# AN12696

## MIFARE SAM AV3 - For MIFARE DESFire

Rev. 1.3 — 14 July 2020 Application note
521513 COMPANY PUBLIC

**Document information**

| Information | Content |
|---|---|
| Keywords | MIFARE SAM AV3, MF4SAM3, TDEA, AES, RSA, MIFARE DESFire EV2, MIFARE DESFire EV3 |
| Abstract | This application note presents some examples of using MIFARE SAM AV3 for MIFARE DESFire in S-mode. |

# Revision history

**Revision history**

| Rev | Date | Description |
|---|---|---|
| 1.3 | 20200714 | Added full EV2 Secure Messaging example |
| 1.2 | 20200512 | MIFARE DESFire EV3 included |
| 1.1 | 20200108 | AN number changed, security status changed into “Company Public” |
| 1.0 | 20190807 | Initial version |

# 1 Introduction

MIFARE SAMs (**S**ecure **A**pplication **M**odule) have been designed to provide the secure storage of cryptographic keys and cryptographic functions for the terminals to access the MIFARE products[1] securely and to enable secure communication between terminals and host (backend).

## 1.1 Scope

This application note presents examples of using MIFARE SAM AV3 (referred to SAM in this document, if not otherwise mentioned) for MIFARE DESFire EV3 and previous versions. In this document, the SAM is used in S-mode (X interface is described in doc nr. 5219xx). There is a set of application note for MIFARE SAM AV3; each of them is addressing specific features. The list of application note is given in [4].

This application note is a supplement document for application development using MIFARE SAM AV3. Should there be any confusion please check MIFARE SAM AV3 data sheet [1]. Best use of this application note will be achieved by reading this specification [1] in advance.

**Note: This application note does not replace any of the relevant data sheets, application notes or design guides.**

In this document, the term „MIFARE DESFire card“ refers to a MIFARE DESFire IC-based contactless card.

**All examples in this document are relevant for all MIFARE DESFire EV3, MIFARE DESFire EV2 and MIFARE DESFire EV1 products, if not explicitly stated otherwise!**

## 1.2 Abbreviation

Refer to Application note “MIFARE SAM AV3 – Quick Start up Guide” [4].

## 1.3 Examples presented in this document

The following symbols have been used to mention the operations in the examples:

= Preparation of data by SAM, PICC or host.

> Data sent by the host to SAM or PICC (if not mentioned, SAM).

< Data Response from SAM or PICC (if not mentioned, SAM).

**C-APDU:**

| CLA | INS | P1 | P2 | Lc | Data (nc) | Le |
|---|---|---|---|---|---|---|

**R-APDU:**

| Response data | SW1 | SW2 |
|---|---|---|

**Please note, that the numerical data are used solely as examples. They appear in the text in order to clarify the commands and command data.**

1 . MIFARE Ultralight C, MIFARE Classic, MIFARE Classic EV1, MIFARE Plus, MIFARE Plus EV1, MIFARE Plus EV2, MIFARE DESFire EV1, MIFARE DESFire EV2, MIFARE DESFire EV3

Any data, values, cryptograms are expressed as hex string format if not otherwise mentioned, e.g., 0x563412 in hex string format represented as “123456”. Byte [0] = 0x12, Byte [1] = 0x34, Byte [2] = 0x56.

## 1.4 S interface

The host is managing the communication to SAM and MIFARE DESFire EV2.

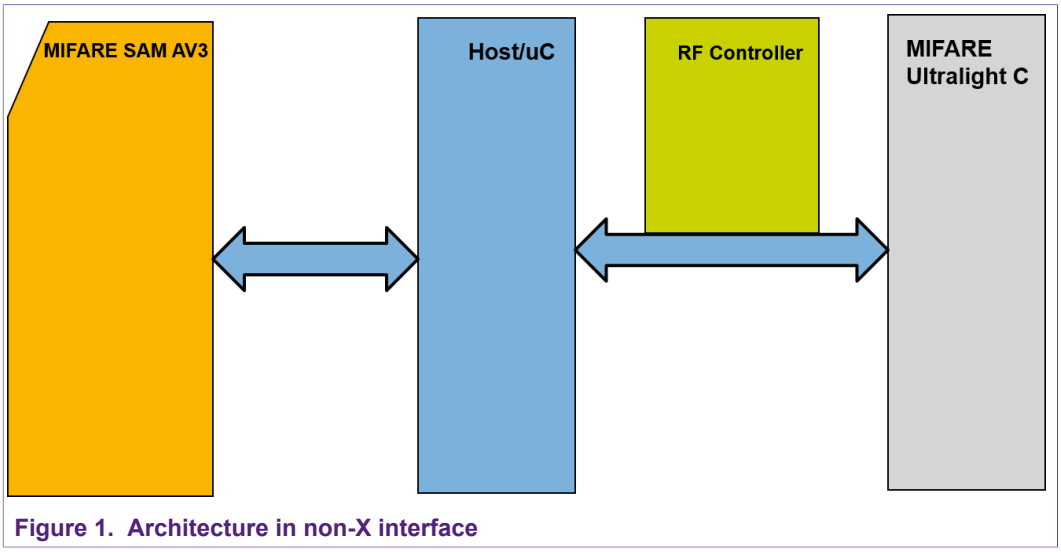



**Figure 1. Architecture in non-X interface**

# 2 Using MIFARE SAM AV3 for MIFARE DESFire

MIFARE SAM AV3 can be used to perform all the crypto and security features offered by MIFARE DESFire EV3, MIFARE DESFire EV2 and MIFARE DESFire EV1. **If not otherwise stated, the examples are valid for all MIFARE DESFire versions.**

## 2.1 Downloading the MIFARE DESFire keys to SAM from Host

Downloading of different keys is explained in [5]. The SAM key entry settings are different for different types of crypto used in MIFARE DESFire. The incorrect setting will result to authentication error. In the following table different options are shown:

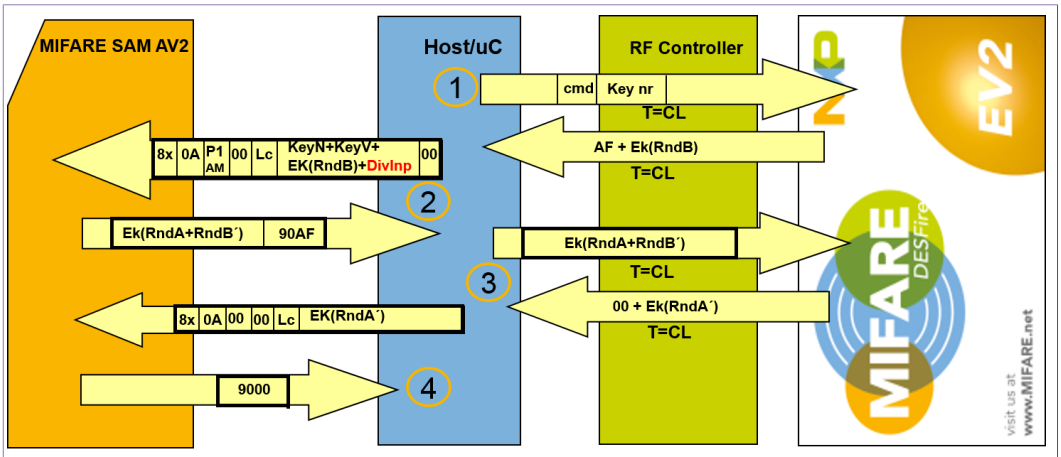
**Table 1. SAM Key Entry setting for different MIFARE DESFire Keys**

| SAM Key entry setting | MIFARE DESFire Native TDEA key | | MIFARE DESFire std. TDEA key | | MIFARE DESFire std. 3KTDEA key | | MIFARE DESFire std. AES key | |
|---|---|---|---|---|---|---|---|---|
| Standard setting “SET” | | | | | | | | |
| b0: Allow dumping session key. | Up to the application ‘0’/ ‘1’. | | Up to the application ‘0’/ ‘1’. | | Up to the application ‘0’/ ‘1’. | | Up to the application ‘0’/ ‘1’. | |
| b1: RFU must be set to 0. | ‘0’ | | ‘0’ | | ‘0’ | | ‘0’ | |
| b2: Keep IV | ‘0’ | | ‘1’ | | ‘1’ | | ‘1’ | |
| b6b5b4b3: Key type | TDEA DESFire 4 (‘000’) | | TDEA ISO 10116 (32-bit CRC, 8-byte MAC) (‘110’) | | 3TDEA ISO 10116 (‘011’) | | AES 128 (‘100’) | |
| b7: PL Key | ‘0’ | | ‘0’ | | ‘0’ | | ‘0’ | |
| b8: Host Auth Key for unlocking the LC | ‘0’ | | ‘0’ | | ‘0’ | | ‘0’ | |
| b9: Disable key entry | ‘0’ | | ‘0’ | | ‘0’ | | ‘0’ | |
| b10: Lock Key | ‘0’ | | ‘0’ | | ‘0’ | | ‘0’ | |
| b11: Disable SAM_ ChangeKeyPICC | ‘0’ in card personalization / issuing machine SAMs. | ‘1’ in check in/out terminal SAMs | ‘0’ in card personalization / issuing machine SAMs. | ‘1’ in check in/out terminal SAMs | ‘0’ in card personalization / issuing machine SAMs. | ‘1’ in check in/out terminal SAMs | ‘0’ in card personalization / issuing machine SAMs. | ‘1’ in check in/out terminal SAMs |
| b15b14b13b12 | ‘0000’ | | ‘0000’ | | ‘0000’ | | ‘0000’ | |
| Extended setting “ExtSET” | | | | | | | | |
| b2b1b0: Key class | ‘001’ PICC Key | | ‘001’ PICC Key | | ‘001’ PICC Key | | ‘001’ PICC Key | |
| b3: Allow dumping secret key. Not recommended to set. | ‘0’ | | ‘0’ | | ‘0’ | | ‘0’ | |

| SAM Key entry setting | MIFARE DESFire Native TDEA key | MIFARE DESFire std. TDEA key | MIFARE DESFire std. 3KTDEA key | MIFARE DESFire std. AES key |
|---|---|---|---|---|
| b4: Restricted for diversification. Strongly recommended to use. | '1' | '1' | '1' | '1' |
| b15b14b13b12b11 b10b9b8b7b6b5 | '00000000000' | '00000000000' | '00000000000' | '00000000000' |

## 2.2 Authenticating MIFARE DESFire using the SAM

The full authentication is managed by host microcontroller.




Only possible with DESFire key types of SAM key entry

1. Host/uC sends the authentication command to MIFARE DESFire, MIFARE DESFire returns the challenge.
2. Host/uC sends the challenge to the SAM. SAM prepares the response of the MIFARE DESFire challenge and generates its own challenge.
3. Host/uC sends the response of SAM to MIFARE DESFire. MIFARE DESFire returns the response of the SAM challenge.
4. Host/uC sends the MIFARE DESFire response to the SAM for verification. SAM checks it and finally returns the status of the authentication.

**Figure 2. MIFARE DESFire EV2 Authentication using SAM**

### 2.2.1 MIFARE DESFire Authentication, key type AES-128 (non-diversified key)

**This example applies for all MIFARE DESFire types that support EV1 secure messaging (MIFARE DESFire EV1 and above)**

In this example key entry number 1 will be used, which has the following attributes

Key Version A = 01

Key Version B = 02

Key Version C = 03

DF_AID = AE0102

DF_KeyNo = 03

KeyNoCEK = 00

KeyVCEK = 00

RefNoKUC = FF

SET = 2400

DO NOT allow dump Session key

DO NOT allow crypto with secret key

Keep IV

Key type: AES 128

ExtSET = 01 (Diversification is not mandatory).

In this example, we use the Key Version “01” for authentication. The reference is to the SAM key entry number instead of DESFire key number.

**Table 2. MIFARE DESFire EV2 AES Authentication (Non-Diversified key)**

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | Send authentication command to DESFire | > | AA03 | Authentication cmd AES and DESFire application key number. |
| 2 | Challenge from DESFire | < | AF90F3859EA795A43F3A32144BAC2B9856 | AF is the status byte and 16-byte Ek(RndB). |
| 3 | First part of the SAM_AuthenticatePICC command | > | 800A000012010190F3859EA795A43F3A32144BAC2B985600 | P1 = 00; no key diversification, key selection is by key entry number. Data field is SAM key entry number, version number and Ek(RndB received in step 2). |
| 4 | Answer of the SAM | < | 025CE60F614B27BEAFA60FDF733E65F7ED6F4A4A8B51B625D1C4ADC43BF0294D90AF | Ek(RndA+RndB´) and status byte 90AF. |
| 5 | Answer of the SAM is sent to DESFire | > | AF025CE60F614B27BEAFA60FDF733E65F7ED6F4A4A8B51B625D1C4ADC43BF0294D | AF is the cmd and Ek(RndA +RndB´). |
| 6 | Response of the DESFire | < | 009B9080079A49F85FEF72C800264BA4DC | 00 is the status means that authentication is successful and Ek(RndA´). |
| 7 | Second part of the SAM_AuthenticatePICC command | > | 800A0000109B9080079A49F85FEF72C800264BA4DC | Ek(RndA´) is sent to the SAM |
| 8 | Answer of the SAM | < | 9000 | SAM decides if the DESFire response is correct or not. |

If the reference is to be made to the DESFire key number in step 3, Key selection by DESFire key number has to be chosen in P1. In case the DESFire Application (DF_AID = AE0102) has to be selected in SAM using SAM_SelectApplication before step 1. The Data field will then contain the DESFire key number (03) (see the next example) here instead of SAM key entry number 01.

### 2.2.2 MIFARE DESFire Authentication, key type AES-128 (diversified key)

**This example applies for all MIFARE DESFire types that support EV1 secure messaging (MIFARE DESFire EV1 and above)**

In this example key entry number 3 will be used, which has the following attributes

Key Version A = 01

Key Version B = 02

Key Version C = 03

DF_AID = AE0102

DF_KeyNo = 03

KeyNoCEK = 00

KeyVCEK = 00

RefNoKUC = FF

SET = 2400

DO NOT allow dump Session key

DO NOT allow crypto with secret key

Keep IV

Key type: AES 128

ExtSET = 01 (Diversification is not mandatory).

In this example, we use the Key Version “01” for authentication. The reference is to the DESFire key number instead of SAM key entry number.

**Table 3. MIFARE DESFire EV1 AES Authentication (Diversified key)**

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | Send authentication command to DESFire | > | AA03 | Authentication cmd AES and DESFire application key number. |
| 2 | Challenge from DESFire | < | AFFEC675C728FCDED6C4822FFE00A3E6F6 | AF is the status byte and 16-byte Ek(RndB). |
| 3 | First part of the SAM_ AuthenticatePICC command | > | 800A0300220301FEC675C728FCDED6C4822FFE00A3E6F68804084561801D808804084561801D8000 | P1 = 03; key diversification, key selection is by DESFire key number. Data field is DESFire key number, version number, Ek(RndB received in step 2) and diversification input (8804084561801D808804084561801D80). |

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 4 | Answer of the SAM | < | 14CD4EC79AAC3AF1AEF0 C2F4241E37C6723520FF28 CDF17BBDC9798EF59FAB DC90AF | Ek(RndA+RndB´) and status byte 90AF. |
| 5 | Answer of the SAM is sent to DESFire | > | AF14CD4EC79AAC3AF1AE F0C2F4241E37C6723520FF 28CDF17BBDC9798EF59FA BDC | AF is the cmd and Ek(RndA +RndB´). |
| 6 | Response of the DESFire | < | 0036E87F56560FA0202F6D 33EA94DA65C7 | 00 is the status means that authentication is successful and Ek(RndA´). |
| 7 | Second part of the SAM_ AuthenticatePICC command | > | 800A00001036E87F56560F A0202F6D33EA94DA65C7 | Ek(RndA´) is sent to the SAM |
| 8 | Answer of the SAM | < | 9000 | SAM decides if the DESFire response is correct or not. |

### 2.2.3 MIFARE DESFire Authentication, key type 3KTDES (non-diversified key)

**This example applies for all MIFARE DESFire types that support EV1 secure messaging (MIFARE DESFire EV1 and above)**

In this example key entry number 2 will be used, which has the following attributes

Key Version A = 05

Key Version B = 06

DF_AID = 3D0102

DF_KeyNo = 04

KeyNoCEK = 00

KeyVCEK = 00

RefNoKUC = FF

SET = 1C00

DO NOT allow dump Session key

DO NOT allow crypto with secret key

Keep IV

Key type: 3KTDES ISO 10116

ExtSET = 01 (Diversification is not mandatory).

In this example, we use the Key Version “05” for authentication. The reference is to the SAM key entry number instead of DESFire key number.

Table 4. MIFARE DESFire EV2 3KTDES Authentication (Non-Diversified key)

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | Send authentication command to DESFire | > | 1A04 | Authentication cmd “standard 3DES” and DESFire application key number. |
| 2 | Challenge from DESFire | < | AFA0B9517BACC35E5B7A6 AAEDA18116B5E | AF is the status byte and 16-byte Ek(RndB). |
| 3 | First part of the SAM_ AuthenticatePICC command | > | 800A0000120205A0B9517B ACC35E5B7A6AAEDA18116 B5E00 | P1 = 00; no key diversification, key selection is by key entry number. Data field is SAM key entry number, version number and Ek(RndB received in step 2). |
| 4 | Answer of the SAM | < | 7B89FB9EFDE0EEA79B09F CA72F40D291640E08DBD5 6374023A7D1D626C8AF261 90AF | Ek(RndA+RndB´) and status byte 90AF. |
| 5 | Answer of the SAM is sent to DESFire | > | AF7B89FB9EFDE0EEA79B0 9FCA72F40D291640E08DB D56374023A7D1D626C8AF 261 | AF is the cmd and Ek(RndA +RndB´). |
| 6 | Response of the DESFire | < | 00C4580DB85C8F8F0CA7E 981671AA0C12C | 00 is the status means that authentication is successful and Ek(RndA´). |
| 7 | Second part of the SAM_ AuthenticatePICC command | > | 800A000010C4580DB85C8F 8F0CA7E981671AA0C12C | Ek(RndA´) is sent to the SAM |
| 8 | Answer of the SAM | < | 9000 | SAM decides if the DESFire response is correct or not. |

### 2.2.4 MIFARE DESFire Authentication - AuthenticateFirst

In this example, the MIFARE DESFire AuthenticateFirst command will be used. **It is exclusive to MIFARE DESFire EV2 and MIFARE DESFire EV3**.

Key Version A = 01

Key Version B = 02

Key Version C = 03

DF_AID = AE0102

DF_KeyNo = 03

KeyNoCEK = 00

KeyVCEK = 00

RefNoKUC = FF

SET = 2400

DO NOT allow dump Session key

DO NOT allow crypto with secret key

Key type: AES 128

ExtSET = 01 (Diversification is not mandatory).

In this example, we use the Key Entry “06” and KeyVersion "01" for authentication. The reference is to the SAM key entry number instead of DESFire key number.

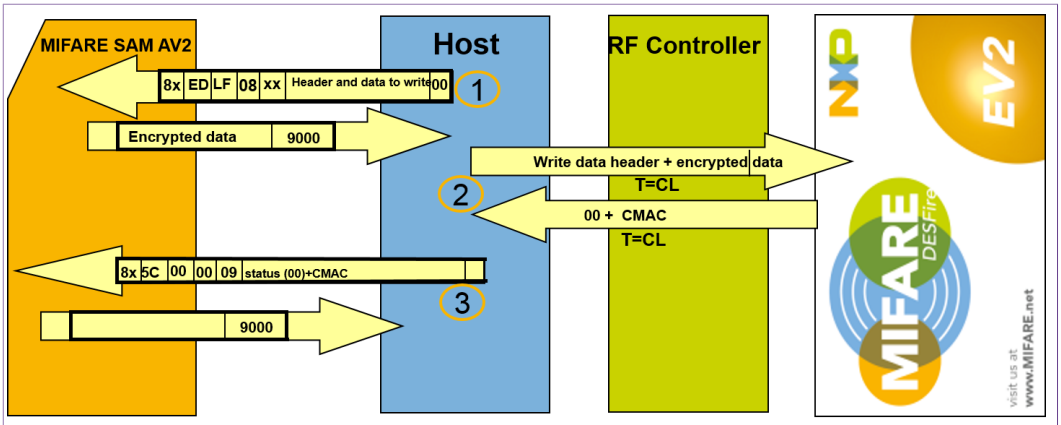
**Table 5. MIFARE DESFire EV2 AES Authentication (Non-Diversified key)**

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | Send authentication command to DESFire | > | 710000 | AuthenticateFirst command |
| 2 | Challenge from DESFire | < | AF3C12D5A104C5E63102D39D60C5D1DBD3 | AF is the status byte and 16-byte PDChal |
| 3 | First part of the SAM_ AuthenticatePICC command | > | 800A8000130601003C12D5A104C5E63102D39D60C5D1DBD300 | P1 = 80; EVx authentication Type, AuthenticateFirst, no key diversification, key selection is by key entry number. Data field is SAM key entry number, version number and Ek(RndB received in step 2). |
| 4 | Answer of the SAM | < | 43AAF5D75D40352FFD9194DE07026728C36839CC3CA4DF4EEA66DD4DBD10836390AF | PCDChalResp and status byte 90AF. |
| 5 | Answer of the SAM is sent to DESFire | > | AF43AAF5D75D40352FFD9194DE07026728C36839CC3CA4DF4EEA66DD4DBD108363 | AF is the cmd and PCDChalResp. |
| 6 | Response of the DESFire | < | 0029B033E337CD4FD21FC3E0B677D3D06FE62B689B540BC9767A2CFA7C00F09770 | 00 is the status means that authentication is successful and PDResp. |
| 7 | Second part of the SAM_ AuthenticatePICC command | > | 800A00002029B033E337CD4FD21FC3E0B677D3D06FE62B689B540BC9767A2CFA7C00F0977000 | PDResp is sent to the SAM |
| 8 | Answer of the SAM | < | 00000000000000000000000009000 | SAM decides if the DESFire response is correct or not and answers with the PICC Capabilities and 9000 |

For AuthenticateNonFirst, only Bit 6 from P1 of step 3 must be set to 1, resulting in 0xC0 for P1.

## 2.3 Encrypted Write to data file using SAM

All command flow is managed by host microcontroller.




1. Enciphering data to write
2. Sending 'WriteData' Command
3. Verifying CMAC received from DESFire

1. Encipher the data to be written using SAM.
2. Send the encrypted data to DESFire.
3. Verify CMAC returned by DESFire using SAM. For DESFire Native 3DES mode, this step is not required.

**Figure 3. Encrypted write data file using SAM**

### 2.3.1 Encrypted Write to Data file using SAM, 3KTDES mode

DESFire application is authenticated using 3KTDES mode.

File Id = 01;

Offset where to write = 000000;

No of bytes to write = 0B0000 ;( 11 byte to write).

Data = 0102030405060708091011.

File communication type = Encrypted.

**Table 6. Write data file encrypted using SAM, 3KTDES mode**

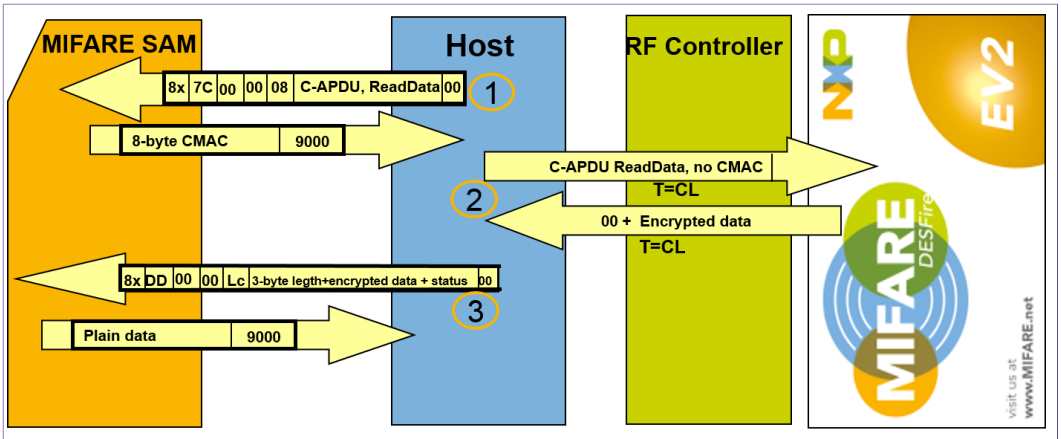
| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | Send data to SAM for encryption | > | 80ED0008133D010000000 B000001020304050607080 9101100 | P2 = 08; means the encryption starts from 8th byte of the data in data field (count from 0th). The header of write data cmd = 3D010000003B0000. |
| 2 | Encrypted data | < | DBC715E2948F68C4BDAC 454B8CE1C4149000 | SAM encrypts the data using the session key. It inserts CRC32 and padding before encryption. |
| 3 | Send write command to DESFire | > | 3D010000000B0000DBC71 5E2948F68C4BDAC454B8 CE1C414 | Header and encrypted data. |
| 4 | DESFire answer | < | 00511652BFFAB9680A | Status byte 00 and CMAC |
| 5 | Verify CMAC using SAM | > | 805C00000900511652BFFA B9680A | P2 = 00, means the whole CMAC will be verified. |

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 6 | SAM confirms | < | 9000 | |

- For AES mode: the write steps are similar.
- For DESFire Native TDES mode: In step 1 the header is not required to add in the data field (C-APDU >80ED00000B0102030405060708091011 00), step 5 and 6 are not required.

## 2.4 Encrypted Read from data file using SAM

All command flow is managed by host microcontroller.




1. Generating CMAC to update the IV
2. Sending 'ReadData' Command
3. Deciphering encrypted data from the SAM

1. Send the Read data command header to SAM to update the IV. For DESFire Native 3DES mode, this step is not required.
2. Send Read data command to DESFire, DESFire replies the encrypted data.
3. Decrypt data using SAM.

**Figure 4. Encrypted Read from a data file using SAM**

### 2.4.1 Encrypted Read from Data file using SAM, 3KTDES mode

DESFire application is authenticated using 3KTDES mode.

File Id = 01;

Offset from where to Read = 000000;

No of bytes to be read = 0B0000 ;( 11 bytes).

File communication type = Encrypted.

**Table 7. Read data file encrypted using SAM, 3KTDES mode**

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | Send data to SAM for generating CMAC | > | 807C000008BD010000000 B000000 | CMAC is generated to update the SAM IV. The read command header is the data field same will be sent to DESFire in step 3. |

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 2 | Generated CMAC from SAM | < | 176B5374BE4DF4699000 | CMAC is not sent to DESFire with read command. |
| 3 | Send read command to DESFire | > | BD010000000B0000 | Read command from file number 1 from offset 000000 and 0B0000 bytes to read. |
| 4 | DESFire answer | < | 006D898674E4655D031CB 76D0C0FAF6F50 | 1-byte status and 16-byte encrypted data. Which includes 11 bytes user data, 4-byte CRC32 and padding? |
| 5 | The encrypted data is sent to SAM for deciphering. | > | 80DD0000140B00006D898 674E4655D031CB76D0C0F AF6F50**00**00 | Status byte is appended in the end of the encrypted data in the data filed. |
| 6 | SAM replies the user data | < | 0102030405060708091011 9000 | 0102030405060708091011 – 11-byte user data and status word. |

- In AES mode: the read data command is similar.
- In DESFire native 3DES mode: step 1 and 2, and adding status byte in step 5 are not required.

## 2.5 Write data (C)MAC communication

The commands flow is similar as described in 2.4. In the following, one example is shown.

### 2.5.1 CMACed Write to Data file using SAM, AES mode

DESFire application is authenticated using AES mode.

File Id = 14;

Offset where to write = 0A0000;

No of bytes to write = 0E0000; (14-byte to write).

Data = 112233445566778899aabbccddee.

File communication type = (C)MAC.

**Table 8. Write data file CMACed AES mode**

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | Calculate CMAC from SAM | > | 807C0000163D140A00000 E000011223344556677889 9AABBCCDDEE00 | Data field contains header and data to be written. |
| 2 | CMAC is returned by SAM | < | 9C47E352194D48689000 | CMAC and status word. |
| 3 | Write command is sent to DESFire together with CMAC | > | 3D140A00000E0000112233 445566778899AABBCCDD EE9C47E352194D4868 | |
| 4 | Answer of DESFire | < | 003F16CACD362B14D9 | 1-byte status “00”and CAMC. |

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 5 | The status and CMAC are sent to SAM for verification | > | 805C000009003F16CACD362B14D9 | |
| 6 | SAM confirms | < | 9000 | CMAC received from DESFire is ok. |

- For standard 3DES mode: the write steps are similar.
- For DESFire Native 3DES Mode: in step 1 adding the header, step 5 and step 6 are not required.

## 2.6 Read data (C)MAC communication

The commands flow is similar to that described in 2.4. In the following, one example is provided.

### 2.6.1 MACed Read from Data file using SAM, AES mode

DESFire application is authenticated using AES mode.

File Id = 14;

Offset where to read = 0A0000;

No of bytes to read = 0E0000; (14-byte).

File communication type = (C)MAC.

**Table 9. Read data file CMAC communication AES mode**

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | Read command is sent to SAM for CMAC generation | > | 807C000008BD140A00000E000000 | CMAC is generated to update the init vector, but won't be sent to DESFire. |
| 2 | CMAC from SAM | < | CCC44BC88134433D9000 | 8-byte CMAC and status |
| 3 | Read command is sent to DESFire | > | BD140A00000E0000 | No CMAC only read command with parameters. |
| 4 | Data together with CMAC from DESFire | < | 00112233445566778899AABBCCDDEE75CC1B0B072B5A4E | First byte is the status and last 8-byte is the CAMC. |
| 5 | Data is sent to SAM for CMAC verification | > | 805C000017112233445566778899AABBCCDDEE0075CC1B0B072B5A4E | Status is appended after the data and before the CMAC. |
| 6 | SAM response | < | 9000 | SAM confirms the CMAC. |

- For standard 3DES mode: the read steps are similar.
- For DESFire Native 3DES Mode: step 1, step 2 and adding status byte in step 5 are not required.

## 2.7 Changing MIFARE DESFire Key using MIFARE SAM AV3

MIFARE SAM AV3 supports the functionality of changing all keys in the MIFARE DESFire, card master key (key number 0 at card level) and application keys (key number 0 up to 13 in applications). Before changing the key, MIFARE DESFire card or application must be authenticated using the correct key using SAM.

### 2.7.1 Changing DESFire Card Master Key TDES native to AES

In this example, MIFARE DESFire EV2 card crypto type and key are changed to AES mode from DESFire native mode.

The old key is DESFire native key SET = “0000”, ExtSET = “01”

The new key is AES 128 key SET = “2400”, ExtSET = “01”

**Table 10. SAM_ChangeKeyPICC for changing card master key**

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| | Authentication using SAM with a key entry of DESFire native type and PICC class. | | | |
| 1 | SAM_ ChangeKeyPICC C-APDU to SAM | > | 80C4011004020003 0000 | MIFARE DESFire current PICC master key = SAM key entry nr. 02 and version 00.<br>MIFARE DESFire new PICC = SAM key entry nr. 02 and version 00.<br>b0 of P1 is set as it is case 2, see detail in [9], b4 of P2 is set as the card master key is changed. |
| 2 | SAM_ ChangeKeyPICC R-APDU to SAM | < | DDFBBC0092D4BB4 20931F67829D2386 6E5DE3D982CD0DF C09000 | The cryptogram for sending to the MIFARE DESFire. |
| 3 | Change key command to MIFARE DESFire | > | C480DDFBBC0092D 4BB420931F67829D 23866E5DE3D982C D0DFC0 | The key number is 80, as the new card crypto will be AES. |
| 4 | MIFARE DESFire response | < | 00 | The key has been changed successfully. |

### 2.7.2 Changing DESFire Card Master Key AES to AES

In this example MIFARE DESFire EV2 card master key is changed, the crypto remains same AES.

The current key is AES 128 key SET = “2400”, ExtSET = “01”

The new key is AES 128 key SET = “2400”, ExtSET = “01”

**Table 11. SAM_ChangeKeyPICC for changing card master key**

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| | Authentication using SAM with a key entry of AES type and PICC class. | | | |

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | SAM_ ChangeKeyPICC C-APDU to SAM | > | 80C4011004030003 0100 | MIFARE DESFire current PICC master key = SAM key entry nr. 03 and version 00.<br>MIFARE DESFire new PICC = SAM key entry nr. 03 and version 01.<br>b0 of P1 is set as it is case 2, see detail in [9], b4 of P2 is set as the card master key is changed. |
| 2 | SAM_ ChangeKeyPICC R-APDU to SAM | < | 8C3DDB802614AA1 5E93998FD00303B0 5C8B1D86AC99CE5 06DCDB36D10C1D8 2659000 | The cryptogram for sending to the MIFARE DESFire. |
| 3 | Change key command to MIFARE DESFire | > | C4808C3DDB80261 4AA15E93998FD003 03B05C8B1D86AC9 9CE506DCDB36D10 C1D8265 | The key number is 80, as the card crypto is AES. |
| 4 | MIFARE DESFire response | < | 00 | The key has been changed successfully. |

### 2.7.3 Changing MIFARE DESFire Application Keys

MIFARE DESFire application keys can be changed using MIFARE SAM AV3. In the following, one example is shown.

The current key is AES 128 key SET = “2400”, ExtSET = “01”

The new key is AES 128 key SET = “2400”, ExtSET = “01”

**Application is authenticated using application change-key key using SAM.**

**Table 12. Change application key number 1 using SAM**

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | SAM_ ChangeKeyPICC C-APDU to SAM | > | 80C422000B03000301041D 7461801D8000 | P1 = 22; Application key case 1 see detail in [9]. New key will be diversified using CMAC-based key diversification.<br>DivInp = 041D7461801D80 (7-byte UID of MIFARE DESFire).<br>P2 = 0x01; key number 0x01 will be changed. |
| 2 | SAM_ ChangeKeyPICC R-APDU to SAM | < | FFBE73AAE8E2DBE7A246 395F46DCFAA4E504A0FC C288FA7DAE2A3E5E71A9 05359000 | Cryptogram for changing key |
| 3 | Send change key command to DESFire | > | C401FFBE73AAE8E2DBE7 A246395F46DCFAA4E504A 0FCC288FA7DAE2A3E5E7 1A90535 | DESFire change key command for key number 01. |
| 4 | Response of DESFire card | < | 00DA03ABEA752659CE | Status “00” and CMAC. |

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 5 | Verify CMAC using SAM | > | 805C00000900DA03ABEA752659CE | Status and CMAC are sent to SAM. |
| 6 | SAM response | < | 9000 | CMAC is correct. |

### 2.7.4 Changing MIFARE DESFire Application Master Key

MIFARE DESFire application keys can be changed using MIFARE SAM AV3. In the following, one example is shown.

The current key is AES 128 key SET = “2400”, ExtSET = “01”

The new key is AES 128 key SET = “2400”, ExtSET = “01”

**Application is authenticated using application master key (key nr. 0) using SAM.**

**Table 13. Change application master key using SAM**

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | SAM_ ChangeKeyPICC C-APDU to SAM | > | 80C423000B03000301041D7461801D8000 | P1 = 23; case 2, see detail in [9]. New will be diversified using the CMAC-based key diversification.<br>DivInp = 041D7461801D80 (7-byte UID of MIFARE DESFire).<br>P2 = 0x00; key number 0x00 will be changed. |
| 2 | SAM_ ChangeKeyPICC R-APDU to SAM | < | B28DB23B8259B1A65CF2BD0B940C8FBE4D91BD4A02187F52D132F0E95D3D12159000 | Cryptogram for changing key |
| 3 | Send change key command to DESFire | > | C400B28DB23B8259B1A65CF2BD0B940C8FBE4D91BD4A02187F52D132F0E95D3D1215 | DESFire change key command for key number 00. |
| 4 | Response of DESFire card | < | 00 | Status “00” and CMAC. |

## 2.8 Full Transaction example using EV2 Secure Messaging

Following is an example of a full transaction using EV2 Secure Messaging, demonstrating the use of SAM_Apply_SM / SAM_Remove_SM commands. The transaction looks like the following:

1. AuthenticateFirst (CardKey 0x01)
2. ReadData (CommMode.Full)
3. GetValue (CommMode.Plain)
4. AuthenticateFirst (CardKey 0x02)
5. WriteRecord (CommMode.Full)
6. Debit (CommMode.Full)
7. CommitTranscation

The example assumes to have a card with an application containing a StdDataFile, a RecordFile and a ValueFile. Key number 0x01 is used for read access to those files, key

number 0x02 is used for writing. CardKey number 0x00(Application Master Key) is not used in this example.

The KeyEntry used in SAM AV3 is number 0x02, which is configured as a PICC Key. The plain key values are

- AA00000000000000000000000000000000, version 0x00
- AA33333333333333333333333333333333, version 0x01
- AA66666666666666666666666666666666, version 0x02

The key versions represent the corresponding MIFARE DESFire card key number. No other special setting is needed for this key entry.

### 2.8.1 AuthenticateFirst with CardKey 0x01

This example shows the second authentication with CardKey Number **0x01** is used, which corresponds to SAM AV3 KeyEntry 0x02 Version **0x01**. Still, AuthenticateFirst command is used.

**Table 14. Example**

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | Send AuthenticateFirst to PICC | > | 710100 | AuthenticateFirst with KeyNumber 0x01 |
| 2 | Receive the challenge from MIFARE DESFire | < | AFD05950B359032E23D36E53418ECF5D26 | StatusByte (0xAF) + 16 bytes challenge |
| 3 | Send SAM_AuthenticatePICC command | > | 800A800013020100D05950B359032E23D36E53418ECF5D2600 | SAM_AuthenticatePICC, P1 = 0x80 (EVx authentication, no key diversification), P2 = 0x00, data field consists of key number, key version, AuthMode (EV2) and the 16 byte challenge |
| 4 | Receive data to be provided to the PICC | < | 6EFC56347B9EABBAA565EEE45E30B19B36849336C40104F926AC8BBCE0F35CD490AF | SAM response, data field to provide to MIFARE DESFire + status word 90AF |
| 5 | Send data from SAM AV3 to PICC | > | AF6EFC56347B9EABBAA565EEE45E30B19B36849336C40104F926AC8BBCE0F35CD4 | Send data field from SAM to DESFire card |
| 6 | Receive PICC response | < | 0026AA64F56B37D8C925185F3D4CA34785DBF1D6ABE6AE60560B713CE59146A8C1 | Response from DESFire card, status byte 0x00 means success, 32 bytes PICC response for the SAM to verify |
| 7 | Provide PICC response to SAM AV3 | > | 800A00002026AA64F56B37D8C925185F3D4CA34785DBF1D6ABE6AE60560B713CE59146A8C100 | Forward data from DESFire to SAM AV3 using again the SAM_AuthenticatePICC command, but with P1 and P2 = 0x00 |
| 8 | Receive Capabilities and status word | < | 00000000000000000000009000 | SAM AV3 sends back the decrypted capabilities + status word 9000, meaning success |

Now, new session keys are created and the counters are reset.

### 2.8.2 Read data file

After a successful authentication, the standard data file can be read in CommMode.Full (fully enciphered + CMAC). For this purpose, the SAM AV3 provides the SAM_Apply_SM and SAM_Remove_SM commands. Those commands automatically prepare dataframes that can be transmitted to, and decrypt dataframes coming from a MIFARE DESFire PICC, given the CommMode.

DataFile to reader: 0x00

Offset: 0x000000

Length: 0x0F0000 (15 byte)

**Table 15. Example**

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | SAM_Apply_SM | > | 80AE001008BD000000000F000000 | The SAM_Apply_SM command creates the CMAC for the given command to send. The command alone is "BD000000000F000000". As CommMode, given in P2 = 0x10, only CommMode.MAC is used, as there is no data to encrypt. |
| 2 | CMAC of command input | < | C0B3B4A1C4F426D99000 | The SAM replies with the CMAC. |
| 3 | Send ReadData command + CMAC to DESFire | > | BD000000000F0000C0B3B4A1C4F426D9 | Command + CMAC are transmitted to the DESFire PICC. |
| 4 | Encrypted Data + CMAC | < | 0037F632279295788522D9A13720073AF6831EB752ACC51BD0 | Status byte (0x00) + 16 byte encrypted data + 8 Byte CMAC |
| 5 | Send the received cryptogram to SAM AV3 | > | 80AD0030190037F632279295788522D9A13720073AF6831EB752ACC51BD000 | The whole received data including the status byte are transmitted to SAM AV3. The CommMode in P2 is now CommMode.Full (0x30). |
| 6 | Decrypted data | < | 44454D4F496E7374616C6C6174696F9000 | The SAM replies with the decrypted data (In this case, it is the hex representation of the ACSII text "DEMOInstallatio" + the status word 0x9000.) |

### 2.8.3 GetValue

The application on the DESFire in this case allows "Free Get Value", this means, the GetValue command needs to be sent in CommMode.Plain.

The ValueFile is the file number 0x01.

Table 16. Example

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | SAM_Apply_SM | > | 80AE0000010100 | This command is needed to maintain internal transaction counters (R_Ctr, W_Ctr) in the SAM AV3 in sync with the PICC. The data field in the command only contains the data field of the DESFire command (in this case 0x01), P2 =0x00 means CommMode.Plain |
| 2 | Status word | < | 9000 | The status word of the SAM. No data is returned, as the command will be sent in plain |
| 3 | Command sent to DESFire PICC | > | 6C01 | Get ValueCommand sent to DESFire |
| 4 | Response in plain | < | 0009000000 | DESFire Response. The value in the value file is 0x09000000 (4 byte LSB first) |

### 2.8.4 AuthenticateFirst with CardKey 0x02

This example shows the second authentication with CardKey Number **0x02** is used, which corresponds to SAM AV3 KeyEntry 0x02 Version **0x02**. Still, AuthenticateFirst command is used.

Table 17. Example

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | Send AuthenticateFirst to PICC | > | 710200 | AuthenticateFirst with KeyNumber 0x02 |
| 2 | Receive the challenge from MIFARE DESFire | < | AF171CDB5D9309280691B01E48243A6AC2 | StatusByte (0xAF) + 16 bytes challenge |
| 3 | Send SAM_ AuthenticatePICC command | > | 800A800013020200171CDB5D9309280691B01E48243A6AC200 | SAM_AuthenticatePICC, P1 = 0x80 (EVx authentication, no key diversification), P2 = 0x00, data field consists of key number, key version, AuthMode (EV2) and the 16 byte challenge |
| 4 | Receive data to be provided to the PICC | < | AE63C37DEEB1A7DAF206EBD1597D10E4C2C0F3BA40892839C3E535DD88BA07E490AF | SAM response, data field to provide to MIFARE DESFire + status word 90AF |
| 5 | Send data from SAM AV3 to PICC | > | AFAE63C37DEEB1A7DAF206EBD1597D10E4C2C0F3BA40892839C3E535DD88BA07E4 | Send data field from SAM to DESFire card |
| 6 | Receive PICC response | < | 008CCAC2A7D5339AE25EF8907EBFDF58422FF0B5B898328C7ED2F0BE1C309909FD | Response from DESFire card, status byte 0x00 means success, 32 bytes PICC response for the SAM to verify |

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 7 | Provide PICC response to SAM AV3 | > | 800A0000208CCAC2A7D5339AE25EF8907EBFDF58422FF0B5B898328C7ED2F0BE1C309909FD00 | Forward data from DESFire to SAM AV3 using again the SAM_AuthenticatePICC command, but with P1 and P2 = 0x00 |
| 8 | Receive Capabilities and status word | < | 00000000000000000000009000 | SAM AV3 sends back the decrypted capabilities + status word 9000, meaning success |

Now, new session keys are created and the counters are reset.

### 2.8.5 WriteRecord

A new record is added to the RecordFile on the MIFARE DESFire PICC.

FileNumber: 0x02

Record size: 32 byte

CommMode.Full

RecordData:
00112233445566778899AABBCCDDEEFF00112233445566778899AABBCCDDEEFF

**Table 18. Example**

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | SAM_Apply_SM | > | 80AE003029083B02000000200000011223344556677 8899AABBCCDDEEFF0011 2233445566778899AABBC CDDEEFF00 | SAM_Apply_SM command, taking the whole DESFire command in plain, with an offset byte in front. The offset is the number of bytes belonging to the DESFire command header (here: 0x08 [command code + file number + 3 byte file offset + 3 byte length] ) |
| 2 | Frame to be sent to DESFire PICC | < | 2F2F06EA824B39CA2B1A38FD284EE8996AF35FC9714AD09CD1AB21D5D72C2DA3EFF1D0EFB2ECC94F60F2CF5ADE35E55D3ED2733DBA9CC5E99000 | Encrypted command data + CMAC |
| 3 | Sent to DESFire | > | 3B0200000020000002F2F06EA824B39CA2B1A38FD284EE8996AF35FC9714AD09CD1AB21D5D72C2DA3EFF1D0EFB2ECC94F60F2CF5ADE35E55D3ED2733D | Command header + encrypted data + CMAC to be sent to DESFire PICC. The comple frame is longer as the configured receive buffer of the DESFire, therefore chaining is needed. |
| 4 | Status Byte: Chaining | < | AF | status code: AF indicates that the DESFire PICC is ready for the next frame |
| 5 | Second part | > | AFBA9CC5E9 | Second part of the command frame |
| 6 | Status byte + CMAC | < | 009BA3CB8F59EADE44 | status byte 0x00 + CMAC |

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 7 | SAM_Remove_SM | > | 80AD001009009BA3CB8F59EADE4400 | SAM_Remove_SM command. As there is no data to decrypt, P2 = 0x01 (CommMode.MAC), the data field contains the status byte + 8 byte CMAC |
| 8 | Status word | < | 9000 | Status word of the SAM, indicating success |

### 2.8.6 Debit

The debit command reduces the value in the value file by a given amount.

CommMode.Full

Value: 0x01000000 (4 byte LSB first)

**Table 19. Example**

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | SAM_Apply_SM | > | 80AE00300702DC010100000000 | SAM_Apply_SM command. The command will be sent in CommMode.Full (P2 = 0x30). The play load of this command is the length of the DESFire command header (0x02), the DESFire command header itself (0xDC01, command code 0xDC + File number of the ValueFile 0x01) and the value to be debited. |
| 2 | Frame to be sent to DESFire PICC | < | C68F8F5E8D13CDF808353DC20C8353830D47153855B7B7439000 | The encrypted frame to be sent to DESFire PICC |
| 3 | Sent to DESFire | > | DC01C68F8F5E8D13CDF808353DC20C8353830D47153855B7B743 | The command header + the encrypted frame is sent to the DESFire PICC |
| 4 | Status byte + CMAC | < | 0013260828D0EF08E3 | Status byte + CMAC |
| 5 | SAM_Remove_SM | > | 80AD0010090013260828D0EF08E300 | SAM_Remove_SM command. As there is no data to decrypt, P2 = 0x01 (CommMode.MAC), the data field contains the status byte + 8 byte CMAC |
| 6 | Status word | < | 9000 | Status word of the SAM, indicating success |

### 2.8.7 CommitTransaction

The last command in this transaction commits the changes to the MIFARE DESFire PICC.

The command itself is sent in CommMode.MAC.

Table 20. Example

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | SAM_Apply_SM | > | 80AE001001C700 | SAM_Apply_SM command with P2 = 0x01 for CommMode.MAC. Data field of the command is the CommitTransaction command code 0xC7 |
| 2 | CMAC for command | < | 066E6991B44751BA9000 | 8 byte CMAC + status word |
| 3 | CommitTransaction + CMAC | > | C7066E6991B44751BA | Frame to be sent to DESFire PICC: command code + CMAC |
| 4 | Status byte + CMAC | < | 00A67A477A0EF1D152 | Response: Status byte + CMAC |
| 5 | SAM_Remove_ SM | > | 80AD00100900A67A477A0 EF1D15200 | SAM_Remove_SM command. As there is no data to decrypt, P2 = 0x01 (CommMode.MAC), the data field contains the status byte + 8 byte CMAC |
| 6 | Status word | < | 9000 | Status word of the SAM, indicating success |

## 2.9 Delegated Application Management

This section shows two examples of how to create delegated applications and change the DAM Keys on a DESFire EV2 Card. **This feature is exclusive for MIFARE DESFire EV2 and MIFARE DESFire EV3**. Details about how delegated applications work can be found in the MIFARE DESFire EV2 datasheet or in features and hints respectively in the MIFARE DESFire EV3 datasheet and features and hints.

### 2.9.1 Card issuer

The delegated application feature allows card issuers to sell/rent space on their cards to 3rd parties. Therefore, the card issuer needs to personalize the card in a way that allows a card user to install applications from 3rdparties, so called application provides, on the card. Therefore, some information needs to be secretly shared between the card issuer and the application provider, in order to allow the application to be installed in an appropriate slot on the card.

The following example gives an idea on how a MIFARE SAM AV3 can be used to handle the keys and data on the card issuer side.

### 2.9.2 Card issuer - example

In this example, a possible way to use SAM AV3 in combination with the delegated application feature on DESFire is shown. The example is intended to make use of all the possibilities on SAM AV3. It is not mandatory to use it exactly like this in a real application(for example, the DAM Default Key does not need to be stored in a SAM)

The structure in the SAM for this example shall look like the following (all Keys key type AES128):

**Table 21. Key structure of the Master-SAM**

| Key No | intended use | Key V 0x00 | Key V 0x01 | comment |
|---|---|---|---|---|
| 0x01 | PICC Keys | FFFFFFFFFFFFFFFF FFFFFFFFFFFFFF | AAAAAAAAAAAAA AAAAAAAAAAAAAA AAAAA | Key version 0x00: PICC MasterKey<br>Key version 0x01: DAMAuthKey |
| 0x02 | DAM Keys, Key Class PICC | 11111111111111111 1111111111111 | 2222222222222222 2222222222222222 | Key version 0x00: DAMMACKey<br>Key version 0x01: DAMEncKey<br>used only to activate DAM Keys on the MIFARE DESFire EV2 |
| 0x03 | DAM Default Key **AES192** | 000000000000000000 000000000000000000 000000000000 | - | DAM default Key, used to initialize the Application key(s).<br>Export secret needs to be allowed |
| 0x04 | DAM Keys, Key Class OfflineCrypto | see KeyNo 0x02 V 0x00 | see KeyNo 0x02 V 0x01 | Key version 0x00: DAMMACKey<br>Key version 0x01: DAMEncKey<br>same Key values as in KeyNo 0x02, but different Key Class. Those Keys can be used for OfflineCrypto purposes(generating the DAMMAC) |

First of all, the DAM Keys on the MIFARE DESFire EV2/EV3 card need to be activated. This happens with a ChangeKey Command, targeting the DAMAuthKey(0x10), DAMMACKey(0x11) and DAMEncKey(0x12). For this, the card needs to be authenticated with the PICC Master Key (KeyNo 0x00, KeyV 0x00). This is not shown here explicitly, see DESFire EV2 Authentication.

Additionally to above mentioned Keys, th "old Key" for the DAM Keys is needed for the ChangeKey command. for a fresh card, this key is all zeros. The examples use the third key entry in KeyNo 0x01, with KeyVersion 0x02.

Note: Due to reasons regarding backwards compatibility, the ChangeKeyEV2 command needs to be used, otherwise, the card key number 0x1y cannot be specified.

**Table 22. Change DAM Keys**

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | ChangeKey Command to SAM AV3 | > | 80C4002006001001020101 00 | Change Key EV2, KeySett number 0x00, card key number 0x10, old key no 0x01, version 0x02(all zeros AES128 key), new key number 0x01, version 0x01 (DAMAuthKey) |

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 2 | Response from SAM AV3 | < | 1789A708DAD267494A6A5A2C09DE9A4258C9DCB5A766BDE2DD432689F82981F31156CE173A7819589000 | cryptogram + SW1SW2 |
| 3 | Send to MIFARE DESFire EV2 | > | C600101789A708DAD267494A6A5A2C09DE9A4258C9DCB5A766BDE2DD432689F82981F31156CE173A781958 | ChangeKeyEV2 command, targeting KeySett 0x00, KeyNo 0x10 |
| 4 | response from MIFARE DESFire EV2 | < | 00A1EDB5BFD9E40AAE | 00 + MAC |
| 5 | SAM_remove_SM | > | 80AD00100900A1EDB5BFD9E40AAE00 | removes secure messaging |
| 6 | response | < | 9000 | success |

Repeat this also for the Card Keys 0x11 and 0x12. (0x11: use SAM Key 0x02 version 0x00; 0x12: use SAM Key 0x02 version 0x01)

Now, the card is prepared for creating delegated applications on it. In order for an application provide to be able to create delegated applications, the application provider also needs a DAMMAC, and an encrypted DAMDefaultKey. This ensures, that nobody except the dedicated application provider can create a delegated application on the card, as no one else can forge this DAMMAC, as the key is **only** known to the card issuer. Also, an application provider can only create a delegated application with the properties negotiated with the card issuer, as otherwise, the DAMMAC would not fit.

**Table 23. Create the Encrypted DAM DefaultKey and DAMMAC**

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 1 | Export value of DAMDefault Key | > | 80D6000002030000 | Dump secret key entry 0x03 version 0x01 |
| 2 | Value of the Key | < | 00000000000000000000000000000000000000000000000000009000 | Key value and SW1SW2 |
| 3 | DAMDefaultKey Version | = | 00 | |
| 4 | Get 7 random bytes | > | 8084000007 | for padding |
| 5 | Response | < | F5E876FEE275609000 | 7 Byte random + SW1SW2 |
| 6 | Input for Encryption | = | F5E876FEE27560000000000000000000000000000000000000000000000000 | random\|\|KeyValue\|\|KeyVersion |
| 7 | Activate OfflineKey | > | 80010000020401 | Activate KeyNo 0x04 version 0x01 for offline usage |
| 8 | Response | < | 9000 | success |
| 9 | EncipherOffline | > | 800E000020F5E876FEE27560000000000000000000000000000000000000000000000000000000000 | Encipher the value from step 6 using SAM_EncipherOffline |

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| 10 | Encrypted DAMDefaultKey | < | 9232C82A913FA1CFCDC7ED5EC63AB45CE991C06A1F485156DB8C3CDCB689BD279000 | Encrypted DAMDefaultKey EncK +SW1SW2 |
| 11 | Input for DAMMAC | = | C91234560000004000EF819232C82A913FA1CFCDC7ED5EC63AB45CE991C06A1F485156DB8C3CDCB689BD27 | DAMMAC = $MAC_{DAM}(K_{PICCDAMMAC}$,Cmd\|\|AID\|\| DAMSlotNo \|\|DAMSlotVersion\|\| QuotaLimit\|\|KeySett1\|\|KeySett2\|\| EncK) |
| 12 | Activate OfflineKey | > | 80010000020401 | Activate KeyNo 0x04 version 0x00 for offline usage |
| 13 | Response | < | 9000 | success |
| 14 | Generate DAMMAC | > | 807C00102BC91234560000004000EF819232C82A913FA1CFCDC7ED5EC63AB45CE991C06A1F485156DB8C3CDCB689BD2700 | Generate the DAMMAC with in step 11 generated input |
| 15 | returned MAC | < | FD5D929451161283E5B95C012261412B9000 | MAC + SW1SW2 |
| 16 | DAMMAC | = | 5D941683B901612B | The DAMMAC is truncated like in the EV2 secure messaging scheme: every second byte is taken for the final 8-byte MAC |

Everything needed from the card issuer is in place. The card issuer can now transfer the DAMAuthKey, the EncK and the DAMMAC to the application provider. This can be done with SAM AV3, or any other secure channel (not in scope of this example).

### 2.9.3 Create a Delegated Application

In this example, the application provider receives all needed data stored in a SAM AV3, called the "Slave-SAM".

**Table 24. Key structure of the Slave-SAM**

| KeyNo | intended use | Key V 0x00 | Comment |
|---|---|---|---|
| 0x11 | PICC Key | AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA | Key Version 0x00: DAMAuthKey<br>a KUC can be applied to this Key to limit the number of usages, hence the number of application installations.<br>**In this example, the Key value of this Key is not known to the application provider** |
| 0x12 | Data, KeyType **AES256** | 9232C82A913FA1CFCDC7ED5EC63AB45CE991C06A1F485156DB8C3CDCB689BD27 | This key entry is used to store the EncK, hence it is not a real key, just used for data storage. Export secret needs to be allowed. |

| KeyNo | intended use | Key V 0x00 | Comment |
|---|---|---|---|
| 0x13 | Data, KeyType **AES256** | 5D941683B901612B000000000000000000000000000000000000000000000000 | This key entry is used to store the DAMMAC, hence it is not a real key, just used for data storage. Export secret needs to be allowed. |

**Note:** For more security, Keys 0x12 and 0x13 should be only accessible if a host authentication with a secret key shared between card issuer and application provider is active. This can be achieved by setting KeyNoAEK to an additionally present host key, or the SAM master key.

**Table 25. Create delegated application**

| step | Indication | | Data / Message | Comment |
|---|---|---|---|---|
| The application provider can now authenticate with the DAMAuthKey using the SAM AV3(see DESFire EV2 Authentication). We assume, the data from Key 0x12 and 0x13 is already dumped. | | | | |
| 1 | Send to MIFARE DESFire EV2 | > | C912345600000004000EF81 | Create Delegated Application 0x123456 with parameters agreed with card issuer(Step 11 in Table 23) |
| 2 | Answer from MIFARE DESFire EV2 | < | AF | Expect more data |
| 3 | Send EncK + DAMMAC | > | AF9232C82A913FA1CFCDC7ED5EC63AB45CE991C06A1F485156DB8C3CDCB689BD275D941683B901612B306F1E706202ADB8 | AF \|\| EncK \|\| DAMMAC, dumped from SAM AV3 |
| 4 | Response from MIFARE DESFire EV2 | < | 00482CD78C55219DE2 | Response 0x00 + MAC |
| 5 | Remove SM using SAM AV3 | > | 80AD00100900482CD78C55219DE200 | Verifies the MAC using the current active Authentication |
| 6 | success | < | 9000 | Success, delegated Application created |

## 2.10 MIFARE DESFire Light and SAM AV3

MIFARE DESFire Light is a subset of the MIFARE DESFire family. For MIFARE SAM AV3 in S-mode, all examples shown in this document also apply for MIFARE DESFire Light. The only difference is, that commands sent to a MIFARE DESFire Light, need to be transferred in the ISO7816-4 APDU format(See datasheet). This is not influenced by the MIFARE SAM AV3 at all.

For X-mode, especially for the command *DESFire_AuthenticatePICC* and *DESFire_ChangeKeyPICC*, **bit 6** of P2 needs to be set, in order to use ISO7816-4 APDU format. Other than that, everything works according to a DESFire EV2.

# 3 References

1. **Data sheet – Data sheet of MIFARE SAM AV3**, document number 3235xx.
2. **System guidance manual – MF4SAM3 (MIFARE SAM AV3)**, document number 5385xx.
3. **Data sheet – MIFARE DESFire EV3**, document number 4489xx.
4. **Data sheet – MIFARE DESFire EV2**, document number 2260xx.
5. **Data sheet – MIFARE DESFire Light**, document number 4307xx.
6. **Application note – AN12695 – MIFARE SAM AV3 – Quick Start up Guide**, document number 5210xx, https://www.nxp.com/docs/en/application-note/AN12695.pdf.
7. **Application note – AN5212 – MIFARE SAM AV3 - Key Management and Personalization**, document number 5212xx.
8. **Application note – Symmetric Key Diversifications**, document number 1653xx.
9. **Application note – MIFARE SAM AV3 for MIFARE Classic EV1,** document number 1828xx.
10. **Application note – AN12704 – MIFARE SAM AV3 Host communication,** document number 5213xx, https://www.nxp.com/docs/en/application-note/AN12704.pdf.
11. **Application note – MIFARE DESFire EV3 Feature and hints,** document number 5881xx.
12. **Application note – MIFARE DESFire EV2 Feature and hints,** document number 3630xx.

# 4 Legal information

## 4.1 Definitions

**Draft** — A draft status on a document indicates that 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 in a draft version of a document and shall have no liability for the consequences of use of such information.

## 4.2 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.

**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.

**Evaluation products** — This product is provided on an "as is" and "with all faults" basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer. In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out the use of or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages. Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

**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.

**Security** — While NXP Semiconductors has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer's applications and products, and NXP Semiconductors accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

## 4.3 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.

## 4.4 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.

**MIFARE Plus** — is a trademark of NXP B.V.

**MIFARE Ultralight** — is a trademark of NXP B.V.

**MIFARE Classic** — is a trademark of NXP B.V.

**NXP** — wordmark and logo are trademarks 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: 14 July 2020**
**Document identifier: AN12696**
**Document number: 521513**