1	-	Option
		00h	Return FCI template: data stored in the file with ID 1Fh should be returned
		0Ch	No response data: no FCI should be returned

Name	Length	Value	Description
Lc	[1]	00h .. 10h	Length of subsequent data field
Data	[1..16]	-	Reference
		Empty	[if P1 == 00h OR P1 == 03h] Select MF
		Full range	[if P1 == 00h OR P1 == 01h OR P1== 02h] Select with the given file identifier
		Full range	[if P1 == 04h] Select DF with the given DF name
Le	[1]	Full range	Empty or length of expected response

Table 88. Response data parameters description - ISOSelectFile

Name	Length	Value	Description
SW1SW2	2	9000h XXXXh	successful execution Refer to Table 89

Table 89. Return code description - ISOSelectFile

SW1|SW2	Value	Description
ISO6700	6700h	Wrong or inconsistent APDU length.
ISO6985	6985h	Wrapped chained command or multiple pass command ongoing.
ISO6A82	6A82h	Application or file not found, currently selected application remains selected.
ISO6A86	6A86h	Wrong parameter P1 and/or P2
ISO6A87	6A87h	Wrong parameter Lc inconsistent with P1-P2
ISO6E00	6E00h	Wrong CLA

11.10.2 ISOReadBinary

The ISOReadBinary is a standard ISO/IEC 7816-4 command. It can be used to read data from the Standard Data File. This command does not support any secure messaging, it is always in plain. For executing ISOReadBinary command either "Read" or "Read&Write", access right must be set to free access rights.

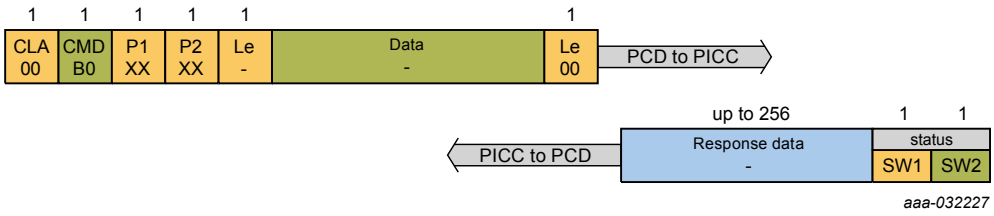


Figure 30. ISOReadBinary command protocol

Table 90. Command parameters description - ISOReadBinary

Name	Length	Value	Description
CLA	1	00h	
INS	1	B0h	
P1	1		ShortFile ID/Offset
	Bit 7		Encoding
		1b	P1[Bit 6..5] are RFU. P1[Bit 4..0] encode a short ISO FileID. P2[Bit 7..0] encode an offset from zero to 255.
		0b	P1 - P2 (15 bits) encode an offset from zero to 32767.
	Bit 6-5	00b	[if P1[7] == 1b] RFU
	Bit 4-0		[if P1[7] == 1b] short ISO FileID
		00h	Targeting currently selected file.
		01h .. 1Eh	Targeting and selecting file referenced by the given short ISO FileID.
		1Fh	RFU
	Bit 6-0	(see P2)	[if P1[7] == 0b] Most significant bits of Offset
P2	1	000000h .. (FileSize - 1)	Offset (see above)
Le	1	-	The number of bytes to be read from the file. The length of a secure messaging MAC (depending on communication mode settings) should be included in this value.
		00h	Read the entire StandardData file, starting from the position specified in the offset value. Note that the short length Le limits response data to 256 byte.
		01h .. FFh	If bigger than (FileSize - Offset), after subtracting MAC length if MAC is to be returned, the entire StandardData file starting from the offset position is returned.

Name	Length	Value	Description
		Full range	

Table 91. Response data parameters description - ISOReadBinary

Name	Length	Value	Description
Data	up to 256	-	Data read.
SW1SW2	2	9000h XXXXh	successful execution Refer to Table 92

Table 92. Return code description - ISOReadBinary

SW1|SW2	Value	Description
ISO6581	6581h	Memory failure
ISO6700	6700h	Wrong or inconsistent APDU length.
ISO6982	6982h	Security status not satisfied: no access allowed as Read and ReadWrite access rights are different from Eh and SDMFileRead (if SDM enabled) access right is set to Fh.
		Security status not satisfied: SDMReadCtr overflow.
		Security status not satisfied: Targeted file cannot be read in not authenticated state as the related SDMReadCtr is equal or bigger than its SDMReadCtrLimit.
		Security status not satisfied: AuthenticatedEV2 not allowed.
		Security status not satisfied: AuthenticatedLRP not allowed.
ISO6985	6985h	Wrapped chained command or multiple pass command ongoing. No file selected. Targeted file is not of StandardData. Application of targeted file holds a TransactionMAC file.
ISO6A82	6A82h	File not found
ISO6A86	6A86h	Wrong parameter P1 and/or P2
ISO6E00	6E00h	Wrong CLA

11.10.3 ISOUpdateBinary

The ISOUpdateBinary command is implemented in compliance with ISO/IEC 7816-4, the command is only possible with CommMode.Plain for a Standard Data File. NT4H2421Tx supports tearing protection for data that is sent within one communication frame to the file. Consequently, when using ISO/IEC 14443-4 chaining to write to a Standard Data File, each frame itself is tearing protected but an interrupted chaining can lead to inconsistent files. Using single-frame ISOUpdateBinary commands instead of using the chaining can enable better control of the overall write process.

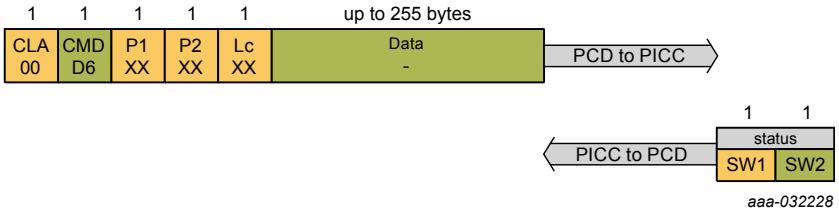


Figure 31. ISOUpdateBinary command protocol

Table 93. Command parameters description - ISOUpdateBinary

Name	Length	Value	Description
CLA	1	00h	
INS	1	D6h	
P1	1		ShortFile ID/Offset
	Bit 7		RFU
		1b	P1[Bit 6..5] are RFU. P1[Bit 4..0] encode a short ISO FileID. P2[Bit 7..0] encode an offset from zero to 255.
		0b	P1 - P2 (15 bits) encode an offset from zero to 32767.
	Bit 6-5	00b	[if P1[7] == 1b] RFU
	Bit 4-0		[if P1[7] == 1b] short ISO FileID
		00h	Targeting currently selected file.
		01h .. 1Eh	Targeting and selecting file referenced by the given short ISO FileID.
		1Fh	RFU
	Bit 6-0	(see P2)	[if P1[7] == 0b] Most significant bits of Offset
P2	1	000000h .. (FileSize - 1)	Offset (see above)
Lc	1	01h .. (FileSize - Offset)	Length of subsequent data field
Data	up to 255 byte	Full range	Data to be written

Table 94. Response data parameters description - ISOUpdateBinary

Name	Length	Value	Description
No response data parameters defined for this command			

Name	Length	Value	Description
SW1SW2	2	9000h XXXXh	successful execution Refer to Table 95

Table 95. Return code description - ISOUpdateBinary

SW1|SW2	Value	Description
ISO6581	6581h	Memory failure
ISO6700	6700h	Wrong or inconsistent APDU length.
ISO6982	6982h	Security status not satisfied: only free write with Write or ReadWrite equal to Eh is allowed. Security status not satisfied: AuthenticatedEV2 not allowed. Security status not satisfied: AuthenticatedLRP not allowed.
ISO6985	6985h	Wrapped chained command or multiple pass command ongoing. No file selected. Attempt to write beyond the file boundary as set during creation.
ISO6A82	6A82h	File not found
ISO6A86	6A86h	Wrong parameter P1 and/or P2
ISO6E00	6E00h	Wrong CLA

11.11 Originality check commands

The originality check allows verification of the genuineness of NTAG 424 DNA TT (NT4H2421Tx). Two ways are offered to check the originality of the PICC: the first is based on a symmetric authentication, the second works on the verification of an asymmetric signature retrieved from the card.

The authentication procedure for AES keys can be used to authenticate to one of the four OriginalityKey and check whether the PICC is a genuine NXP product. NT4H2421Tx supports targeting the OriginalityKey with the LRP authentication using AES. For details on the authentication command, see Section 9.2. The following variants can be used:

- AuthenticateLRPFirst, see Section 9.2.5
- AuthenticateLRPNonFirst, see Section 9.2.6

The asymmetric originality signature is based on ECC and only requires a public key for the verification, which is done outside the card. The Read_Sig command can be used in both ISO/IEC 14443-3 and ISO/IEC 14443-4 protocols to retrieve the signature. If the PICC is not configured for Random ID, the command is freely available. There is no authentication required. If the PICC is configured for Random ID, an authentication is required.

11.11.1 Read_Sig

The Read_Sig retrieves the asymmetric originality signature based on an asymmetric cryptographic algorithm Elliptic Curve Cryptography Digital Signature Algorithm (ECDSA), see [12] and can be used in both ISO/IEC 14443-3 and ISO/IEC 14443-4 protocol. The purpose of originality check signature is to protect from mass copying of non NXP originated ICs. The purpose of originality check signature is not to completely prevent HW copy or emulation of individual ICs.

A public key is required for the verification, which is done outside the card. The NXPOriginalitySignature is computed over the UID and written during manufacturing. If the PICC is not configured for Random ID, the command is freely available. There is no authentication required. If the PICC is configured for Random ID, an authentication with any authentication key is required. If there is an active authentication, the command requires encrypted secure messaging.

Remark: The originality function is provided to prove that the IC has been manufactured by NXP Semiconductors.

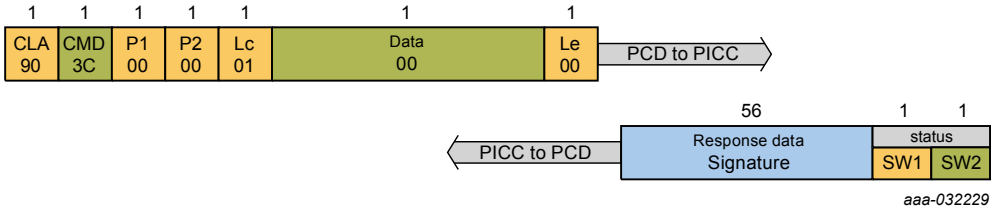


Figure 32. Read_Sig command protocol

Table 96. Command parameters description - Read_Sig

Name	Length	Value	Description
Command Header Parameters			
CMD	1	3Ch	Command code
Command Data Parameters			

Name	Length	Value	Description
Command Header Parameters			
Data	1	-	Targeted ECC originality check signature
		00h	Targeting NXPOriginalitySignature
		01h .. FFh	RFU

Table 97. Response data parameters description - Read_Sig

Name	Length	Value	Description
Data	56	Full range	NXPOriginalitySignature
SW1SW2	2	9100h 91XXh	successful execution Table 98

Table 98. Return code description - Read_Sig

Status	Value	Description
COMMAND_NOT_FOUND	0Bh	Not allowed without valid authentication due to Random ID configuration.
		Missing or incorrect MAC when authenticated.
COMMAND_ABORTED	CAh	Previous Command was not fully completed. Not all Frames were requested or provided by the PCD.
COMMAND_FORMAT_ERROR	0Ch	Unexpected command length.
		Unsupported Address

The NXPOriginalitySignature is computed over the UID with the use of asymmetric cryptographic algorithm Elliptic Curve Cryptography Digital Signature Algorithm (ECDSA), see [12]. No hash is computed: M is directly used as H. The NXP Originality Signature calculation uses curve secp224r1. NXP Originality signature verification together with example is explained in NT4H2421Tx - Feature and Hints application note.

12 Limiting values

Stresses exceeding one or more of the limiting values, can cause permanent damage to the device. Exposure to limiting values for extended periods can affect device reliability.

Table 99. Limiting values

In accordance with the Absolute Maximum Rating System (IEC 60134).

Symbol	Parameter	Conditions		Min	Max	Unit
I_I	input current	on LA/LB		-	40	mA
P_{tot}	total power dissipation			-	120	mW
T_{stg}	storage temperature			-55	125	°C
T_{amb}	ambient temperature			-25	70	°C
V_{ESD}	electrostatic discharge voltage	on LA/LB	[1]	-	2	kV

[1] ANSI/ESDA/JEDEC JS-001; Human body model: C = 100 pF, R = 1.5 kΩ

CAUTION

This device is sensitive to ElectroStatic Discharge (ESD). Observe precautions for handling electrostatic sensitive devices.

Such precautions are described in the *ANSI/ESD S20.20, IEC/ST 61340-5, JESD625-A* or equivalent standards.

13 Characteristics

Table 100. Characteristics

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
C_i	input capacitance		[1]	45.0	50.0	55.0	pF
f_i	input frequency			-	13.56	-	MHz
R_{on}	resistance tag taper closed	DP - GND		-	-	50	Ω
R_{off}	resistance tag tamper open	DP - GND		1 M	-	-	Ω
EEPROM characteristics							
t_{ret}	retention time	T_{amb} = 22 °C		20	-	-	year
$N_{endu(W)}$	write endurance	T_{amb} = 22 °C		200.000	-	-	cycle

[1] T_{amb} = 22 °C, f = 13.56 MHz, V_{LaLb} = 2 V RMS

14 Abbreviations

Table 101. Abbreviations

Acronym	Description
AES	Advanced Encryption Standard
AID	Application Identifier
APDU	Application Protocol Data Unit
AppKey	Application Key
AppMasterKey	Application Master Key
ASCII	American Standard Code for Information Interchange
ATQA	Answer to Request A
ATS	Answer to Select
C-APDU	Command APDU
CBC	Cipher Block Chain, one mode of operation for block ciphers
CC	Capability Container
CID	Channel Identifier
CLA	Class
CMAC	Cipher-based Message Authentication Code
CmdCtr	Command Counter
DF	Dedicated File (Application)
DP	Detection Pin
EAL	Evaluation Assurance Level
ECC	Elliptic Curve Cryptography
EF	Elementary File (File)
FCI	File Control Information
FFC	Film Frame Carrier
FID	File Identifier
FSC	Frame Size for proximity Card
FWI	Frame Waiting Time Integer
INS	Instructions
IV	Initialization Vector
LRP	Leakage Resilient Primitive
LSB	Least Significant Byte
MAC	Message Authentication Code
MF	Master File (PICC Level)
MSB	Most Significant Byte
NDEF	NFC Data Exchange Format
NFC	Near Field Communication

Acronym	Description
NVM	Non-Volatile Memory
PCD	Proximity Coupling Device (Contactless Reader)
PCDCap	Proximity Coupling Device Capabilities
PD	Proximity Device, used as synonym for the PICC
PDCap	Proximity Device Capabilities
PICC	Proximity IC Card
PICCData	PICC data targeted for mirroring (e.g. UID, SDMReadCtr)
PPS	Protocol Parameter Select
RC	Return Code
R-APDU	Response APDU
RFU	Reserved for future use
RID	Random ID
SAK	Select Acknowledge
SAM	Secure Access Module
SDM	Secure Dynamic Messaging
SDMCtrRet	SDM Counter Retrieval, access right for GetFileCounters
SDMENCFileData	Refers to the encrypted part of data in the NDEF file
SDMFileRead	SDM File Reading, key/access setting for Secure Dynamic Messaging
SDMFileReadKey	Refers to the AppKey which is used for SDM MAC calculation
SDMMAC	Refers to the MAC calculated over response
SDMMetaRead	SDM Meta Reading, specifies PICCData encryption key or plain mirroring
SDMMetaReadKey	Refers to the AppKey which is used for SDM encryption of PICCData
SDMReadCtr	SDM Read Counter, counting number of interactions with a PICC
SesAuthENCKey	Session key for encryption
SesAuthMACKey	Session key for MACing
SesTMENCKey	Transaction MAC Session Key for Encryption
SesTMMACKey	Transaction MAC Session Key for MACing
SM	Secure Messaging
SUN	Secure Unique NFC
SV	Session Vector, input for session key calculation
SW	Status Word
TI	Transaction Identifier
TotFailCtr	Total Failed Authentication Counter
TotFailCtrDecr	Total Failed Authentication Counter Decrement
TotFailCtrLimit	Total Failed Authentication Counter Limit
TT	Tag Tamper

Acronym	Description
TTCurrStatus	Current status of the Tag Tamper loop
TTPermStatus	Permanently stores an Open status on the Tag Tamper loop
TTStatusKey	Refers to an AppKey, authentication with this key is required to issue the GetTTStatus command
UID	Unique IDentifier

15 References

[1] **ISO/IEC 14443-2:2016**
Identification cards -- Contactless integrated circuit cards -- Proximity cards -- Part 2: Radio frequency power and signal interface

[2] **ISO/IEC 14443-3:2016**
Identification cards -- Contactless integrated circuit cards -- Proximity cards -- Part 3: Initialization and anti-collision

[3] **ISO/IEC 14443-4:2016**
Identification cards -- Contactless integrated circuit cards -- Proximity cards -- Part 4: Transmission protocol

[4] **ISO/IEC 7816-4:2005**
Identification cards – Integrated circuit cards – Part 4: Organization, security and commands for interchange

[5] **NIST Special Publication 800-38A**
National Institute of Standards and Technology (NIST). Recommendation for BlockCipher Modes of Operation.
http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf

[6] **NIST Special Publication 800-38B**
National Institute of Standards and Technology (NIST). Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication.
http://csrc.nist.gov/publications/nistpubs/800-38B/SP_800-38B.pdf

[7] **ISO/IEC 9797-1:1999**
Information technology – Security techniques – Message Authentication Codes (MACs) – Part 1: Mechanisms using a block cipher.

[8] **NIST Special Publication 800-108**
National Institute of Standards and Technology (NIST). Recommendation for key derivation using pseudorandom functions.

[9] **IEEE Std 802.3-2008**
IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements Part 3: Carrier sense multiple access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications.

[10] **LRP**
Leakage Resilient Primitive (LRP) Specification, Document number 4660**[1]

[11] **Data sheet addendum**
NT4H2421Tx - Wafer specification, Document number 4658**[1]

[12] **Certicom Research. Sec 1**
Elliptic curve cryptography. Version 2.0, May 2009.

1 ** ... document version number

[13] **Product Data Sheet**
NXP Semiconductors, NTAG213/215/216: NFC Forum Type 2 Tag compliant IC with 144/504/888 bytes user memory, Document number 2653** [1]

[14] **NFC Forum: Type 4 Tag - Technical Specification**
NFC Forum: Type 4 Tag - Technical Specification - Version 1.0 - [T4T] - 2016.07.26, 07 2016.

[15] **NFC Data Exchange Format (NDEF)**
NFC Forum - Technical Specification - Version 1.0 - 24.07.2006

16 Revision history

Table 102. Revision history NT4H2421Tx

Document ID	Release date	Data sheet status	Change notice	Supersedes
465530	20190131	Product data sheet	-	465511
Modifications:	• Data sheet status changed into Product data sheet • Security status changed into "Company public"			
465511	20181105	Objective data sheet		465510
Modifications:	• added *response codes* in Status word • changed data retention time in Section 5 and Section 13 • changed ATS value in Section 8.1.1 • clarified generation of *SDMMetaReadUpdateKey* in Section 9.3.4.2			
465510	20180321	Objective data sheet	-	-

17 Legal information

17.1 Data sheet status

Document status[1][2]	Product status[3]	Definition
Objective [short] data sheet	Development	This document contains data from the objective specification for product development.
Preliminary [short] data sheet	Qualification	This document contains data from the preliminary specification.
Product [short] data sheet	Production	This document contains the product specification.

[1] Please consult the most recently issued document before initiating or completing a design.
[2] The term 'short data sheet' is explained in section "Definitions".
[3] The product status of device(s) described in this document may have changed since this document was published and may differ in case of multiple devices. The latest product status information is available on the Internet at URL http://www.nxp.com.

17.2 Definitions

Draft — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

Short data sheet — A short data sheet is an extract from a full data sheet with the same product type number(s) and title. A short data sheet is intended for quick reference only and should not be relied upon to contain detailed and full information. For detailed and full information see the relevant full data sheet, which is available on request via the local NXP Semiconductors sales office. In case of any inconsistency or conflict with the short data sheet, the full data sheet shall prevail.

Product specification — The information and data provided in a Product data sheet shall define the specification of the product as agreed between NXP Semiconductors and its customer, unless NXP Semiconductors and customer have explicitly agreed otherwise in writing. In no event however, shall an agreement be valid in which the NXP Semiconductors product is deemed to offer functions and qualities beyond those described in the Product data sheet.

17.3 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors. In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory. Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification. Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products. NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Limiting values — Stress above one or more limiting values (as defined in the Absolute Maximum Ratings System of IEC 60134) will cause permanent damage to the device. Limiting values are stress ratings only and (proper) operation of the device at these or any other conditions above those given in the Recommended operating conditions section (if present) or the Characteristics sections of this document is not warranted. Constant or repeated exposure to limiting values will permanently and irreversibly affect the quality and reliability of the device.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at http://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

No offer to sell or license — Nothing in this document may be interpreted or construed as an offer to sell products that is open for acceptance or the grant, conveyance or implication of any license under any copyrights, patents or other industrial or intellectual property rights.

Quick reference data — The Quick reference data is an extract of the product data given in the Limiting values and Characteristics sections of this document, and as such is not complete, exhaustive or legally binding.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Non-automotive qualified products — Unless this data sheet expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications. In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

Translations — A non-English (translated) version of a document is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

17.4 Licenses

ICs with DPA Countermeasures functionality

LICENSED DPA COUNTERMEASURES™

NXP ICs containing functionality implementing countermeasures to Differential Power Analysis and Simple Power Analysis are produced and sold under applicable license from Cryptography Research, Inc.

17.5 Trademarks

Notice: All referenced brands, product names, service names and trademarks are the property of their respective owners.

MIFARE — is a trademark of NXP B.V.

DESFire — is a trademark of NXP B.V.

NTAG — is a trademark of NXP B.V.

Tables

Figures

Contents

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 31 January 2019
Document identifier: 465530
Document number: 465530

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

NXP:

NT4H2421TSDUD/020Z NT4H2421TTDUD/02Z NT4H2421TTDUD/02V NT4H2421TTDUF/02V